INTERNATIONAL STANDARD

ISO 16484-5

Seventh edition
2022-09

Building automation and control systems (BACS) —

Part 5: Data communication protocol

Systèmes d'automatisation et de gestion technique du bâtiment —
Partie 5: Protocole de communication de données

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Reference number
ISO 16484-5:2022(E)

© ISO 2022

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

COPYRIGHT PROTECTED DOCUMENT

© ISO 2022

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

© ISO 2022 – All rights reserved

CONTENTS

Page

FOREWORD xii
1 PURPOSE 1
2 SCOPE 1
3 DEFINITIONS 1
3.1 Terms Adopted from International Standards 1
3.2 Terms Defined for this Standard 3
3.3 Abbreviations and Acronyms Used in this Standard 7
4 BACnet PROTOCOL ARCHITECTURE 11
4.1 The BACnet Collapsed Architecture 12
4.2 BACnet Network Topology 14
4.3 Security 14
5 THE APPLICATION LAYER 16
5.1 The Application Layer Model 16
5.2 Segmentation of BACnet Messages 20
5.3 Transmission of BACnet APDUs 21
5.4 Application Protocol State Machines 25
5.5 Application Protocol Time Sequence Diagrams 41
5.6 Application Layer Service Conventions 50
6 THE NETWORK LAYER 52
6.1 Network Layer Service Specification 52
6.2 Network Layer PDU Structure 54
6.3 Messages for Multiple Recipients 59
6.4 Network Layer Protocol Messages 60
6.5 Network Layer Procedures 64
6.6 BACnet Routers 66
6.7 Point-To-Point Half-Routers 71
7 DATA LINK/PHYSICAL LAYERS: Ethernet (ISO 8802-3) LAN 76
7.1 The Use of ISO 8802-2 Logical Link Control (LLC) 76
7.2 Parameters Required by the LLC Primitives 76
7.3 Parameters Required by the MAC Primitives 76
7.4 Physical Media 76
8 DATA LINK/PHYSICAL LAYERS: ARCNET (ATA 878.1) LAN 77
8.1 The Use of ISO 8802-2 Logical Link Control (LLC) 77
8.2 Parameters Required by the LLC Primitives 77
8.3 Mapping the LLC Services to the ARCNET MAC Layer 77
8.4 Parameters Required by the MAC Primitives 77
8.5 Physical Media 77
9 DATA LINK/PHYSICAL LAYERS: MASTER-SLAVE/TOKEN-PASSING (MS/TP) LAN 79
9.1 Service Specification 79
9.2 Physical Layer 81
9.3 MS/TP Frame Format 90
9.4 Overview of the MS/TP Network 92
9.5 MS/TP Medium Access Control 92
9.6 Cyclic Redundancy Check (CRC) 111
9.7 Interfacing MS/TP LANs with Other BACnet LANs 112
9.8 Responding BACnet User Processing of Messages from MS/TP 112
9.9 Repeaters 113
9.10 COBS (Consistent Overhead Byte Stuffing) Encoding 114
9.11 Documenting MS/TP Device Design Choices 118
10 DATA LINK/PHYSICAL LAYERS: POINT-TO-POINT (PTP) 119
10.1 Overview 119
10.2 Service Specification 119
10.3 Point-to-Point Frame Format 124
10.4 PTP Medium Access Control Protocol 126
11 DATA LINK/PHYSICAL LAYERS: LonTalk (ISO/IEC 14908.1) LAN 147
11.1 The Use of ISO 8802-2 Logical Link Control (LLC) 147
11.2 Parameters Required by the LLC Primitives 147

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

11.3 Mapping the LLC Services to the LonTalk Application Layer 147
11.4 Parameters Required by the Application Layer Primitives 147
11.5 Physical Media 148
12 MODELING CONTROL DEVICES AS A COLLECTION OF OBJECTS 149
12.1 Object Characteristics and Requirements 149
12.2 Analog Input Object Type 155
12.3 Analog Output Object Type 162
12.4 Analog Value Object Type 169
12.5 Averaging Object Type 177
12.6 Binary Input Object Type 181
12.7 Binary Output Object Type 188
12.8 Binary Value Object Type 197
12.9 Calendar Object Type 205
12.10 Command Object Type 208
12.11 Device Object Type 214
12.12 Event Enrollment Object Type 226
12.13 File Object Type 234
12.14 Group Object Type 238
12.15 Life Safety Point Object Type 241
12.16 Life Safety Zone Object Type 248
12.17 Loop Object Type 255
12.18 Multi-state Input Object Type 265
12.19 Multi-state Output Object Type 271
12.20 Multi-state Value Object Type 278
12.21 Notification Class Object Type 285
12.22 Program Object Type 291
12.23 Pulse Converter Object Type 298
12.24 Schedule Object Type 306
12.25 Trend Log Object Type 313
12.26 Access Door Object Type 323
12.27 Event Log Object Type 332
12.28 Load Control Object Type 340
12.29 Structured View Object Type 350
12.30 Trend Log Multiple Object Type 355
12.31 Access Point Object Type 365
12.32 Access Zone Object Type 382
12.33 Access User Object Type 390
12.34 Access Rights Object Type 394
12.35 Access Credential Object Type 400
12.36 Credential Data Input Object Type 410
12.37 CharacterString Value Object Type 416
12.38 DateTime Value Object Type 423
12.39 Large Analog Value Object Type 429
12.40 BitString Value Object Type 437
12.41 OctetString Value Object Type 444
12.42 Time Value Object Type 448
12.43 Integer Value Object Type 454
12.44 Positive Integer Value Object Type 462
12.45 Date Value Object Type 470
12.46 DateTime Pattern Value Object Type 476
12.47 Time Pattern Value Object Type 482
12.48 Date Pattern Value Object Type 488
12.49 Deleted Clause 494
12.50 Global Group Object Type 495
12.51 Notification Forwarder Object Type 502
12.52 Alert Enrollment Object Type 510
12.53 Channel Object Type 514
12.54 Lighting Output Object Type 524
12.55 Binary Lighting Output Object Type 539
12.56 Network Port Object Type 549

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

12.57 Timer Object Type 573
12.58 Elevator Group Object Type 585
12.59 Lift Object Type 589
12.60 Escalator Object Type 601
12.61 Accumulator Object Type 608
12.62 Staging Object Type 618
12.63 Audit Reporter Object Type 628
12.64 Audit Log Object Type 633
13 ALARM AND EVENT SERVICES 640
13.1 Change of Value Reporting 640
13.2 Event Reporting 645
13.3 Event Algorithms 656
13.4 Fault Algorithms 685
13.5 AcknowledgeAlarm Service 692
13.6 ConfirmedCOVNotification Service 694
13.7 UnconfirmedCOVNotification Service 696
13.8 ConfirmedEventNotification Service 697
13.9 UnconfirmedEventNotification Service 700
13.10 GetAlarmSummary Service 703
13.11 GetEnrollmentSummary Service 705
13.12 GetEventInformation Service 708
13.13 LifeSafetyOperation Service 710
13.14 SubscribeCOV Service 712
13.15 SubscribeCOVProperty Service 715
13.16 SubscribeCOVPropertyMultiple Service 718
13.17 ConfirmedCOVNotificationMultiple Service 723
13.18 UnconfirmedCOVNotificationMultiple Service 726
13.19 AuditLogQuery 728
13.20 ConfirmedAuditNotification 732
13.21 UnconfirmedAuditNotification 733
14 FILE ACCESS SERVICES 734
14.1 AtomicReadFile Service 734
14.2 AtomicWriteFile Service 737
15 OBJECT ACCESS SERVICES 740
15.1 AddListElement Service 740
15.2 RemoveListElement Service 742
15.3 CreateObject Service 744
15.4 DeleteObject Service 747
15.5 ReadProperty Service 748
15.6 Deleted Clause 750
15.7 ReadPropertyMultiple Service 751
15.8 ReadRange Service 754
15.9 WriteProperty Service 762
15.10 WritePropertyMultiple Service 764
15.11 WriteGroup Service 767
16 REMOTE DEVICE MANAGEMENT SERVICES 769
16.1 DeviceCommunicationControl Service 769
16.2 ConfirmedPrivateTransfer Service 771
16.3 UnconfirmedPrivateTransfer Service 773
16.4 ReinitializeDevice Service 774
16.5 ConfirmedTextMessage Service 776
16.6 UnconfirmedTextMessage Service 778
16.7 TimeSynchronization Service 779
16.8 UTCTimeSynchronization Service 780
16.9 Who-Has and I-Have Services 781
16.10 Who-Is and I-Am Services 783
16.11 Who-Am-I and You-Are Services 785
17 VIRTUAL TERMINAL SERVICES 787
17.1 Virtual Terminal Model 787
17.2 VT-Open Service 791

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.3 VT-Close Service 793
17.4 VT-Data Service 794
17.5 Default Terminal Characteristics 796
18 ERROR, REJECT, and ABORT CODES 800
18.1 Error Class - DEVICE 800
18.2 Error Class - OBJECT 800
18.3 Error Class - PROPERTY 801
18.4 Error Class - RESOURCES 802
18.5 Error Class - SECURITY 803
18.6 Error Class - SERVICES 804
18.7 Error Class - COMMUNICATION 805
18.8 Error Class - VT 809
18.9 Reject Reason 810
18.10 Abort Reason 810
18.11 Confirmed Service Common Errors 811
19 BACnet PROCEDURES 812
19.1 Backup and Restore 812
19.2 Command Prioritization 817
19.3 Device Restart Procedure 822
19.4 Determining Maximum Conveyable APDU 823
19.5 Value Source Mechanism 825
19.6 Audit Logging 827
19.7 Unconfigured Device Discovery and Address Assignment 838
20 ENCODING BACnet PROTOCOL DATA UNITS 839
20.1 Encoding the Fixed Part of BACnet APDUs 840
20.2 Encoding the Variable Part of BACnet APDUs 851
21 FORMAL DESCRIPTION OF APPLICATION PROTOCOL DATA UNITS 866
21.1 APDU Definitions 866
21.2 Confirmed Service Productions 868
21.3 Unconfirmed Service Productions 878
21.4 Error Productions 881
21.5 Application Types 891
21.6 Base Types 892
22 CONFORMANCE AND INTEROPERABILITY 960
22.1 Conformance to BACnet 960
22.2 BACnet Interoperability 961
23 EXTENDING BACnet TO ACCOMMODATE VENDOR PROPRIETARY INFORMATION 963
23.1 Extending Enumeration Values 963
23.2 Using the PrivateTransfer Services to Invoke Non-Standardized Services 964
23.3 Adding Proprietary Properties to a Standardized Object 964
23.4 Adding Proprietary Object Types to BACnet 965
23.5 Restrictions on Extending BACnet 965
24 DELETED CLAUSE 966
25 REFERENCES 967
ANNEX A - PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT (NORMATIVE) 971
ANNEX B - GUIDE TO SPECIFYING BACnet DEVICES (INFORMATIVE) 974
ANNEX C - Removed 975
ANNEX D - Removed 976
ANNEX E - EXAMPLES OF BACnet APPLICATION SERVICES (INFORMATIVE) 977
E.1 Alarm and Event Services 977
E.2 File Access Services 981
E.3 Object Access Services 983
E.4 Remote Device Management Services 989
E.5 Virtual Terminal Services 993
ANNEX F - EXAMPLES OF APDU ENCODING (INFORMATIVE) 995
F.1 Example Encodings for Alarm and Event Services 995
F.2 Example Encodings for File Access Services 1006
F.3 Example Encodings for Object Access Services 1008
F.4 Example Encodings for Remote Device Management Services 1018
F.5 Example Encodings for Virtual Terminal Services 1024

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

ANNEX G - CALCULATION OF CRC (INFORMATIVE) 1026
G.1 Calculation of the Header CRC 1026
G.2 Calculation of the Data CRC 1032
G.3 Calculation of the Encoded CRC-32K 1036
ANNEX H - COMBINING BACnet NETWORKS WITH NON-BACnet NETWORKS (NORMATIVE) 1040
H.1 BACnet Gateways 1040
H.2 Requirements and Best Practices for BACnet Gateway Implementations 1040
H.3 Using BACnet with the DARPA Internet Protocols 1042
H.4 Using BACnet with the IPX Protocol 1044
H.5 Using BACnet with EIB/KNX 1045
H.6 Using BACnet with the Former BACnet/WS Web Services Interface Defined by Annex N 1055
H.7 Virtual MAC Addressing 1057
ANNEX I - COMMANDABLE PROPERTIES WITH MINIMUM ON AND OFF TIMES (INFORMATIVE) 1059
ANNEX J - BACnet/IP (NORMATIVE) 1061
J.1 General 1061
J.2 BACnet Virtual Link Layer 1062
J.3 BACnet/IP Directed Messages 1065
J.4 BACnet/IP Broadcast Messages 1065
J.5 Addition of Foreign B/IP Devices to an Existing B/IP Network 1068
J.6 Routing Between B/IP and non-B/IP BACnet Networks 1069
J.7 Routing Between Two B/IP BACnet Networks 1070
J.8 Use of IP Multicast within BACnet/IP 1075
ANNEX K - BACnet INTEROPERABILITY BUILDING BLOCKS (BIBBs) (NORMATIVE) 1077
K.1 Data Sharing BIBBs 1077
K.2 Alarm and Event Management BIBBs 1098
K.3 Scheduling BIBBs 1111
K.4 Trending BIBBs 1115
K.5 Device Management BIBBs 1118
K.6 Network Management BIBBs 1125
K.7 Gateway BIBBs 1128
K.8 Audit Reporting BIBBs 1129
ANNEX L - DESCRIPTIONS AND PROFILES OF STANDARDIZED BACnet DEVICES (NORMATIVE) 1131
L.1 Operator Interface Profiles 1131
L.2 Life Safety Operator Interface Profiles 1134
L.3 Access Control Operator Interface Profiles 1136
L.4 Controller Profiles 1139
L.5 Life Safety Controller Profiles 1142
L.6 Access Control Controller Profiles 1143
L.7 Miscellaneous Profiles 1145
L.8 BACnet General (B-GENERAL) Profile 1148
L.9 Lighting Operator Interface Profiles 1149
L.10 Lighting Control Station Profiles 1151
L.11 Lighting Controller Profiles 1152
L.12 Elevator Operator Interface Profiles 1154
L.13 Elevator Controller Profiles 1156
ANNEX M - GUIDE TO EVENT NOTIFICATION PRIORITY ASSIGNMENTS (INFORMATIVE) 1159
M.1 Life Safety Message Group (0 - 31) 1159
M.2 Property Safety Message Group (32 - 63) 1160
M.3 Supervisory Message Group (64 - 95) 1160
M.4 Trouble Message Group (96 - 127) 1161
M.5 Miscellaneous Higher Priority Message Group (128 - 191) 1162
M.6 Miscellaneous Lower Priority Message Group (192 - 255) 1162
ANNEX N - FORMER BACnet/WS WEB SERVICES INTERFACE (INFORMATIVE) 1163
N.1 Data Model 1163
N.2 Paths 1164
N.3 Normalized Points 1164
N.4 Reference Nodes 1165
N.5 Localization 1165
N.6 Security 1165
N.7 Sessions 1166

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

N.8 Attributes 1166
N.9 Standard Nodes 1172
N.10 Encodings 1172
N.11 Service Options 1173
N.12 Services 1176
N.13 Errors 1193
N.14 Extending BACnet/WS 1194
ANNEX O - BACnet OVER ZigBee AS A DATA LINK LAYER (NORMATIVE) 1195
O.1 General 1195
O.2 ZigBee Overview 1195
O.3 Definitions 1196
O.4 Unicast Addressing 1196
O.5 Broadcast Addressing 1196
O.6 BACnet/ZigBee Data Link Layer (BZLL) 1197
O.7 Maximum Payload Size 1200
O.8 Vendor Specific Commands 1200
ANNEX P - BACnet ENCODING OF STANDARD AUTHENTICATION FACTOR FORMATS (NORMATIVE) 1201
ANNEX Q - XML DATA FORMATS (NORMATIVE) 1208
Q.1 Introduction 1208
Q.2 XML Document Structure 1211
Q.3 Expressing Data 1215
Q.5 Expressing Values 1216
Q.6 Binary Encoding and Access Rules 1218
Q.7 Extensibility 1218
Q.8 BACnet URI Scheme 1219
ANNEX R - MAPPING NETWORK LAYER ERRORS (NORMATIVE) 1220
ANNEX S - Removed 1221
ANNEX T - COBS (CONSISTENT OVERHEAD BYTE STUFFING) FUNCTIONS (INFORMATIVE) 1222
T.1 Preparing a COBS-Encoded MS/TP Frame for Transmission 1222
T.2 Decoding an Extended MS/TP Frame upon Reception 1224
T.3 Example COBS-Encoded Frame - Who-Has Service 1226
ANNEX U - BACnet/IPv6 (NORMATIVE) 1228
U.1 General 1228
U.2 BACnet/IPv6 BACnet Virtual Link Layer 1229
U.3 BACnet/IPv6 Directed Messages 1233
U.4 BACnet/IPv6 Broadcast Messages 1233
U.5 BACnet /IPv6 VMAC Table Management 1238
ANNEX V - MIGRATION FROM SOAP SERVICES (INFORMATIVE) 1239
V.1 Services 1239
V.2 Service Options 1241
ANNEX W - BACnet/WS RESTful WEB SERVICES INTERFACE (NORMATIVE) 1242
W.1 Data Model 1242
W.2 Paths 1242
W.3 Security 1243
W.4 Sessions 1252
W.5 Standard Data Items 1252
W.6 Metadata 1257
W.7 Functions 1257
W.8 Query Parameters 1259
W.9 Representation of Data 1261
W.10 Representation of Metadata 1261
W.11 Representation of Logs 1262
W.12 Filtering Items 1266
W.13 Limiting Number of Items 1268
W.14 Selecting Children 1268
W.15 Controlling Content of Data Representations 1269
W.16 Specifying Ranges 1272
W.17 Localized Values 1274
W.18 Accessing Individual Tags and Bits 1275
W.19 Semantics 1275

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

W.20 Links and Relationships ... 1275
W.21 Foreign XML and Other Media Types ... 1275
W.22 Logical Modeling ... 1276
W.23 Mapped Modeling ... 1277
W.24 Commandability ... 1277
W.25 Writability and Visibility ... 1277
W.26 Working with Optional Data ... 1279
W.27 Working with Optional Metadata ... 1279
W.28 Creating Data ... 1280
W.29 Setting Data ... 1280
W.30 Deleting Data ... 1282
W.31 Parentally Inherited Values ... 1283
W.32 Concurrency Control ... 1283
W.33 Server Support for Data Definitions ... 1283
W.34 Server Support for Metadata ... 1284
W.35 Client Implementation Guidelines ... 1285
W.36 Subscriptions ... 1285
W.37 Reading Multiple Resources ... 1287
W.38 Writing Multiple Resources ... 1288
W.39 Mapping of BACnet Systems ... 1288
W.40 Errors ... 1292
W.41 Examples ... 1294
ANNEX X - EXTENDED DISCOVERY OF DEVICES, PROFILES, AND VIEWS (NORMATIVE) ... 1321
X.1 Profiles ... 1321
X.2 xdd Files ... 1322
X.3 Example of Definition of Objects, Properties, and Datatypes ... 1323
X.4 Views ... 1324
X.5 PICS Declarations ... 1330
ANNEX Y - ABSTRACT DATA MODEL (NORMATIVE) ... 1331
Y.1 Model Components ... 1331
Y.2 Trees ... 1333
Y.3 Base Types ... 1335
Y.4 Common Metadata ... 1335
Y.5 Named Values ... 1349
Y.6 Named Bits ... 1352
Y.7 Primitive Values ... 1353
Y.8 Range Restrictions ... 1355
Y.9 Engineering Units ... 1356
Y.10 Length Restrictions ... 1357
Y.11 Collections ... 1359
Y.12 Primitive Data ... 1360
Y.13 Constructed Data ... 1364
Y.14 Data of Undefined Type ... 1368
Y.15 Logical Modeling ... 1368
Y.16 Links ... 1368
Y.17 Change Indications ... 1370
Y.18 Definitions, Types, Instances, and Inheritance ... 1370
Y.19 Data Revisions ... 1376
Y.20 BACnet-Specific Base Types ... 1379
Y.21 BACnet-Specific Metadata ... 1380
ANNEX Z - JSON DATA FORMATS (NORMATIVE) ... 1384
Z.1 Introduction ... 1384
Z.2 JSON Document Structure ... 1387
Z.3 Expressing Data ... 1390
Z.4 Expressing Metadata ... 1390
Z.5 Expressing Values ... 1392
Z.6 Extensibility ... 1393
ANNEX AA – TIME SERIES DATA EXCHANGE FILE FORMAT (NORMATIVE) ... 1395
AA.1 File Format ... 1395
AA.2 Representation of Data ... 1395

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

AA.3 File Generation 1396
AA.4 Example Files 1397
ANNEX AB – BACnet Secure Connect (NORMATIVE) 1398
AB.1 BACnet Secure Connect Data link 1398
AB.2 BACnet/SC Virtual Link Layer Messages 1404
AB.3 BACnet/SC Node Operation 1414
AB.4 Node Switch and Direct Connections 1416
AB.5 Hub Function and Hub Connector 1419
AB.6 BACnet/SC Connections 1422
AB.7 Application of WebSockets in BACnet/SC 1426
HISTORY OF REVISIONS 1432

NOTE

Approved addenda, errata, or interpretations for this standard can be downloaded free of charge from the ASHRAE Web site at www.ashrae.org/technology.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 205, *Building environment design*, in collaboration with the European Committee for Standardization (CEN) Technical Committee CEN/TC 247, *Building automation, controls and building management,* in accordance with the Agreement on technical cooperation between ISO and CEN (Vienna Agreement) and with the American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc. (ASHRAE).

This seventh edition cancels and replaces the sixth edition (ISO 16484-5:2017), which has been technically revised. It also incorporates the Amendment ISO 16484-5:2017/Amd 1:2020.

The main changes are as follows:

- See the detailed list of changes on pages 1432 to 1451.

A list of all parts in the ISO 16484 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

FOREWORD

BACnet, the ASHRAE building automation and control networking protocol, has been designed specifically to meet the communication needs of building automation and control systems for applications such as heating, ventilating, and air-conditioning control, lighting control, access control, and fire detection systems. The BACnet protocol provides mechanisms by which computerized equipment of arbitrary function may exchange information, regardless of the particular building service it performs. As a result, the BACnet protocol may be used by head-end computers, general-purpose direct digital controllers, and application specific or unitary controllers with equal effect.

The motivation for this Standard was the widespread desire of building owners and operators for "interoperability," the ability to integrate equipment from different vendors into a coherent automation and control system - and to do so competitively. To accomplish this, the Standard Project Committee (SPC) solicited and received input from dozens of interested firms and individuals; reviewed all relevant national and international data communications standards, whether de facto or the result of committee activity; and spent countless hours in debate and discussion of the pros and cons of each element of the protocol.

What has emerged from the committee deliberations is a network protocol model with these principal characteristics:

(a) All network devices (except MS/TP slaves) are peers, but certain peers may have greater privileges and responsibilities than others.

(b) Each network device is modeled as a collection of network-accessible, named entities called "objects." Each object is characterized by a set of attributes or "properties." While this Standard prescribes the most widely applicable object types and their properties, implementors are free to create additional object types if desired. Because the object model can be easily extended, it provides a way for BACnet to evolve in a backward compatible manner as the technology and building needs change.

(c) Communication is accomplished by reading and writing the properties of particular objects and by the mutually acceptable execution of other protocol "services." While this Standard prescribes a comprehensive set of services, mechanisms are also provided for implementors to create additional services if desired.

(d) Because of this Standard's adherence to the ISO concept of a "layered" communication architecture, the same messages may be exchanged using various network access methods and physical media. This means that BACnet networks may be configured to meet a range of speed and throughput requirements with commensurately varying cost. Multiple BACnet networks can be interconnected within the same system forming an internetwork of arbitrarily large size. This flexibility also provides a way for BACnet to embrace new networking technologies as they are developed.

BACnet was designed to gracefully improve and evolve as both computer technology and demands of building automation systems change. Upon its original publication in 1995, a Standing Standards Project Committee was formed to deliberate enhancements to the protocol under ASHRAE rules for "continuous maintenance." Much has happened since the BACnet standard was first promulgated. BACnet has been translated into Chinese, Japanese, and Korean, and embraced across the globe. BACnet devices have been designed, built and deployed on all seven continents. Suggestions for enhancements and improvements have been continually received, deliberated, and, ultimately, subjected to the same consensus process that produced the original standard. This publication is the result of those deliberations and brings together all of the corrections, refinements, and improvements that have been adopted.

Among the features that have been added to BACnet are: increased capabilities to interconnect systems across wide area networks using Internet Protocols, new objects and services to support fire detection, other life safety applications, lighting, physical access control, and elevator monitoring, capabilities to backup and restore devices, standard ways to collect trend data, new tools to make specifying BACnet systems easier, a mechanism for making interoperable extensions to the standard visible, and many others. The successful addition of these features demonstrates that the concept of a protocol deliberately crafted to permit extension of its capabilities over time as technology and needs change is viable and sound.

The latest update to this standard includes many new capabilities and several clarifications. These new capabilities include lighting and elevator device profiles that provide a standardized description for these types of devices along with audit reporting and logging to capture auditable actions. BACnet Secure Connect, a BACnet datalink layer, was also added. This datalink is fully compatible with all existing BACnet datalinks and provides an interoperable secure path for BACnet communications.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

All communication protocols are, in the end, a collection of arbitrary solutions to the problems of information exchange and all are subject to change as time and technology advance. BACnet is no exception. Still, it is the hope of those who have contributed their time, energies, and talents to this work that BACnet will help to fulfill, in the area of building automation and control, the promise of the information age for the public good!

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Building automation and control systems (BACS) —

Part 5:
Data communication protocol

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

1 PURPOSE

The purpose of this standard is to define data communication services and protocols for computer equipment used for monitoring and control of HVAC&R and other building systems and to define, in addition, an abstract, object-oriented representation of information communicated between such equipment, thereby facilitating the application and use of digital control technology in buildings.

2 SCOPE

2.1 This protocol provides a comprehensive set of messages for conveying encoded binary, analog, and alphanumeric data between devices including, but not limited to:

(a) hardware binary input and output values,
(b) hardware analog input and output values,
(c) software binary and analog values,
(d) text string values,
(e) schedule information,
(f) alarm and event information,
(g) files, and
(h) control logic.

2.2 This protocol models each building automation and control computer as a collection of data structures called "objects," the properties of which represent various aspects of the hardware, software, and operation of the device. These objects provide a means of identifying and accessing information without requiring knowledge of the details of the device's internal design or configuration.

3 DEFINITIONS

3.1 Terms Adopted from International Standards

The following terms used in this standard are defined by international standards or draft standards for open system interconnection (OSI). The definitions are repeated here and a reference to the appropriate standard is provided. Clause 25 contains the titles of all national and international standards referenced in this clause and elsewhere in this standard. Words or phrases in italics refer to terms defined elsewhere in this clause.

abstract syntax: the specification of application layer data or application-protocol-control-information by using notation rules which are independent of the encoding technique used to represent them (ISO 8822).

application: a set of a USER's information processing requirements (ISO 8649).

application-entity: the aspects of an application-process pertinent to OSI (ISO 7498).

application-process: an element within a real open system which performs the information processing for a particular application (ISO 7498).

© ISO 2022 – All rights reserved

application-protocol-control-information: information exchanged between application-entities, using presentation services, to coordinate their joint operation (ISO 9545).

application-protocol-data-unit: a unit of data specified in an application protocol and consisting of application-protocol-control-information and possibly application-user-data (ISO 9545).

application-service-element: that part of an application-entity which provides an OSI environment capability, using underlying services when appropriate (ISO 7498).

concrete syntax: those aspects of the rules used in the formal specification of data which embody a specific representation of that data (ISO 7498).

confirm (primitive): a representation of an interaction in which a service-provider indicates, at a particular service-access-point, completion of some procedure previously invoked, at that service-access-point, by an interaction represented by a request primitive (ISO TR 8509).

indication (primitive): a representation of an interaction in which a service-provider either

(a) indicates that it has, on its own initiative, invoked some procedure; or

(b) indicates that a procedure has been invoked by the service-user at the peer service-access-point (ISO TR 8509).

peer-entities: entities within the same layer (ISO 7498).

persistent: when used to describe property values, the property value is stored in non-volatile storage so as to survive device resets and power failures.

real open system: a real system which complies with the requirements of OSI standards in its communication with other real systems (ISO 7498).

real system: a set of one or more computers, the associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that forms an autonomous whole capable of performing information processing and/or information transfer (ISO 7498).

request (primitive): a representation of an interaction in which a service-user invokes some procedure (ISO TR 8509).

response (primitive): a representation of an interaction in which a service-user indicates that it has completed some procedure previously invoked by an interaction represented by an indication primitive (ISO TR 8509).

(N)-service-access-point: the point at which (N)-services are provided by an (N)-entity to an (N+1)-entity (ISO 7498).

(N)-service-data-unit: an amount of (N)-interface-data whose identity is preserved from one end of an (N)-connection to the other (ISO 7498).

service-user: an entity in a single open system that makes use of a service through service-access-points (ISO TR 8509).

service-primitive; **primitive**: an abstract, implementation-independent representation of an interaction between the service-user and the service-provider (ISO TR 8509).

service-provider: an abstract of the totality of those entities which provide a service to peer service-users (ISO TR 8509).

transfer-syntax: that concrete syntax used in the transfer of data between open systems (ISO 7498).

user element: the representation of that part of an application-process which uses those application-service-elements needed to accomplish the communications objectives of that application-process (ISO 7498).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

3.2 Terms Defined for this Standard

access control: a method for regulating or restricting access to network resources.

access rights (physical access control): the access privileges granted to a credential.

access user (physical access control): the person or asset holding one or more credentials.

alarm: 1. An annunciation, either audible or visual or both, that alerts an operator to an off-normal condition that may require corrective action. **2.** An abnormal condition detected by a device or controller that implements a rule or logic specifically designed to look for that condition.

alarm-acknowledgment: the process of indicating that a human operator has seen and responded to an event notification.

algorithmic change reporting: the detection and reporting of an alarm or event, based on an algorithm specified in an Event Enrollment object. See intrinsic reporting.

authentication: the act of verifying identity

authentication factor: a data element of the credential which is used to verify a credential's identity.

authorization (network security): the control of access to network resources based on known identity and access rules.

authorization (physical access control): the process of determining whether the access user is permitted to enter a protected zone through an access controlled point.

BACnet address: address format used by the BACnet network layer, as defined in Clause 6, consisting of a network number and a MAC address.

BACnet device: any device, real or virtual, that supports digital communication using the BACnet protocol.

BACnet network: a network of BACnet devices that share the MAC or VMAC address space under a particular BACnet network number.

BACnet-user: that portion of an application-process that is represented by the BACnet user element.

blink-warn: in lighting control, typically a method of notifying room occupants of an impending automated command to turn off the lights whereby the lights may be blinked, once or multiple times, or an audible signal is generated. After the warning occurs, the room lights are held on for a grace period to allow occupants to either safely leave the room or to initiate a request to keep the room lights on. Also known as "flick warn" or "flash warn."

bridge: a device that connects two or more segments at the physical and data link layers. This device may also perform message filtering based upon MAC layer addresses.

broadcast: a message sent as a single unit, which may apply to more than one device.

change of state: an event that occurs when a measured or calculated Boolean or discrete enumerated value changes.

change of value: an event that occurs when a measured or calculated analog value changes by a predefined amount.

client: a system or device that makes use of another device for some particular purpose via a service request instance. A client requests service from a server.

confidentiality: the property that information is not made available or disclosed to unauthorized individuals, entities, or processes.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

configurable: a property, setting, or value in a device is configurable if it can be changed via BACnet services or some other method. A property, setting, or value that is one-time writable or not changeable in situ is not considered to be configurable.

context: a set of data or information that completely describes a particular communication environment at a particular point in time.

controller: a device for regulation or management of a system or component.

credential (physical access control): the combination of authentication factors and access rights.

date pattern: a date that may contain one or more unspecified octets or special date values.

directly connected network: a network that is accessible from a BACnet router without messages being relayed through an intervening BACnet router. A PTP connection is to a directly connected network if the PTP connection is currently active and no intervening BACnet router is used.

download: a particular type of file transfer that refers to the transfer of an executable program or database to a remote device where it may be executed.

encrypted message: a message that is wrapped in a security header, signed, and encrypted.

entity: something that has a separate and distinct existence. An identifiable item that is described by a set or collection of properties.

error detection: a procedure used to identify the presence of errors in a communication.

error recovery: a procedure invoked in response to a detected error that permits the information exchange to continue.

event algorithm: the rules that determine when an event-initiating object changes between normal and offnormal states. The event algorithm has no impact on an event-initiating object's transition to or from fault.

event-initiating object: an object that generates event notifications.

event-notification-distribution: the process that a notification-server performs in the determination of notification-clients and in the sending of notifications to notification-clients when an event-initiating object changes the event or acknowledgment state.

event notification message: a ConfirmedEventNotification or UnconfirmedEventNotification service request used to indicate a change in the event or acknowledgment state of an event-initiating object.

event-state-detection: the process of executing an event-initiating object's event algorithm and monitoring the object's Reliability property to detect changes in the object's event state.

event-summarization: the querying of event-initiating objects in a device through one of the event summarization services to determine those that meet specific event state or reporting conditions.

fading: the gradual increase or decrease of the actual output from one setting to another over a fixed period of time.

gateway: a device that connects two or more dissimilar networks, permitting information exchange between them.

global: pertaining to all devices or nodes on a communication internetwork.

global broadcast: a message addressed to all devices or nodes on all networks in a BACnet internet.

half router: a device or node that can participate as one partner in a PTP connection. The two half-router partners that form an active PTP connection together make up a single BACnet router.

initialization: the process of establishing a known state, usually from a power up condition. Initialization may require re-establishment of a node's logical or physical address.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

integrity: the property that data has not been altered or destroyed in an unauthorized manner.

internetwork: a set of two or more networks interconnected by BACnet routers. In a BACnet internetwork interconnected by BACnet routers, there exists exactly one message path between any two nodes.

intrinsic reporting: the detection and reporting of an alarm or event, based on an algorithm defined as part of the object type specification. No external reference to an Event Enrollment is involved. See algorithmic change reporting.

invalid character: Character-level encoding within a BACnet character string that is not defined by the governing body of the character encoding, or that breaks the rules of the governing body, or is not allowed by the rules of BACnet such as a null character when printable characters are required.

inverted network: a BACnet internetwork where two or more BACnet networks are connected by a BACnet network with an NPDU size smaller than the networks it joins.

key: a sequence of symbols that controls the operations of encipherment and decipherment.

link: a communication facility or medium over which nodes can communicate at the IP link layer.

local: pertaining to devices on the same network as the referenced device.

local broadcast: a message addressed to all devices or nodes on the same BACnet network as the originator.

medium: the physical transmission entity. Typical media are twisted-pair wire, fiber optic cable, and coaxial cable.

medium access control: a process used to maintain order and provide access to the communication medium.

multicast domain: A group of B/IPv6 nodes that can communicate with each other using a particular IPv6 multicast address and UDP port combination.

network: a set of one or more segments interconnected by bridges that have the same network address.

network resource: any physical or logical entity that may be accessed via a communication medium.

node: an addressable device connected to the communication medium.

notification-client: a BACnet device that receives and processes event notification messages.

notification-server: a BACnet device that contains event-initiating objects and performs event-notification-distribution.

numeric quantity: a value that is the magnitude of a quantity. A representation of a specific amount such as temperature, volume or counts.

object: a specific instance of an object type. While an object type is identified by a unique Object_Type property, an object is identified by its Object_Identifier property.

object profile: an object profile is a means of defining objects beyond those defined in Clause 12. A profile defines the set of properties, behavior, and/or requirements for a proprietary object, or for proprietary extensions to a standard object.

object type: a generic classification of data that is defined by a set of properties.

operator authentication: the corroboration that the operator logging on to a device is as claimed.

peer entity authentication: the corroboration that a peer entity in an association is the one claimed.

physical access control system: an electronic system that controls the ability of people or vehicles to enter a protected area, by means of authentication and authorization at access control points.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

physical segment: a single contiguous medium to which BACnet nodes are attached.

physically secure: a device or network that is protected from physical access by unauthorized individuals.

printable character: a character that represents a printable symbol as opposed to a device control character. Printable characters include, but are not limited to, upper and lowercase letters, punctuation marks, and mathematical symbols. The exact set depends upon the character set being used.

property: a particular characteristic of an object type.

proprietary: within the context of BACnet, any extension of or addition to object types, properties, PrivateTransfer services, or enumerations specified in this standard.

ramping: the gradual increase or decrease of the actual output from one setting to another at a fixed rate of change.

receiving BACnet-user: the BACnet-user that receives an indication or confirm service primitive.

reliability-evaluation: the process by which an object determines its reliability and thus the value to set into its Reliability property.

remote: pertaining to devices or nodes on a different network than the referenced device.

remote broadcast: a message addressed to all devices or nodes on a different BACnet network than the originator.

repeater: a device that connects two or more physical segments at the physical layer.

requesting BACnet-user: the BACnet-user that assumes the role of a client in a confirmed service.

responding BACnet-user: the BACnet-user that assumes the role of a server in a confirmed service.

role-based access control: access privileges that are assigned to specific roles. Access users acquire privileges through their assigned role.

router: a device that connects two or more BACnet networks at the network layer.

secure network: a network on which all traffic is required to be signed or encrypted.

security: any of a variety of procedures used to ensure that information exchange is guarded to prevent disclosure to unauthorized individuals. Security measures are intended to prevent disclosure of sensitive information even to those who have valid access to the communication network. Security is distinct from access control, although some security can be provided by limiting physical access to the communication medium itself.

segment: a segment consists of one or more physical segments interconnected by repeaters.

sending BACnet-user: the BACnet-user that issues a request or response service primitive.

server: a system or device that responds to a service request instance for some particular purpose. The server provides service to a client.

special date value: a date value that is one of the special values such as "even months", "last day of month", etc. These special date values are used in subcomponents (octets) of a value of type Date.

specific date: a fully specified date. For example, January 24, 1991, Day of week = Thursday. A specific date shall contain no unspecified octets or Special Date Values.

specific datetime: a BACnetDateTime construct composed of a specific date and a specific time.

specific time: a fully specified time. For example, 17:35:45.17 (= 5:35:45.17 P.M.). A specific time shall contain no unspecified octets.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

 © ISO 2022 – All rights reserved

standard object type: an object type defined by this standard where the numerical value is within the range reserved for ASHRAE.

standard property: a required or optional property of a standard object type where the numerical value of the property identifier is within the range reserved for ASHRAE and the property is listed in the object type's properties table in Clause 12.

stepping: the increase or decrease of an output value in discrete steps.

synchronization: a facility that allows processes to define and identify specific places in a transmission or exchange that can be used to reset a communication session to a predefined state.

time pattern: a time that may contain one or more unspecified octets.

timestamp: the indication of the point in time recorded for and accompanying the record of an event or operation.

trusted: a term used to refer to devices or networks from which messages are believed to be authentic, either through the use of secure messages or based on the physical security of that device or network.

unit_time: the length of time required to transmit one octet with a start bit and a single stop bit. Ten bit-times.

universal floor number: the number of a floor, beginning with 1 for the absolute lowest floor of the building and incrementing by one for each subsequent floor up.

unspecified date: a date composed entirely of unspecified octets (A value of X'FF' = D'255').

unspecified datetime: a BACnetDateTime construct composed of an unspecified date and an unspecified time.

unspecified octet: an octet used in the context of date, time or BACnetWeekNDay values that contains the value X'FF' = D'255'.

unspecified time: a time composed entirely of unspecified octets (A value of X'FF' = D'255').

upload: the process of transferring an executable program image or a database from a remote device in such a manner as to allow subsequent download.

virtual BACnet device: a BACnet device that is modeled in software, usually in a gateway, and does not exist as a physical BACnet device.

virtual BACnet network: a BACnet network of virtual BACnet devices, usually modeled by a gateway where no physical BACnet network exists.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

3.3 Abbreviations and Acronyms Used in this Standard

A application layer (prefix)
ABA American Bankers Association
AE application entity
AHU air handling unit
ANSI American National Standards Institute
APCI application protocol control information
APDU application layer protocol data unit
API application program interface
ARCNET attached resource computer network
ASE application service element
ASN.1 Abstract Syntax Notation One (ISO 8824)
B' ' denotes that binary notation is used between the single quotes
BAC building automation and control
BAS building automation system
BBMD BACnet/IP broadcast management device

© ISO 2022 – All rights reserved

BDT	broadcast distribution table
B/IP	BACnet/IP
B/IP-M	BACnet/IP multicast
B/IPv6	BACnet/IPv6
BVLC	BACnet virtual link control
BVLCI	BACnet virtual link control information
BVLL	BACnet virtual link layer
C	conditional
C(=)	conditional (The parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.)
CA	certificate authoritiy
CNF	confirm primitive
COV	change of value
CRC	cyclic redundancy check
CSML	control system modeling language
CSV	comma-separated values, as defined by RFC 4180
D' '	denotes that decimal notation is used between the single quotes
DA	local destination MAC layer address
DADR	ultimate destination MAC layer address
DER	data expecting reply (parameter in primitives and messages)
DER	distinguished encoding rules (see ISO/IEC 8825-1)
DESFire	Data Encryption Standard Fast, Innovative, Reliable and Secure
DHCP	Dynamic Host Configuration Protocol
DID	ARCNET destination MAC address
DLEN	1-octet length of ultimate destination MAC layer address
DNET	2-octet ultimate destination network number
DNS	Domain Name Service
DSAP	LLC destination service access point (X'82' for BACnet)
EIB	European Installation Bus
EIBA	European Installation Bus Association
EUI	Extended Unique Identifier (see IEEE 802)
EXEC	capable of executing a service request
FDT	foreign device table
HTTP	hypertext transfer protocol - RFC 2616
IANA	The Internet Assigned Numbers Authority
ICI	interface control information
IEEE	Institute of Electrical and Electronics Engineers
IL	ARCNET information length field
IND	indication primitive
INF	"Infinity", a unique binary pattern representing positive infinity (see ANSI/IEEE 754-1985)
-INF	"Negative infinity", a unique binary pattern representing negative infinity (see ANSI/IEEE 754-1985)
INIT	capable of initiating a service request
IP	Internet Protocol - RFC 791
IPv4	Internet Protocol version 4 - RFC 791
IPv6	Internet Protocol version 6 - RFC 2460
IRI	internationalized resource identifier - RFC 3987
ISO	International Organization for Standardization
JSON	JavaScript object notation - RFC 4627, 7159
JWT	JSON web token - RFC 7519
KNX	The Konnex System Specification: EIB is the core protocol of the Konnex standard. The Konnex System Specification reflects the current status for EIB.
L	data link (prefix)
LAN	local area network
LLC	logical link control (ISO 8802-2)
LPCI	link protocol control information
LPDU	link protocol data unit
LRC	Longitudinal Redundancy Check
LSAP	link service access point
LSDU	link service data unit

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

M	mandatory
M(=)	mandatory (The parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.)
MA	medium access (prefix)
MAC	medium access control
MPCI	MAC protocol control information
MPDU	MAC layer protocol data unit
MSDU	MAC service data unit
MS/TP	master-slave/token-passing
N	network layer (prefix)
NaN	"Not a Number", a unique binary pattern representing an invalid number (see ANSI/IEEE 754-1985)
NAT	Network Address Translation or Port Address Translation - RFC 2663
NP	network priority
NPCI	network protocol control information
NPDU	network layer protocol data unit
NRZ	non-return to zero
NSAP	network service access point
NSDU	network service data unit
O	indicates that support of a property is optional
OEO	occupant evacuation operation
OSI	open systems interconnection
P	physical layer (prefix)
PAC	ARCNET data packet header octet
PACS	physical access control system
PCI	protocol control information
PDU	protocol data unit
PICS	protocol implementation conformance statement
PKI	public key infrastructure
PPCI	physical layer protocol control information
PPDU	physical protocol data unit
PPP	Point-To-Point protocol - RFC 1661
PSDU	physical service data unit
PTP	point-to-point
R	indicates that a property shall be supported and readable using BACnet services
RBAC	role-based access control
REQ	request primitive
REST	representational state transfer
RFC	request for comment
RSA	Rivest-Shamir-Adleman cryptosystem
RSP	response primitive
S	selection
S(=)	selection (The parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.)
SA	local network source MAC layer address
SAP	service access point
SC	ARCNET system code (X'CD' for BACnet)
SDU	service data unit
SIA	Security Industry Association
SID	ARCNET source MAC address
SLAAC	IPv6 Stateless Auto Address Configuration - RFC 4862
SLEN	1-octet length of original source MAC layer address
SLIP	Serial Line Internet Protocol - RFC 1055
SNET	2-octet original source network number
SOAP	simple object access protocol
SPC	standard project committee
SSAP	LLC source service access point (X'82' for BACnet)
TLS	transport layer security - RFC 2246
TSM	transaction state machine

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

U	user option
U(=)	user option (The parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.)
UART	universal asynchronous receiver/transmitter
UDP	User Datagram Protocol - RFC 768
URI	uniform resource identifier - RFC 3986
URL	uniform resource locator - RFC 1738
UTC	Universal Time Coordinated
UUID	universally unique identifier - RFC 4122
VAV	variable air volume
VT	virtual terminal
W	indicates that a property shall be supported, readable, and writable using BACnet services
X' '	denotes that hexadecimal notation is used between the single quotes
XID	eXchange IDentification (ISO 8802-2)
XML	extensible markup language

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

4 BACnet PROTOCOL ARCHITECTURE

The Open System Interconnection (OSI) - Basic Reference Model (ISO 7498) is an international standard that defines a model for developing multi-vendor computer communication protocol standards. The OSI model addresses the general problem of computer-to-computer communication and breaks this very complex problem into seven smaller, more manageable sub-problems, each of which concerns itself with a specific communication function. Each of these sub-problems forms a "layer" in the protocol architecture.

The seven layers are arranged in a hierarchical fashion as shown in Figure 4-1. A given layer provides services to the layers above and relies on services provided to it by the layers below. Each layer can be thought of as a black box with carefully defined interfaces on the top and bottom. An application process connects to the OSI application layer and communicates with a second, remote application process. This communication appears to take place between the two processes as if they were connected directly through their application layer interfaces. Minimal knowledge or understanding of the other layers is required. In a similar manner, each layer of the protocol relies on lower layers to provide communication services and establishes a virtual peer-to-peer communication with its companion layer on the other system. The only real connection takes place at the physical layer.

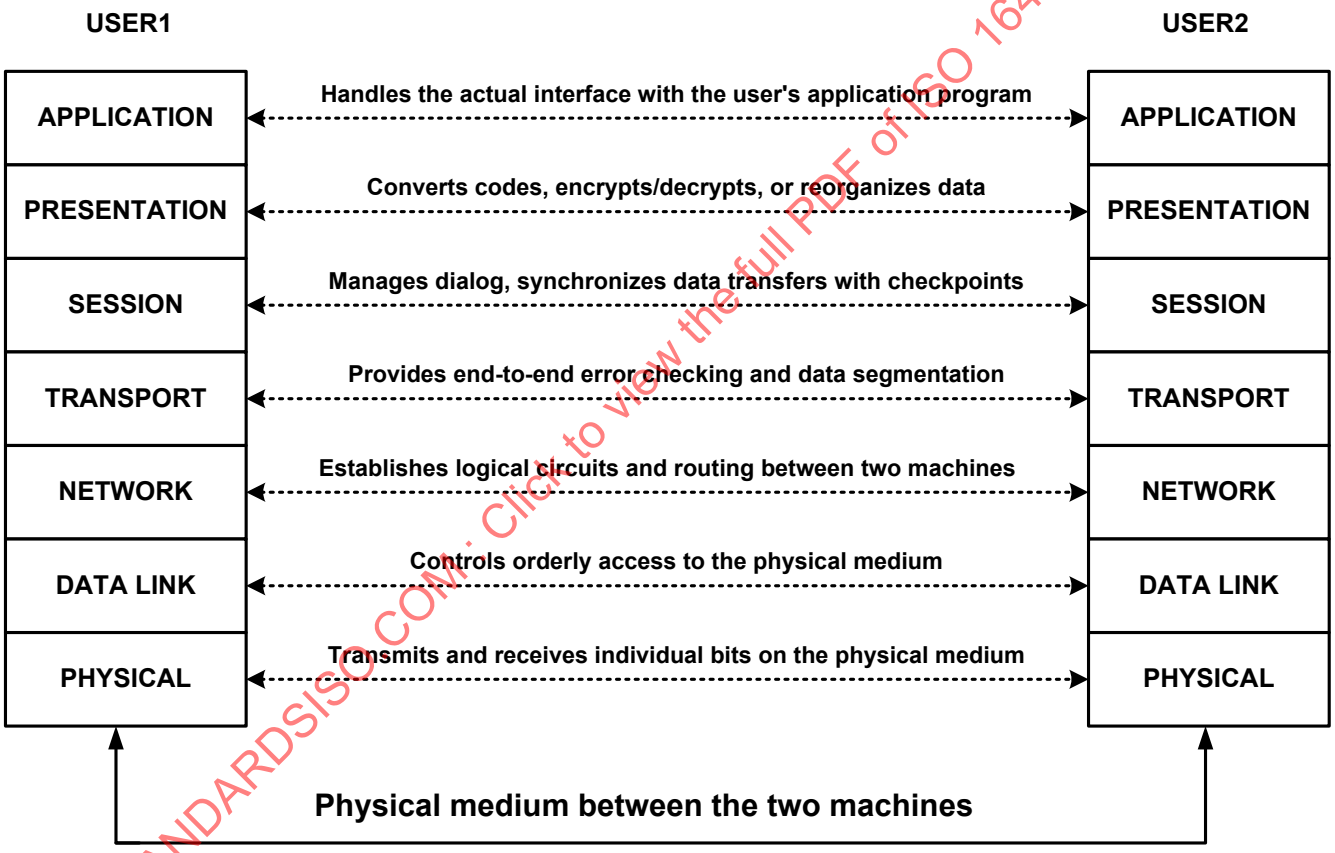


Figure 4-1. The ISO Open Systems Interconnection Basic Reference Model.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The OSI model addresses computer-to-computer communication from a very general perspective. It was designed to deal with the problems associated with computers in large, complex networks communicating with other computers in networks anywhere in the world. In this environment, computers can be separated by long distances and the messages might pass through several intermediate points, each of which may have to make routing decisions or perform some type of translation. Complex synchronization and error recovery schemes may also be needed.

The cost of implementing such a protocol today is prohibitively high for most building automation applications and is not generally required. Nevertheless, the OSI model is a good one to use for a building automation protocol if consideration is given to including only the OSI functionality that is actually needed, thereby collapsing the seven-layer architecture. In a collapsed architecture, only selected layers of the OSI model are included. The other layers are effectively null, thus reducing message length and communication processing overhead. Such a collapsed architecture permits the building automation industry to take advantage of lower cost, mass-produced processor and local area

© ISO 2022 – All rights reserved

network technologies such as have been developed for the process control and office automation industries. The use of readily available, widespread technologies, such as Ethernet[1], ARCNET[2], and LonTalk[3], will lower the cost, increase performance, and open new doors to system integration.

[1] Ethernet is a registered trademark of Digital Equipment Corporation, Intel, and Xerox and is the basis for international standard ISO 8802-3.
[2] ARCNET is a registered trademark of Datapoint Corporation.
[3] LonTalk is a registered trademark of Echelon Corporation.

4.1 The BACnet Collapsed Architecture

BACnet is based on a four-layer collapsed architecture that corresponds to the physical, data link, network, and application layers of the OSI model as shown in Figure 4-2. The application layer and a simple network layer are defined in the BACnet standard. BACnet provides the follwing options that correspond to the OSI data link and physical layers.

Ethernet (ISO 8802-3)	Clause 7
ARCNET (ATA 878.1)	Clause 8
MS/TP	Clause 9
PTP	Clause 10
LonTalk (ISO/IEC 14908.1)	Clause 11
BACnet/IP	Annex J
BACnet/IPv6	Annex U
ZigBee	Annex O
BACnet/SC	Annex AB

Collectively these options provide a master/slave MAC, deterministic token-passing MAC, high-speed contention MAC, dial-up access, Internet access, star and bus topologies, and a choice of twisted-pair, coax, or fiber optic media, in addition to wireless connectivity.

A four-layer collapsed architecture was chosen after careful consideration of the particular features and requirements of BAC networks, including a constraint that protocol overhead needed to be as small as possible. The reasoning behind the selection of the physical, data link, network, and application layers for inclusion in the BACnet architecture is outlined in this clause.

What layers are required for the proper operation of a BAC network? BAC networks function as local area networks, either physically, as with MS/TP, or logically, as with BACnet/IP. This is true even though in some applications it is necessary to exchange information with devices in a building that is very far away. This long-distance communication is done through the telephone networks or across the Internet. The routing, relaying, and guaranteed delivery issues are handled by the telephone and Internet systems and can be considered external to the BAC network. BAC devices are static. They don't move from place to place and the functions that they are asked to perform do not change in the sense that a manufacturing device may make one kind of part today and some very different part tomorrow. These are among the features of BAC networks that can be used to evaluate the appropriateness of the layers in the OSI model.

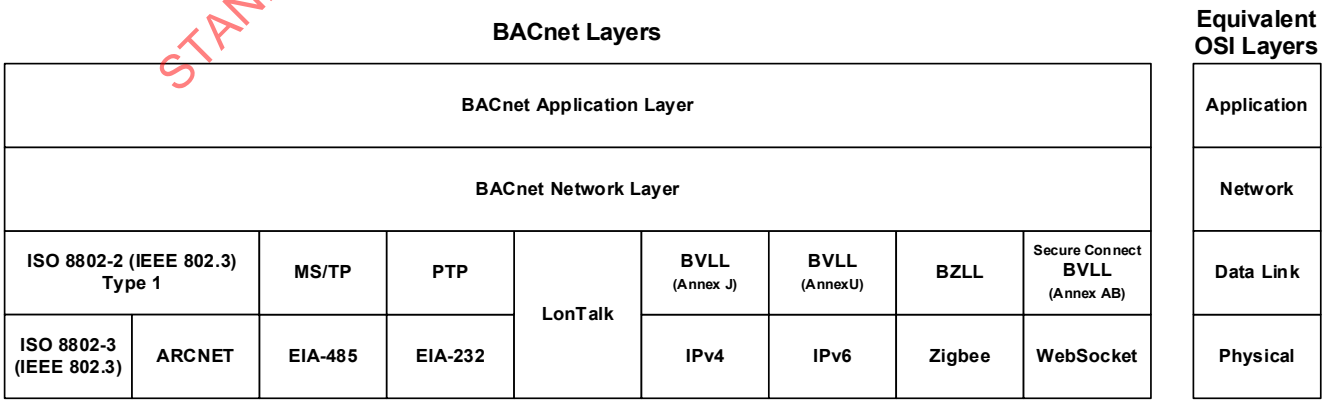


Figure 4-2. BACnet collapsed architecture.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The physical layer provides a means of connecting the devices and transmitting the electronic signals that convey the data. Clearly the physical layer is needed in a BAC protocol.

The data link layer organizes the data into frames or packets, regulates access to the medium, provides addressing, and handles some error recovery and flow control. These are all functions that are required in a BAC protocol. The conclusion is that the data link layer is needed. For the purposes of BACnet, a data link layer entity may ignore any service request primitive that is not defined for the particular data link type.

Functions provided by the network layer include translation of global addresses to local addresses, routing messages through one or more networks, accommodating differences in network types and in the maximum message size permitted by those networks, sequencing, flow control, error control, and multiplexing. BACnet is designed so that there is only one logical path between devices, thus eliminating the need for optimal path routing algorithms. A network is made up of one or more physical segments connected by repeaters or bridges but with a single local address space. In the case of a single network, most network layer functions are either unnecessary or duplicate data link layer functions. For some BACnet systems, however, the network layer is a necessity. This is the case when two or more networks in a BACnet internet use different MAC layer options. When this occurs, there is a need to recognize the difference between local and global addresses and to route messages to the appropriate networks. BACnet provides this limited network layer capability by defining a network layer header that contains the necessary addressing and control information.

The transport layer is responsible for guaranteeing end-to-end delivery of messages, segmentation, sequence control, flow control, and error recovery. Most of the functions of the transport layer are similar to functions in the data link layer, though different in scope. The scope of transport layer services is end-to-end whereas the scope of data link services is point-to-point across a single network. Since BACnet supports configurations with multiple networks, the protocol must provide the end-to-end services of the transport layer. Guaranteed end-to-end delivery and error recovery are provided in the BACnet application layer via message retry and timeout capabilities. Message segmentation and end-to-end flow control is required for buffer and processor resource management. This is because potentially large amounts of information may be returned for even simple BACnet requests. These functions are provided in the BACnet application layer. Last, sequence control is required in order to properly reassemble segmented messages. This is provided in the BACnet application layer within the segmentation procedure. Since BACnet is based on a connectionless communication model, the scope of the required services is limited enough to justify implementing these at a higher layer, thus saving the communication overhead of a separate transport layer.

The session layer is used to establish and manage long dialogues between communicating partners. Session layer functions include establishing synchronization checkpoints and resetting to previous checkpoints in the event of error conditions to avoid restarting an exchange from the beginning. Most communications in a BAC network are very brief. For example, reading or writing one or a few values, notifying a device about an alarm or event, or changing a setpoint. Occasionally longer exchanges take place, such as uploading or downloading a device. The few times when the services of this layer would be helpful do not justify the additional overhead that would be imposed on the vast majority of transactions, which are very brief and do not need them.

The presentation layer provides a way for communicating partners to negotiate the transfer syntax that will be used to conduct the communication. This transfer syntax is a translation from the abstract user view of data at the application layer to sequences of octets treated as data at the lower layers. If only one transfer syntax is permitted, then the presentation layer function reduces to an encoding scheme for representing the application data. BACnet defines such a fixed encoding scheme and includes it in the application layer, making an explicit presentation layer unnecessary.

The application layer of the protocol provides the communication services required by the applications to perform their functions, in this case monitoring and control of HVAC&R and other building systems. Clearly an application layer is needed in the protocol.

In summary:

(a) The resource and overhead costs for implementing a full OSI seven-layer architecture make it impractical for current building automation devices.
(b) Following the OSI model offers advantages in terms of adopting existing computer networking technology. This can result in cost savings and make integration with other computer network systems easier.
(c) The expectations and environment of building automation systems permit simplification of the OSI model by eliminating the functionality of some of the layers.
(d) A collapsed architecture made up of the physical, data link, network, and application layers is the optimum solution for today's building automation systems.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

4.2 BACnet Network Topology

In the interest of application flexibility, BACnet does not prescribe a rigid network topology. Rather, BACnet devices are physically connected to one of the types of local area networks (LANs) or via dedicated or dial-up serial, asynchronous lines. These networks may then be further interconnected by BACnet routers as described in Clause 6.

In terms of LAN topology, each BACnet device is attached to an electrical medium or physical segment. A BACnet segment consists of one or more physical segments connected at the physical layer by repeaters. A BACnet network consists of one or more segments interconnected by bridges, devices that connect the segments at the physical and data link layers and may perform message filtering based upon MAC addresses. A BACnet network forms a single MAC address domain. Multiple BACnet networks, possibly employing different LAN technologies, may be interconnected by BACnet routers to form a BACnet internetwork. In a BACnet internetwork, there exists exactly one message path between any two nodes. These concepts are shown graphically in Figure 4-3.

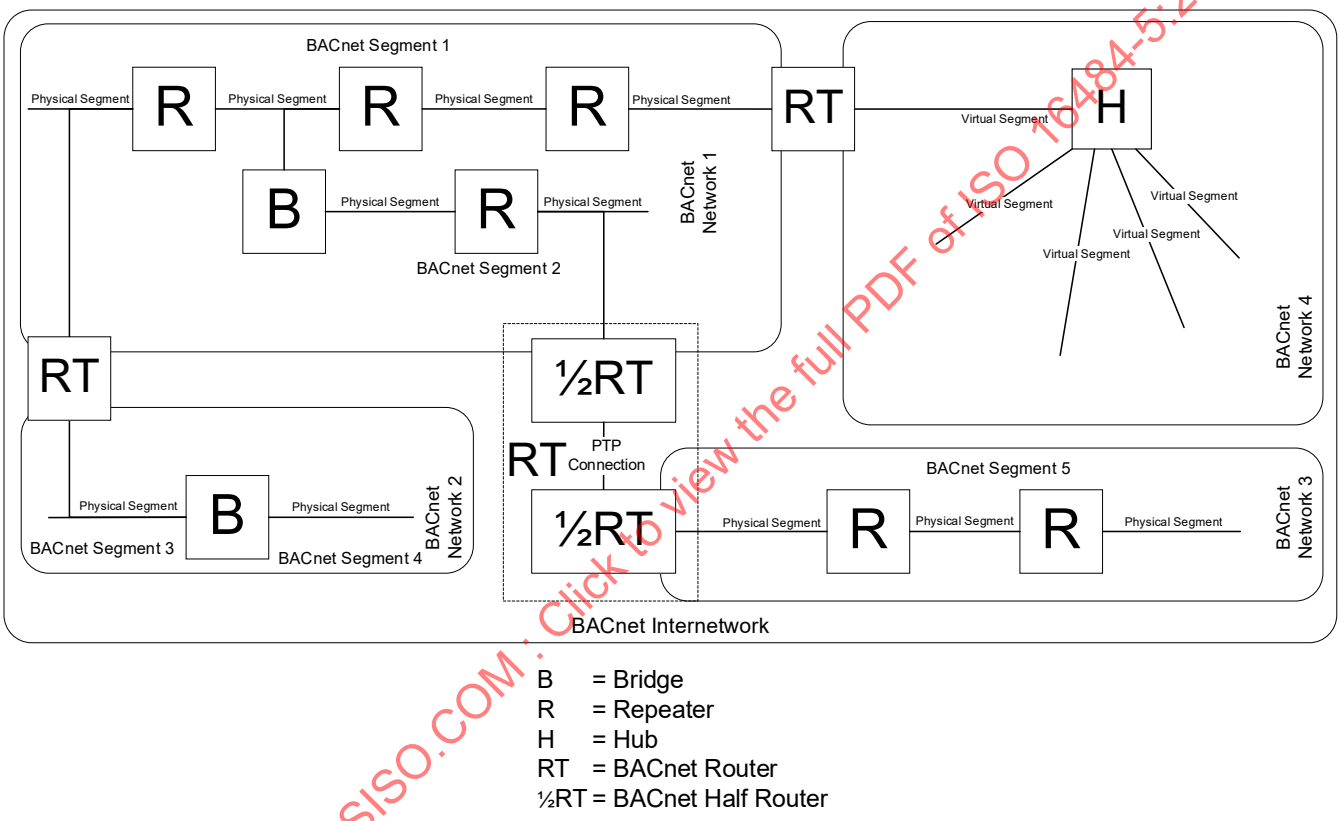


Figure 4-3. A BACnet internetwork illustrating the concepts of Physical Segments, Virtual Segments, Repeaters, Bridges, Hubs, BACnet Segments, BACnet Networks, BACnet Half Routers, and BACnet Routers.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

4.3 Security

The principal security threats to BACnet systems are people who, intentionally or by accident, modify a device's configuration or control parameters. Problems due to a malfunctioning or misconfigured computer are outside the realm of security considerations. One important place for security measures is the operator-machine interface. Since the operator-machine interface is not part of the communication protocol, vendors are free to include password protection, audit trails, or other controls to this interface as needed. In addition, write access to any properties that are not explicitly required to be "writable" by this standard may be restricted to modifications made only in virtual terminal mode or be prohibited entirely. This permits vendors to protect key properties with a security mechanism that is as sophisticated as they consider appropriate.

It is recommended that BACnet devices support updating of the device's firmware and software. The procedures for firmware and software upgrades are a local matter.

© ISO 2022 – All rights reserved

For the BACnet/SC data link layer option, standard network security mechanisms based on Transport Layer Security (TLS, successor of SSL) are used to provide peer authentication, message integrity, and encryption for communication within a BACnet/SC network. See Annex AB.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

5 THE APPLICATION LAYER

5.1 The Application Layer Model

This clause presents a model of the BACnet application layer. The purpose of the model is to describe and illustrate the interaction between the application layer and application programs, the relationship between the application layer and lower layers in the protocol stack, and the peer-to-peer interactions with a remote application layer. This model is not an implementation specification.

An Application Process is that functionality within a system that performs the information processing required for a particular application. All parts of the Application Process outside the Application Layer, (i.e. those that do not concern the communication function) are outside the scope of BACnet. The part of the Application Process that is within the Application Layer is called the Application Entity. In other words, an Application Entity is that part of the Application Process related to the BACnet communication function. An application program interacts with the Application Entity through the Application Program Interface (API). This interface is not defined in BACnet, but it would probably be a function, procedure, or subroutine call in an actual implementation. These concepts are illustrated in Figure 5-1. The shaded region indicates the portion of the Application Process that is within the BACnet Application Layer.

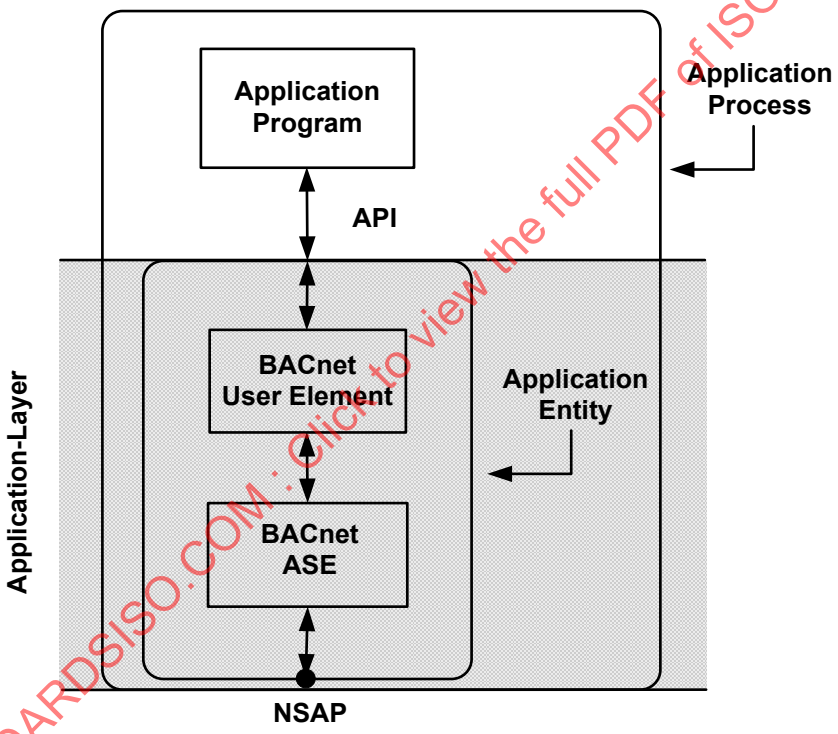


Figure 5-1. Model of a BACnet Application Process.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The Application Entity is itself made up of two parts: the BACnet User Element and the BACnet Application Service Element (ASE). The BACnet ASE represents the set of functions or application services specified in Clauses 13 through 17. The BACnet User Element carries out several functions in addition to supporting the local API. It represents the implementation of the "service procedure" portion of each application service. It is responsible for maintaining information about the context of a transaction, including generating invoke IDs and remembering which invoke ID goes with which application service request (response) to (from) which device. It is also responsible for maintaining the time-out counters that are required for the retrying of a transmission. The BACnet User Element also presides over the mapping of a device's activities into BACnet objects.

Information exchanged between two peer application processes is represented in BACnet as an exchange of abstract service primitives, following the ISO conventions contained in the OSI technical report on service conventions, ISO TR 8509. These primitives are used to convey service-specific parameters that are defined in Clauses 13 through 17. Four service primitives are defined: request, indication, response, and confirm. The information contained in the primitives

© ISO 2022 – All rights reserved

is conveyed using a variety of protocol data units (PDUs) defined in this standard. In order to make clear which BACnet PDU is being used, the notation will be as follows:

CONF_SERV.request	CONF_SERV.indication	CONF_SERV.response	CONF_SERV.confirm
UNCONF_SERV.request	UNCONF_SERV.indication		
SEGMENT_ACK.request	SEGMENT_ACK.indication		
REJECT.request	REJECT.indication		
ABORT.request	ABORT.indication		
	SEC_ERR.indication		

The designation CONF_SERV indicates that BACnet confirmed service PDUs are being used. Similarly, the designations UNCONF_SERV, SEGMENT_ACK, ERROR, REJECT, and ABORT indicate that unconfirmed service PDUs, segment acknowledge PDUs, error PDUs, reject PDUs, and abort PDUs, respectively, are being used. The designation SEC_ERR indicates that an error occurred in the BACnet security layer and is being indicated up to the BACnet application program. The format of a SEC_ERR.indication is a local matter.

An application program that needs to communicate with a remote application process accesses the local BACnet User Element through the API. Some of the API parameters, such as the identity (address) of the device to which the service request is to be sent and protocol control information, is passed directly down to the network or data link layers. The remainder of the parameters make up an application service primitive that is passed from the BACnet User Element to the BACnet ASE. Conceptually, the application service primitive results in the generation of an APDU that becomes the data portion of a network service primitive, which is passed to the network layer through the Network Service Access Point (NSAP). Similarly this request passes down through the lower layers of the protocol stack in the local device. This process is illustrated in Figure 5-2. The message is then transmitted to the remote device, where it is passed up through the protocol stack in the remote device, eventually appearing as an indication primitive passed from the remote BACnet ASE to the remote BACnet User Element. The response from the remote device, if any, returns to the initiator of the service in a similar fashion (see Clause 5.5).

In addition to the service primitives and the service specific parameters, the application entity exchanges interface control information (ICI) parameters with the application program via the API. The content of the ICI is dependent upon the service primitive type. The ICI parameters received by the application entity provide the information that is passed on to the lower layers (as ICI across layer interfaces) to help them construct their PDUs. The ICI parameters that are provided by the application entity to the application programs contain information recovered by the lower layers from their respective PDUs.

The following ICI parameters are exchanged with the various service primitives across an API:

'destination_address' (DA): the address of the device(s) intended to receive the service primitive. Its format (device name, network address, etc.) is a local matter. This address may also be a multicast, local broadcast or global broadcast type.

'source_address' (SA): the address of the device from which the service primitive was received. Its format (device name, network address, etc.) is a local matter.

'network_priority' (NP): a four-level network priority parameter described in Clause 6.2.2.

'data_expecting_reply' (DER): a Boolean parameter that indicates whether (TRUE) or not (FALSE) a reply service primitive is expected for the service being issued.

'data_attributes' (DAT): The optional parameter that provides extra information about the data for the request to send, or from the received request. This optionally includes security related information in a portion called 'security_parameters'. The format of this parameter is a local matter.

'security_parameters' (SEC): An optional portion of the 'data_attributes' parameter that provides security related information for the request to send, or from the received request. Its format is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

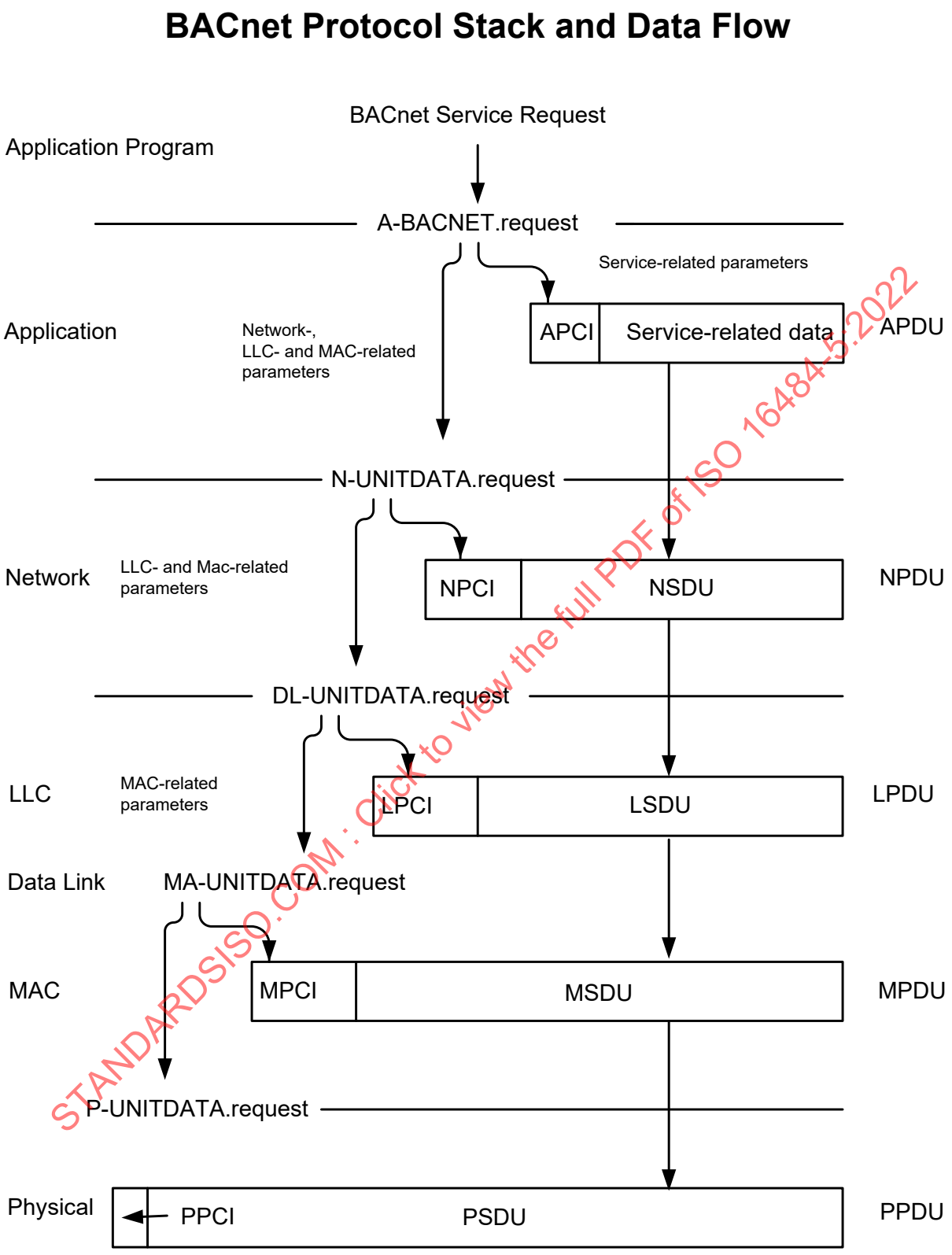


Figure 5-2. BACnet protocol stack and data flow.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 5-1 describes the applicability of the ICI parameters to the service primitives.

Table 5-1. Applicability of ICI Parameters for Abstract Service Primitives

Service Primitive	DA	SA	NP	DER	SEC
CONF_SERV.request	Yes	No	Yes	Yes	Yes
CONF_SERV.indication	Yes	Yes	Yes	Yes	Yes
CONF_SERV.response	Yes	No	Yes	Yes	Yes
CONF_SERV.confirm	Yes	Yes	Yes	No	Yes
UNCONF_SERV.request	Yes	No	Yes	No	Yes
UNCONF_SERV.indication	Yes	Yes	Yes	No	Yes
REJECT.request	Yes	No	Yes	No	Yes
REJECT.indication	Yes	Yes	Yes	No	Yes
SEGMENT_ACK.request	Yes	No	Yes	No	Yes
SEGMENT_ACK.indication	Yes	Yes	Yes	No	Yes
ABORT.request	Yes	No	Yes	No	Yes
ABORT.indication	Yes	Yes	Yes	No	Yes
SEC_ERR.indication	Yes	Yes	No	No	Yes

A "BACnet device" is any device, real or virtual, that supports digital communication using the BACnet protocol. Each BACnet device contains exactly one Device object, as defined in Clause 12.11. A BACnet device is uniquely located by an NSAP, which consists of a network number and a MAC address.

In most cases, a physical device will implement a single BACnet device. It is possible, however, that a single physical device may implement a number of "virtual" BACnet devices. This is described in Annex H.

Note that some application layer requirements are relaxed for BACnet gateways to non-BACnet protocols. See Annex H.

5.1.1 Confirmed Application Services

BACnet defines confirmed application services based on a client and server communication model. A client requests service from a server via a particular service request instance. The server provides service to a client and responds to the request. This relationship is illustrated in Figure 5-3. The BACnet-user that assumes the role of a client is called the "requesting BACnet-user" and the BACnet-user that assumes the role of the server is called the "responding BACnet-user."

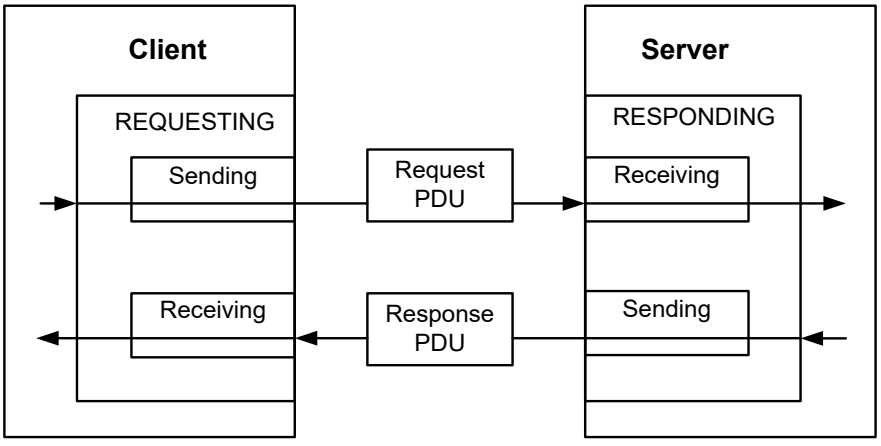


Figure 5-3. Relationship of a client and server.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

A requesting BACnet-user issues a CONF_SERV.request primitive, which causes a request PDU to be sent. When a response PDU arrives, the requesting BACnet-user receives a CONF_SERV.confirm primitive. When a request PDU arrives, the responding BACnet-user receives a CONF_SERV.indication primitive. The responding BACnet-user issues a CONF_SERV.response primitive, which causes a response PDU to be sent. Thus, the requesting BACnet-user and the responding BACnet-user play a role in both sending and receiving PDUs. The term "sending BACnet-user" applies to a BACnet user that initiates the sending of a PDU. The term "receiving BACnet-user" applies to a BACnet-user that receives an indication that a PDU has arrived.

5.1.2 Unconfirmed Application Services

The client and server model, and the terms "requesting BACnet-user" and "responding BACnet-user," do not apply to unconfirmed services. The terms "sending BACnet-user" and "receiving BACnet-user" do apply, however, and they are used to define the service procedure for unconfirmed services.

In some cases, errors will be encountered when sending unconfirmed requests. While the standard does not provide a mechanism for reporting such errors to the application program, it is acceptable for an implementation to provide such a mechanism, if desired.

5.2 Segmentation of BACnet Messages

To provide for messages that are longer than the maximum length supported by a communications network, or by the sending or receiving device, BACnet provides a method to perform application layer segmentation. In BACnet, only Confirmed-Request and ComplexACK messages may be segmented. Segmentation is an optional feature of BACnet.

5.2.1 Message Segmentation Rules

This clause prescribes rules for dividing a message into segments.

5.2.1.1 Rules for Segmenting APDU Data Streams

Each BACnet message is encoded into a sequence of tags and values according to the relevant ASN.1 definitions in Clause 21 and the encoding rules of Clause 20. The following rules apply to segmenting this data stream:

(a) If possible, an entire message shall be sent in a single APDU.
(b) If an entire message cannot be sent in a single APDU, the message shall be segmented into the minimum number of APDUs possible.
(c) Messages shall be segmented only at octet boundaries.

5.2.1.2 Maximum APDU Length

The maximum length of a BACnet APDU shall be the smallest of

(a) the maximum APDU size transmittable by a device, which may be restricted by local buffer limitations and is a local matter;
(b) the maximum APDU size conveyable by the internetwork to the remote device, which is constrained by the maximum NPDU length permitted by the data links used by the local, remote, and any intervening networks, as specified in Clause 6;
(c) the maximum APDU size accepted by the remote peer device, which must be at least 50 octets.

If the sending device is the requesting BACnet-user, i.e. the APDU to be sent is a BACnet-Confirmed-Request-PDU or a BACnet-Unconfirmed-Request-PDU, then the maximum APDU size accepted by the remote peer is specified by the Max_APDU_Length_Accepted property of the remote peer's Device object. The value of this property may be read using the read property services described in Clause 15 or the value may be obtained from the 'Max APDU Length Accepted' parameter of an I-Am service request received from the remote device. The remote peer may be solicited to transmit an I-Am service request by sending it a Who-Is service request, as described in Clause 16.9

If the sending device is not the requesting BACnet-user, i.e. the APDU to be sent is a BACnet-ComplexACK-PDU, then the maximum APDU size accepted by the remote peer is specified in the 'Max APDU Length Accepted' parameter of the BACnet-Confirmed-Request-PDU for which this is a response.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The value determined by the above constraints will be designated the maximum-transmittable-length. Note that maximum-transmittable-length will in general not be a constant unless minimum values are used for each constraint.

See Clause 19.4 for an approach to determine the maximum APDU conveyable by the internetwork.

5.2.1.3 Maximum Segments Accepted

The maximum number of segments transmitted in a Confirmed-Request or ComplexACK message shall be the smallest of:

(a) the maximum number of segments transmittable by a device, which may be restricted by local limitations and is a local matter;
(b) the maximum number of segments accepted by the remote peer device.

If the sending device is the requesting BACnet-user, i.e. the message to be sent is a Confirmed-Request, then the maximum number of segments accepted by the remote peer device is specified in the Max_Segments_Accepted property of the remote peer's Device object.

If the sending device is not the requesting BACnet-user, i.e. the message to be sent is a ComplexACK, then the maximum number of segments accepted by the remote peer device is specified in the 'Max Segments Accepted' parameter of the BACnet-Confirmed-Request-PDU for which this is a response.

5.2.2 Segmentation Protocol Control Information (PCI)

To provide for the possibility of segmented messages, the headers of the BACnet-Confirmed-Request-PDU and BACnet-ComplexACK-PDU contain two Boolean parameters called 'Segmented Message' and 'More Follows.'

If the length of a fully encoded message of the type conveyed by one of the above APDUs results in an APDU whose length is less than or equal to the maximum-transmittable-length as determined according to Clause 5.2.1, the 'Segmented Message' and 'More Follows' parameters shall both be set to FALSE.

If, however, the encoded length of a message would result in an APDU length greater than the maximum-transmittable-length as determined according to Clause 5.2.1, the 'Segmented Message' parameter shall be set to TRUE for all segments, and the 'More Follows' parameter shall be set to TRUE for all segments except the last.

Two additional parameters are also present, conditionally, in the header of each APDU carrying a segment of a Confirmed-Request message or a ComplexACK message. The first conditional parameter is the 'Sequence Number.' This one-octet unsigned integer is used by the segment transmitter to indicate the position of the current segment in the series of segments composing the complete message. The second conditional parameter is the 'Proposed Window Size.' This one-octet unsigned integer is used by the segment transmitter to indicate the maximum number of message segments that it is prepared to transmit before it must receive a SegmentACK. The use of these parameters in the transmission of segmented messages is described in Clauses 5.3 and 5.4.

The 'Sequence Number' of the initial segment shall be zero. The segment receiver may request the transmission of the next segment or group of segments by sending a SegmentACK-PDU containing the 'Sequence Number' parameter of the last successfully received segment. Such a request shall also serve as an acknowledgment of this segment. If the Window Size is greater than one, such a SegmentACK-PDU shall also serve to acknowledge any previously transmitted but unacknowledged segments.

If either party in a segmented transaction wishes to terminate the transaction, that party may issue an Abort-PDU.

5.3 Transmission of BACnet APDUs

The formal description of the transmission and reception protocol for BACnet APDUs is contained in the Transaction State Machine description given in Clause 5.4. This clause is intended only as an overview of the protocol.

5.3.1 Confirmed-Request Message Transmission

Upon transmitting a complete unsegmented Confirmed-Request message or upon receiving the SegmentACK acknowledging the final segment of a segmented Confirmed-Request message, a client device shall start a timer that indicates the length of time the message has been outstanding. The timer shall be canceled upon the receipt of an Error,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Reject, Abort, SimpleACK, or ComplexACK APDU for the outstanding Confirmed-Request message, and the client application shall be notified. If the timer exceeds the value of the APDU_Timeout property in the client's Device object, then the complete Confirmed-Request message shall be retransmitted and the timer shall be reset to zero. All retransmitted Confirmed-Request messages shall follow this same procedure until the message has been retransmitted the number of times indicated in the Number_Of_APDU_Retries property of the client's Device object. If, after the Confirmed-Request message is retransmitted the appropriate number of times, a response is still not received, the message shall be discarded and the client application shall be notified.

5.3.2 Segmented Confirmed-Request Message Transmission

Before sending the first segment of a segmented Confirmed-Request-PDU, a client device shall choose a Proposed Window Size to indicate the maximum number of message segments it is prepared to transmit before it must receive a SegmentACK. The means of choosing the Proposed Window Size are a local matter, except that the value shall be in the range 1 to 127, inclusive. The Proposed Window Size shall be carried by the parameter of that name in each segment of the Confirmed-Request-PDU. The value of Proposed Window Size shall be the same in each segment of the Confirmed-Request-PDU.

Upon transmitting the first segment of a Confirmed-Request message, a client device shall start a timer that indicates the length of time this message segment has been outstanding. The timer shall be canceled upon the receipt of a Reject, Abort, or SegmentACK APDU for the outstanding Confirmed-Request message segment. If the timer exceeds the value of the APDU_Segment_Timeout property in the client's Device object, then the segment shall be retransmitted and the timer shall be reset to zero. All retransmitted segments shall follow this same procedure until the message segment has been retransmitted the number of times indicated in the Number_Of_APDU_Retries property of the client's Device object. If, after the message segments are retransmitted the appropriate number of times, a response is still not received, the message shall be discarded and the client application shall be notified.

Upon receipt of the first segment of a segmented Confirmed-Request-PDU, the server device shall choose an Actual Window Size to indicate the number of sequential message segments it expects to receive before it transmits a SegmentACK. The means of choosing the Actual Window Size are a local matter, except that the value shall be less than or equal to the 'proposed-window-size' parameter contained in the Confirmed-Request-PDU and shall be in the range 1 to 127, inclusive. The value of Actual Window Size shall be the same in each SegmentACK sent in response to a given Confirmed-Request. Regardless of the value of Actual Window Size, a SegmentACK shall be sent in response to the first segment of a Confirmed-Request.

Upon receipt of a SegmentACK APDU, the client device shall set its Actual Window Size equal to the value associated with the 'actual-window-size' parameter in the SegmentACK APDU. After this point, the client has authorization to send as many segments as the 'actual-window-size' parameter indicates before waiting for a SegmentACK APDU. No more than T_{seg} may be allowed to elapse between the receipt of a SegmentACK APDU and the transmission of a segment. No more than T_{seg} may be allowed to elapse between the transmission of successive segments of a group. After transmitting a set of segments that fills the window or completes the message, a client device shall start a timer that indicates the length of time these message segments have been outstanding. The timer shall be canceled upon receipt of a Reject, Abort, or SegmentACK APDU for some or all of the outstanding Confirmed-Request message segments. If the timer exceeds the value of the APDU_Segment_Timeout property in the client's Device object, then the segments shall be retransmitted and the timer shall be reset to zero. All retransmitted segments shall follow this same procedure until the message segments have been retransmitted the number of times indicated in its Device object's Number_Of_APDU_Retries property. If, after the Confirmed-Request message segments are retransmitted the appropriate number of times, a response is still not received, the message shall be discarded and the client application shall be notified.

It is possible to receive a Reject, Abort, or SegmentACK APDU during the sending of a sequence of Confirmed-Request segments even though the number of outstanding segments is less than indicated by the Actual Window Size. In this case, receipt of a Reject or Abort APDU shall terminate the Confirmed-Request transaction. Receipt of a SegmentACK APDU shall be considered as an acknowledgment for the segments up to and including the number indicated in the 'sequence-number' parameter of the SegmentACK APDU. Any unacknowledged segments shall be retransmitted following the above procedure.

It is recognized that in some cases where a Reject, Abort, or SegmentACK APDU is received, the client device may have sent, or irretrievably queued for sending, one or more (but less than Actual Window Size) additional Confirmed-Request-PDU segments.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

5.3.3 Segmented ComplexACK Message Transmission

Before sending the first segment of a segmented ComplexACK-PDU, a server device shall choose a Proposed Window Size to indicate the maximum number of message segments it is prepared to transmit before it must receive a SegmentACK. The means of choosing the Proposed Window Size are a local matter, except that the value shall be in the range 1 to 127, inclusive. The Proposed Window Size shall be carried by the parameter of that name in each segment of the Confirmed-Request-PDU. The value of Proposed Window Size shall be the same in each segment of the ComplexACK-PDU.

Upon transmitting the first segment of a ComplexACK message, a server device shall start a timer that indicates the length of time this message segment has been outstanding. The timer shall be canceled upon the receipt of an Abort or SegmentACK APDU for the outstanding ComplexACK message segment. If the timer exceeds the value of the APDU_Segment_Timeout property in the server's Device object, then the segment shall be retransmitted and the timer shall be reset to zero. All retransmitted segments shall follow this same procedure until the message segment has been retransmitted the number of times indicated in the Number_Of_APDU_Retries property of the server's Device object. If, after the message segments are retransmitted the appropriate number of times, a response is still not received, the message shall be discarded.

Upon receipt of the first segment of a segmented ComplexACK APDU, the client device shall choose an Actual Window Size to indicate the number of sequential message segments it expects to receive before it transmits a SegmentACK. The means of choosing the Actual Window Size are a local matter, except that the value shall be less than or equal to the 'proposed-window-size' parameter contained in the ComplexACK-PDU and shall be in the range 1 to 127, inclusive. The value of Actual Window Size shall be the same in each SegmentACK sent in response to a given ComplexACK. Regardless of the value of Actual Window Size, a SegmentACK shall be sent in response to the first segment of a ComplexACK.

Upon receipt of a SegmentACK APDU, the server device shall set its Actual Window Size equal to the value associated with the 'actual-window-size' parameter in the SegmentACK APDU. After this point, the server has authorization to send as many segments as the 'actual-window-size' parameter indicates before waiting for a SegmentACK APDU. No more than T_{seg} may be allowed to elapse between the receipt of a SegmentACK APDU and the transmission of a segment. No more than T_{seg} may be allowed to elapse between the transmission of successive segments of a group. After transmitting a set of segments that fills the window or completes the message, a server device shall start a timer that indicates the length of time these message segments have been outstanding. The timer shall be canceled upon receipt of an Abort or SegmentACK APDU for some or all of the outstanding ComplexACK message segments. If the timer exceeds the value of the APDU_Segment_Timeout property in the server's Device object, then the segments shall be retransmitted and the timer shall be reset to zero. All retransmitted segments shall follow this same procedure until the message segments have been retransmitted the number of times indicated in the Number_Of_APDU_Retries property of the server's Device object. If, after the ComplexACK message segments are retransmitted the appropriate number of times, a response is still not received, the message shall be discarded.

It is possible to receive an Abort or SegmentACK APDU during the sending of a sequence of ComplexACK segments even though the number of outstanding segments is less than indicated by the Actual Window Size. In this case, receipt of an Abort APDU shall terminate the ComplexACK transaction. Receipt of a SegmentACK APDU shall be considered as an acknowledgment for the segments up to and including the number indicated in the 'sequence-number' parameter of the SegmentACK APDU. Any unacknowledged segments shall be retransmitted following the above procedure.

It is recognized that in some cases where an Abort or SegmentACK APDU is received, the server device may have sent, or irretrievably queued for sending, one or more (but less than Actual Window Size) additional ComplexACK segments.

5.3.4 SegmentACK APDU Transmission

A device shall transmit a SegmentACK upon any of the following conditions:

(a) The device receives the initial segment of a segmented message. In this case, the 'negative-ack' parameter of the SegmentACK shall have a value of FALSE, indicating that this is a positive acknowledgment, and the 'sequence-number' parameter of the SegmentACK shall have a value of zero, indicating that the first segment has been acknowledged and that the segment transmitter may continue sending, commencing with the next sequential segment.

(b) The device receives a quantity of unacknowledged, sequentially numbered segments for this transaction equal to the Actual Window Size. In this case, the 'negative-ack' parameter of the SegmentACK shall have a value of FALSE, indicating that this is a positive acknowledgment, and the 'sequence-number' parameter of the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

SegmentACK shall have a value equal to the 'sequence-number' parameter of the last received segment, indicating that all segments up to and including 'sequence-number' have been acknowledged and that the segment transmitter may continue sending, commencing with the next sequential segment.

(c) The device receives a segment out of order (possibly indicating that a segment has been missed). In this case, the segment receiver shall discard the out-of-order segment. In this context, "out of order" means a segment whose 'sequence-number' is not equal to the next expected 'sequence-number.' The 'negative-ack' parameter of the SegmentACK shall have a value of TRUE, indicating that this is a negative acknowledgment. The 'sequence-number' parameter of the SegmentACK shall have a value equal to the 'sequence-number' parameter of the last received correctly ordered segment, indicating that all segments up to and including 'sequence-number' have been acknowledged and that the segment transmitter should resend, commencing with the next sequential segment after that indicated by the 'sequence-number' parameter contained in the SegmentACK.

(d) The device receives the final segment of a message. In this case, the 'negative-ack' parameter of the SegmentACK shall have a value of FALSE, indicating that this is a positive acknowledgment, and the 'sequence-number' parameter of the SegmentACK shall have a value equal to the 'sequence-number' parameter of the final message segment, indicating that all segments up to and including the final segment have been acknowledged.

5.3.5 Duplicate APDUs and Message Segments

5.3.5.1 Terminating Client TSMs

When using the BACnet error recovery procedures there is a possibility of the reception of duplicate messages or message segments during a transaction. At the client, a transaction begins, and a Transaction State Machine is created, when the first or only segment of a Confirmed-Request APDU is sent. The transaction ends when the client discards the Transaction State Machine due to one of the following circumstances:

(a) after reception from the server of a SimpleACK, unsegmented ComplexACK, Error, Reject, or Abort APDU containing the transaction's invoke-id;

(b) after transmission to the server of a SegmentACK APDU for the final segment of a segmented ComplexACK APDU received from the server;

(c) after exhausting the timeout and retry logic described in the previous clauses;

(d) after transmission to the server of an Abort APDU containing the transaction's invoke-id (i.e. the client aborts the transaction).

5.3.5.2 Terminating Server TSMs

At the server, a transaction begins, and a Transaction State Machine is created when the first or only segment of a Confirmed-Request APDU is received. The transaction ends when the server discards the Transaction State Machine due to one of the following circumstances:

(a) after transmission to the client of a SimpleACK, unsegmented ComplexACK, Error, Reject, or Abort APDU containing the transaction's invoke-id;

(b) after reception from the client of a SegmentACK APDU for the final segment of a segmented ComplexACK APDU transmitted by the server;

(c) after reception from the client of an Abort APDU containing the transaction's invoke-id;

(d) after exhausting the timeout and retry logic described in the previous clauses during the transmission of a segmented ComplexACK APDU.

5.3.5.3 Duplicate Message Procedures

The procedure for handling duplicate messages and message segments is as follows:

(a) The server receives a duplicate Confirmed-Request message. If the server has the capability of detecting a duplicate Confirmed-Request message, the message shall be discarded. If the server cannot distinguish between duplicate and non-duplicate messages, then the Confirmed-Request message shall be serviced. In this case, the client shall discard the server's response since the Invoke ID of the response will not bind to an active state transaction state machine.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(b) The server receives a duplicate Confirmed-Request message segment, that is, one that has already been acknowledged with a SegmentACK. In this case, the server shall discard the duplicate segment but shall return an appropriate SegmentACK APDU. A segment can be identified uniquely by the peer address, Invoke ID, and Sequence Number of the segment.

(c) The client receives a duplicate ComplexACK segment, that is, one that has already been acknowledged with a SegmentACK. In this case, the client shall discard the duplicate segment but shall return an appropriate SegmentACK APDU. A segment can be identified uniquely by the peer address, Invoke Id, and Sequence Number of the segment.

(d) A Device receives a duplicate SegmentACK APDU. In this case, the device shall discard the duplicate SegmentACK APDU. Other actions, including the possible re-sending of message segments, shall occur as specified in Clause 5.4.

5.3.6 Stale Resource Disposal

The error recovery procedure described here requires resources from both the server and the client. In the event that the error recovery process fails, the resources dedicated to this process need to be freed. In general, the resources that need to be freed are transaction specific and consist of a Transaction State Machine (TSM), timers, and APDU or APDU segment buffers. The exact time period before the resources should be freed is a local matter dependent upon the system design. As a design suggestion, it is recommended that resources should be considered stale and consequently freed:

(a) at the client, when a complete response to the Confirmed-Request APDU is received;

(b) at the client, when a Confirmed-Request APDU has been retransmitted the number of times specified in the Number_Of_APDU_Retries property without success;

(c) at the client, when a Confirmed-Request APDU segment has been retransmitted the number of times specified in the Number_Of_APDU_Retries property without success;

(d) at the server, when a complete response to a Confirmed-Request APDU has been transmitted and any associated SegmentACK received;

(e) at the server, when a ComplexACK APDU segment has been retransmitted the number of times specified in the Number_Of_APDU_Retries property without success;

(f) at any device, when a SegmentACK APDU has been transmitted and additional segments have not been received before the segment timeout expires.

5.4 Application Protocol State Machines

BACnet APDUs may be divided into two classes: those sent by requesting BACnet-users (clients) and those sent by responding BACnet-users (servers). All BACnet devices shall be able to act as responding BACnet-users and therefore shall be prepared to receive APDUs sent by requesting BACnet-users. Many devices will also be able to act as requesting BACnet-users, and such devices shall be prepared to receive APDUs sent by responding BACnet-users.

APDUs sent by requesting BACnet-users (clients):

BACnet-Unconfirmed-Request-PDU
BACnet-Confirmed-Request-PDU
BACnet-SegmentACK-PDU with 'server' = FALSE
BACnet-Abort-PDU with 'server' = FALSE

APDUs sent by responding BACnet-users (servers):

BACnet-SimpleACK-PDU
BACnet-ComplexACK-PDU
BACnet-Error-PDU
BACnet-Reject-PDU
BACnet-SegmentACK-PDU with 'server' = TRUE
BACnet-Abort-PDU with 'server' = TRUE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Both the requesting and the responding BACnet-user shall create and maintain a Transaction State Machine (TSM) for each transaction. The TSM shall be created when the transaction begins and shall be disposed of when the transaction ends. In the state machine descriptions that follow, the creation of a TSM is represented by a transition out of the IDLE state, and the disposal of a TSM is represented by a transition into the IDLE state. A transaction is uniquely identified by the client BACnetAddress, the server BACnetAddress, and the Invoke ID (if any).

When a PDU is received from the network layer, the PDU type, the source and destination BACnetAddresses, and the Invoke ID (if any) of the PDU shall be examined to determine the type (requesting BACnet-user or responding BACnet-user) and the identity of the TSM to which the PDU shall be passed. If no such TSM exists, one shall be created.

When a request is received from the application program, the request type, the source and destination BACnetAddresses, and the Invoke ID (if any) of the request shall be examined to determine the type (requesting BACnet-user or responding BACnet-user) and the identity of the TSM to which the request shall be passed. If no such TSM exists, one shall be created.

In order to simplify the state machine description, only the case of segmentation by the Application Entity is shown. Segmentation by the Application Program is possible as well. In this case, wherever the current TSM receives a segment or group of segments and sends SegmentACK, the modified TSM would instead pass the segments to the Application Program, and SegmentACK would be sent only upon direction from the Application Program via the SEGMENT_ACK.request primitive. Reception by the modified state machine of a SegmentACK-PDU would cause it to pass a SEGMENT_ACK.indication primitive to the Application Program.

5.4.1 Variables And Parameters

The following variables are defined for each instance of Transaction State Machine:

RetryCount used to count APDU retries

SegmentRetryCount used to count segment retries

DuplicateCount used to count duplicate segments

SentAllSegments used to control APDU retries and the acceptance of server replies

LastSequenceNumber stores the sequence number of the last segment received in order

InitialSequenceNumber stores the sequence number of the first segment of a sequence of segments that fill a window

ActualWindowSize stores the current window size

ProposedWindowSize stores the window size proposed by the segment sender

SegmentTimer used to perform timeout on PDU segments

RequestTimer used to perform timeout on Confirmed Requests

The following parameters are used in the description:

T_{seg} This parameter is the length of time a node shall wait for a SegmentACK-PDU after sending the final segment of a sequence. Its value is the value of the APDU_Segment_Timeout property of the node's Device object.

$T_{wait_for_seg}$ This parameter is the length of time a node shall wait after sending a SegmentACK-PDU for an additional segment of the message. Its value is equal to four times the value of the APDU_Segment_Timeout property of the node's Device object.

T_{out} This parameter represents the value of the APDU_Timeout property of the node's Device object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

N_{retry} This parameter represents the value of the Number_Of_APDU_Retries property of the node's Device object.

N_{dup} This parameter represents the number of duplicates that will be silently dropped per window before a negative segment ack is returned. This parameter shall be equal to ActualWindowSize.

5.4.2 Window Query Functions

5.4.2.1 Function InWindow

The function "InWindow" performs a modulo 256 compare of two unsigned eight-bit sequence numbers. All computations and comparisons are modulo 256 operations on unsigned eight-bit quantities.

function InWindow(seqA, seqB)

(1) if seqA minus seqB, modulo 256, is less than ActualWindowSize, then return TRUE.
(2) else return FALSE.

Example (not normative): if ActualWindowSize is equal to 4, then
InWindow(0, 0) returns TRUE
InWindow(1, 0) returns TRUE
InWindow(3, 0) returns TRUE
InWindow(4, 0) returns FALSE
InWindow(4, 5) returns FALSE (since the modulo 256 difference 4 - 5 = 255)
InWindow(0, 255) returns TRUE (since the modulo 256 difference 0 - 255 = 1)

5.4.2.2 Function DuplicateInWindow

The function "DuplicateInWindow" determines whether a value, seqA, is within the range firstSeqNumber through lastSequenceNumber, modulo 256, or if called at the start of a new Window and no new message segments have been received yet, it determines if the value seqA is within the range of the previous Window. All computations and comparisons are modulo 256 operations on unsigned eight-bit quantities.

function DuplicateInWindow(seqA, firstSeqNumber, lastSequenceNumber)

(1) Set local variable receivedCount to lastSequenceNumber minus firstSeqNumber, modulo 256.
(2) If receivedCount is greater than ActualWindowSize, then return FALSE.
(3) If seqA minus firstSeqNumber, modulo 256, is less than or equal to receivedCount, then return TRUE.
(4) If receivedCount is zero and firstSeqNumber minus seqA, modulo 256, is less than or equal to ActualWindowSize, then return TRUE.
(5) Else return FALSE.

Example (not normative): if ActualWindowSize is equal to 4, then
DuplicateInWindow(0, 0, 1) returns TRUE
DuplicateInWindow(1, 0, 1) returns TRUE
DuplicateInWindow(2, 0, 1) returns FALSE
DuplicateInWindow(3, 0, 1) returns FALSE

5.4.3 Function FillWindow

The function "FillWindow" sends PDU segments either until the window is full or until the last segment of a message has been sent. No more than **T_{seg}** may be allowed to elapse between the receipt of a SegmentACK APDU and the transmission of a segment. No more than **T_{seg}** may be allowed to elapse between the transmission of successive segments of a sequence.

function FillWindow(sequenceNumber)

(1) Set local variable ix to zero.
(2) If the next segment to transmit (the segment numbered sequenceNumber plus ix) is the final segment, goto step (g).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(3) Issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit the next BACnet APDU segment, with 'segmented-message' = TRUE, 'more-follows' = TRUE, 'proposed-window-size' equal to ProposedWindowSize, and 'sequence-number' = sequenceNumber plus ix, modulo 256.
(4) Set ix equal to ix plus one.
(5) If ix is less than ActualWindowSize, goto step (2).
(6) Goto step (9).
(7) Issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit the final BACnet APDU segment with 'segmented-message' = TRUE, 'more-follows' = FALSE, 'proposed-window-size' = ProposedWindowSize, and 'sequence-number' = sequenceNumber plus ix, modulo 256.
(8) Set SentAllSegments to TRUE, indicating that all segments have been transmitted at least once.
(9) Return to the caller.

5.4.4 State Machine for Requesting BACnet User (client)

5.4.4.1 IDLE

In the IDLE state, the device waits for the local application program to request a service.

SendUnconfirmed
If UNCONF_SERV.request is received from the local application program,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Unconfirmed-Request-PDU, and enter the IDLE state.

SendConfirmedUnsegmented
If CONF_SERV.request is received from the local application program and the length of the APDU is less than or equal to maximum-transmittable-length as determined according to Clause 5.2.1,

then assign an 'invoke-id' to this transaction; set SentAllSegments to TRUE; set RetryCount to zero; start RequestTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit a BACnet-Confirmed-Request-PDU with 'segmented-message' = FALSE; and enter the AWAIT_CONFIRMATION state to await a reply.

CannotSend
If CONF_SERV.request is received from the local application program and the length of the APDU is greater than maximum-transmittable-length as determined according to Clause 5.2.1 and the Max_Segments_Accepted property of the destination's Device object is known and the total APDU cannot be transmitted without exceeding the maximum number of segments accepted,

then send an ABORT.indication with 'server' = FALSE and 'abort-reason' =APDU_TOO_LONG to the local application program and enter the IDLE state.

SendConfirmedSegmented
If CONF_SERV.request is received from the local application program and the length of the APDU is greater than maximum-transmittable-length as determined according to Clause 5.2.1, and the Max_Segments_Accepted property of the destination's Device object is not known, or Max_Segments_Accepted is known and the total APDU can be transmitted without exceeding the maximum number of segments accepted,

then assign an 'invoke-id' to this transaction; set SentAllSegments to FALSE; set RetryCount to zero; set SegmentRetryCount to zero; set InitialSequenceNumber to zero; set ProposedWindowSize to whatever value is desired; set ActualWindowSize to 1; start SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit a BACnet-Confirmed-Request-PDU containing the first segment of the message, with 'segmented-message' = TRUE, 'more-follows' = TRUE, 'sequence-number' = zero, and 'proposed-window-size' = ProposedWindowSize; and enter the SEGMENTED_REQUEST state to await an acknowledgment. (The method used to determine ProposedWindowSize is a local matter, except that the value shall be in the range 1 to 127, inclusive.)

UnexpectedSegmentInfoReceived
If an unexpected PDU indicating the existence of an active server TSM (BACnet-ComplexACK-PDU with 'segmented-message' = TRUE or BACnet-SegmentACK-PDU with 'server' = TRUE) is received from the network layer,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE and 'abort-reason' = INVALID_APDU_IN_THIS_STATE and enter the IDLE state.

UnexpectedPDU_Received

If an unexpected PDU not indicating the existence of an active server TSM (BACnet-SimpleACK-PDU, BACnet-ComplexACK-PDU with 'segmented-message' = FALSE, BACnet-Error-PDU, BACnet-Reject-PDU, or BACnet-Abort-PDU with 'server' = TRUE) is received from the network layer,

then enter the IDLE state. (There is no reason to issue REJECT.indication, ABORT.indication, etc., as the client has no knowledge of the transaction in question.)

SecurityError_Received

If a security error is received via an N-REPORT.indication from the network layer for which there is no active TSM associated with it,

then enter the IDLE state.

5.4.4.2 SEGMENTED_REQUEST

In the SEGMENTED_REQUEST state, the device waits for a BACnet-SegmentACK-PDU for one or more segments of a BACnet-Confirmed-Request-PDU.

SecurityError_Received

If a security error is received via an N-REPORT.indication from the network layer,

then stop SegmentTimer; send SEC_ERR.indication with the appropriate security error information to the local application program; and enter the IDLE state.

InsufficientSecurity_Received

If a PDU is received from the network layer and the security parameters do not match the initial request,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE and an 'abort-reason' = INSUFFICIENT_SECURITY describing the security error; send SEC_ERR.indication indicating insufficient security to the local application program; and enter the IDLE state.

DuplicateACK_Received

If a BACnet-SegmentACK-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer and InWindow ('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of FALSE,

then restart SegmentTimer and enter the SEGMENTED_REQUEST state to await an acknowledgment.

NewACK_Received

If a BACnet-SegmentACK-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer and InWindow ('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of TRUE and there is at least one segment remaining to send,

then set InitialSequenceNumber equal to the 'sequence-number' parameter of the BACnet-SegmentACK-PDU plus one, modulo 256; set ActualWindowSize equal to the 'actual-window-size' parameter of the BACnet-SegmentACK-PDU; set SegmentRetryCount to zero; call FillWindow (InitialSequenceNumber) to transmit one or more BACnet-Confirmed-Request-PDUs containing the next ActualWindowSize segments of the message; restart SegmentTimer; and enter the SEGMENTED_REQUEST state to await an acknowledgment.

FinalACK_Received

If a BACnet-SegmentACK-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer and InWindow ('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of TRUE and there are no more segments to send,

then stop SegmentTimer; start RequestTimer; and enter the AWAIT_CONFIRMATION state to await a reply.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Timeout

If SegmentTimer becomes greater than T_{seg} and SegmentRetryCount is less than N_{retry},

then increment SegmentRetryCount; call FillWindow(InitialSequenceNumber) to retransmit one or more BACnet-Confirmed-Request-PDUs containing the next ActualWindowSize segments of the message; restart SegmentTimer; and enter the SEGMENTED_REQUEST state to await an acknowledgment.

FinalTimeout

If SegmentTimer becomes greater than T_{seg} and SegmentRetryCount is greater than or equal to N_{retry},

then stop SegmentTimer; send ABORT.indication with 'server' = FALSE and 'abort-reason' = TSM_TIMEOUT to the local application program; and enter the IDLE state.

AbortPDU_Received

If a BACnet-Abort-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer,

then stop SegmentTimer; send ABORT.indication to the local application program; and enter the IDLE state.

SimpleACK_Received

If a BACnet-SimpleACK-PDU that has sufficient security parameters is received from the network layer and SentAllSegments is TRUE,

then stop SegmentTimer; send CONF_SERV.confirm(+) to the local application program; and enter the IDLE state.

UnsegmentedComplexACK_Received

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is FALSE and SentAllSegments is TRUE,

then stop SegmentTimer; send CONF_SERV.confirm(+) to the local application program; and enter the IDLE state.

SegmentedComplexACK_Received

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is zero and this device supports segmentation and SentAllSegments is TRUE,

then save the BACnet-ComplexACK-PDU segment; stop SegmentTimer; compute ActualWindowSize based on the 'proposed-window-size' parameter of the received BACnet-ComplexACK-PDU and on local conditions; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server' = FALSE, and 'actual-window-size' = ActualWindowSize; start SegmentTimer; set LastSequenceNumber to zero; set InitialSequenceNumber to zero; set DuplicateCount to zero; and enter the SEGMENTED_CONF state to receive the remaining segments. (The method used to determine ActualWindowSize is a local matter, except that the value shall be less than or equal to the 'proposed-window-size' parameter of the received BACnet-ComplexACK-PDU and shall be in the range 1 to 127, inclusive.)

ErrorPDU_Received

If a BACnet-Error-PDU that has sufficient security parameters is received from the network layer and SentAllSegments is TRUE,

then stop SegmentTimer; send CONF_SERV.confirm(-) to the local application program; and enter the IDLE state.

RejectPDU_Received

If a BACnet-Reject-PDU that has sufficient security parameters is received from the network layer,

then stop SegmentTimer; send REJECT.indication to the local application program; and enter the IDLE state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

UnexpectedPDU_Received

If a BACnet-SimpleACK-PDU, BACnet-ComplexACK-PDU, or BACnet-Error-PDU that has sufficient security parameters is received from the network layer and SentAllSegments is FALSE,

or if a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE and this device does not support segmentation,

or if a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not zero,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; send ABORT.indication with 'server' = FALSE and 'abort-reason' = INVALID_APDU_IN_THIS_STATE to the local application program; and enter the IDLE state.

SendAbort

If ABORT.request is received from the local application program,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; and enter the IDLE state.

5.4.4.3 AWAIT_CONFIRMATION

In the AWAIT_CONFIRMATION state, the device waits for a response to a BACnet-Confirmed-Request-PDU.

SecurityError_Received

If a security error is received via an N-REPORT.indication from the network layer,

then stop RequestTimer; send SEC_ERR.indication with the appropriate security error information to the local application program; and enter the IDLE state.

InsufficientSecurity_Received

If a PDU is received from the network layer and the security parameters do not match the initial request,

then stop RequestTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE and an 'abort-reason' = INSUFFICIENT_SECURITY; send SEC_ERR.indication indicating insufficient security to the local application program; and enter the IDLE state.

SimpleACK_Received

If a BACnet-SimpleACK-PDU that has sufficient security parameters is received from the network layer,

then stop RequestTimer; send CONF_SERV.confirm(+) to the local application program; and enter the IDLE state.

UnsegmentedComplexACK_Received

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is FALSE,

then stop RequestTimer; send CONF_SERV.confirm(+) to the local application program; and enter the IDLE state.

SegmentedComplexACK_Received

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is zero and this device supports segmentation,

then stop RequestTimer; compute ActualWindowSize based on the 'proposed-window-size' parameter of the received BACnet-ComplexACK-PDU and on local conditions; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server' = FALSE, and 'actual-window-size' = ActualWindowSize; start SegmentTimer; set LastSequenceNumber to zero; set InitialSequenceNumber to zero; set DuplicateCount to zero; and enter the SEGMENTED_CONF state to

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

receive the remaining segments. (The method used to determine ActualWindowSize is a local matter, except that the value shall be less than or equal to the 'proposed-window-size' parameter of the received BACnet-ComplexACK-PDU and shall be in the range 1 to 127, inclusive.)

ErrorPDU_Received
If a BACnet-Error-PDU that has sufficient security parameters is received from the network layer,

then stop RequestTimer; send CONF_SERV.confirm(-) to the local application program; and enter the IDLE state.

RejectPDU_Received
If a BACnet-Reject-PDU that has sufficient security parameters is received from the network layer,

then stop RequestTimer; send REJECT.indication to the local application program; and enter the IDLE state.

AbortPDU_Received
If a BACnet-Abort-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer,

then stop RequestTimer; send ABORT.indication to the local application program; and enter the IDLE state.

SegmentACK_Received
If a BACnet-SegmentACK-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer,

then discard the PDU as a duplicate, and re-enter the current state.

UnexpectedPDU_Received
If an unexpected PDU (BACnet-ComplexACK-PDU with 'segmented-message' = TRUE and 'sequence-number' not equal to zero or 'segmented-message' = TRUE and this device does not support segmentation) that has sufficient security parameters is received from the network layer,

then stop RequestTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; send ABORT.indication with 'server' = FALSE and 'abort-reason' = INVALID_APDU_IN_THIS_STATE to the local application program; and enter the IDLE state.

TimeoutUnsegmented
If RequestTimer becomes greater than $\mathbf{T_{out}}$ and RetryCount is less than Number_Of_APDU_Retries and the length of the Confirmed Request APDU is less than or equal to maximum-transmittable-length as determined according to Clause 5.2.1,

then stop RequestTimer; increment RetryCount; issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit a BACnet-Confirmed-Request-PDU with 'segmented-message' = FALSE; start RequestTimer; and enter the AWAIT_CONFIRMATION state to await a reply.

TimeoutSegmented
If RequestTimer becomes greater than $\mathbf{T_{out}}$ and RetryCount is less than Number_Of_APDU_Retries and the length of the Confirmed-Request APDU is greater than maximum-transmittable-length as determined according to Clause 5.2.1,

then stop RequestTimer; increment RetryCount; set SegmentRetryCount to zero; set SentAllSegments to FALSE; start SegmentTimer; set InitialSequenceNumber to zero; set ActualWindowSize to 1; issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit a BACnet-Confirmed-Request-PDU containing the first segment of the message, with 'segmented-message' = TRUE, 'more-follows' = TRUE, and 'sequence-number' = zero; and enter the SEGMENTED_REQUEST state to await an acknowledgment.

FinalTimeout
If RequestTimer becomes greater than $\mathbf{T_{out}}$ and RetryCount is greater than or equal to Number_Of_APDU_Retries,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then stop RequestTimer; send ABORT.indication with 'server' = FALSE and 'abort-reason' = TSM_TIMEOUT to the local application program; and enter the IDLE state.

SendAbort

If ABORT.request is received from the local application program,

then stop RequestTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; and enter the IDLE state.

5.4.4.4 SEGMENTED_CONF

In the SEGMENTED_CONF state, the device waits for one or more segments in response to a BACnet-SegmentACK-PDU.

SecurityError_Received

If a security error is received via an N-REPORT.indication from the network layer,

then stop SegmentTimer; send SEC_ERR.indication with the appropriate security error information to the local application program; and enter the IDLE state.

InsufficientSecurity_Received

If a PDU is received from the network layer and the security parameters that do not match the initial request,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE and an 'abort-reason' = INSUFFICIENT_SECURITY; send SEC_ERR.indication indicating insufficient security to the local application program; and enter the IDLE state.

NewSegmentReceived_NoSpace

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; and the segment cannot be saved due to local conditions,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE and 'abort-reason' = BUFFER_OVERFLOW; send ABORT.indication with 'server' = FALSE and 'abort-reason' = BUFFER_OVERFLOW to the local application program; and enter the IDLE state.

NewSegmentReceived

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'more-follows' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; and whose 'sequence-number' parameter is not equal to InitialSequenceNumber plus ActualWindowSize, modulo 256,

then save the BACnet-ComplexACK-PDU segment; increment LastSequenceNumber, modulo 256; restart SegmentTimer; issue a N-RELEASE.request to the network layer entity specifying the source_address value of the segment for the destination_address parameter (in case the data link layer is waiting for a reply); and enter the SEGMENTED_CONF state to receive additional segments.

LastSegmentOfGroupReceived

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; whose 'more-follows' parameter is TRUE; and whose 'sequence-number' parameter is equal to InitialSequenceNumber plus ActualWindowSize, modulo 256,

then save the BACnet-ComplexACK-PDU segment; increment LastSequenceNumber, modulo 256; set InitialSequenceNumber to LastSequenceNumber; set DuplicateCount to zero; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, server = FALSE, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; and enter the SEGMENTED_CONF state to receive additional segments.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

LastSegmentOfComplexACK_Received

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; and whose 'more-follows' parameter is FALSE (i.e. the final segment),

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server = FALSE', and 'actual-window-size' = ActualWindowSize; send CONF_SERV.confirm(+) containing all of the received segments to the local application program; and enter the IDLE state.

DuplicateSegmentReceived

If a BACnet-ComplexACK-PDU is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of TRUE and DuplicateCount is less than Ndup,

then discard the BACnet-ComplexACK-PDU segment; restart SegmentTimer; increment DuplicateCount, and enter the SEGMENTED_CONF state to receive the remaining segments.

TooManyDuplicateSegmentsReceived

If a BACnet-ComplexACK-PDU is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of TRUE and DuplicateCount is equal to Ndup,

then discard the BACnet-ComplexACK-PDU segment; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = TRUE, 'server' = FALSE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; set DuplicateCount to zero; and enter the SEGMENTED_CONF state to receive the remaining segments.

SegmentReceivedOutOfOrder

If a BACnet-ComplexACK-PDU that has sufficient security parameters is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of FALSE,

then discard the BACnet-ComplexACK-PDU segment; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = TRUE, 'server' = FALSE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; set InitialSequenceNumber = LastSequenceNumber; set DuplicateCount to zero; and enter the SEGMENTED_CONF state to receive the remaining segments.

AbortPDU_Received

If a BACnet-Abort-PDU that has sufficient security parameters and whose 'server' parameter is TRUE is received from the network layer,

then stop SegmentTimer; send ABORT.indication to the local application program; and enter the IDLE state.

UnexpectedPDU_Received

If an unexpected PDU (BACnet-SimpleACK-PDU, BACnet-ComplexACK-PDU with 'segmented-message' = FALSE, BACnet-Error-PDU, BACnet-Reject-PDU, or BACnet-SegmentACK-PDU with 'server' = TRUE) that has sufficient security parameters is received from the network layer,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; send ABORT.indication with 'server' = FALSE and 'abort-reason' = INVALID_APDU_IN_THIS_STATE to the local application program; and enter the IDLE state.

Timeout
If SegmentTimer becomes greater than T_{seg} times four,

then stop SegmentTimer; send ABORT.indication with 'server' = FALSE and 'abort-reason' = TSM_TIMEOUT to the local application program; and enter the IDLE state.

SendAbort
If ABORT.request is received from the local application program,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = FALSE; and enter the IDLE state.

5.4.5 State Machine for Responding BACnet User (server)

5.4.5.1 IDLE

In the IDLE state, the device waits for a PDU from the network layer.

SecurityError_Received
If a security error is received via an N-REPORT.indication from the network layer,

then enter the IDLE state.

UnconfirmedReceived
If a BACnet-Unconfirmed-Request-PDU is received from the network layer,

then send an UNCONF_SERV.indication to the local application program, and enter the IDLE state.

ConfirmedBroadcastReceived
If a BACnet-Confirmed-Request-PDU whose destination address is a multicast or broadcast address is received from the network layer,

then enter the IDLE state.

ConfirmedUnsegmentedReceived
If a BACnet-Confirmed-Request-PDU whose 'segmented-message' parameter is FALSE is received from the network layer,

then send a CONF_SERV.indication to the local application program, start RequestTimer; and enter the AWAIT_RESPONSE state.

ConfirmedSegmentedReceivedNotSupported
If a BACnet-Confirmed-Request-PDU whose 'segmented-message' parameter is TRUE is received from the network layer, and this device does not support the reception of segmented messages,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = SEGMENTATION_NOT_SUPPORTED, and enter the IDLE state.

ConfirmedSegmentedReceived
If a BACnet-Confirmed-Request-PDU whose 'segmented-message' parameter is TRUE, whose 'sequence-number' parameter is zero, and whose 'proposed-window-size' is greater than zero and less than or equal to 127 is received from the network layer and the local device supports the reception of segmented messages,

then save the BACnet-Confirmed-Request-PDU segment; compute ActualWindowSize based on the 'proposed-window-size' parameter of the received BACnet-Confirmed-Request-PDU and on local conditions; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'negative-ack' = FALSE, 'server' = TRUE, and 'actual-window-size' = ActualWindowSize; start SegmentTimer; set LastSequenceNumber to zero; set InitialSequenceNumber to zero; set DuplicateCount to zero; and enter the SEGMENTED_REQUEST state to receive the remaining segments. (The method used to determine ActualWindowSize is a local matter, except that the value shall be less than or equal to the 'proposed-window-size' parameter of the received BACnet-Confirmed-Request-PDU and shall be in the range 1 to 127, inclusive.)

ConfirmedSegmentedReceivedWindowSizeOutOfRange
If a BACnet-Confirmed-Request-PDU whose 'segmented-message' parameter is TRUE, whose 'sequence-number' parameter is zero, and whose 'proposed-window-size' is zero or greater than 127 is received from the network layer and the local device supports the reception of segmented messages,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = WINDOW_SIZE_OUT_OF_RANGE, and enter the IDLE state.

AbortPDU_Received
If a BACnet-Abort-PDU whose 'server' parameter is FALSE is received from the network layer,

then enter the IDLE state.

UnexpectedPDU_Received
If an unexpected PDU (BACnet-Confirmed-Request-PDU with 'segmented-message' = TRUE and 'sequence-number' not equal to zero or BACnet-SegmentACK-PDU with 'server' = FALSE) is received from the network layer,

then issue an N-RELEASE.request to the network layer entity specifying the source_address value of the segment for the destination_address parameter (to prevent the following abort response from being transmitted ahead of other response message frames that may be queued); then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and enter the IDLE state.

5.4.5.2 SEGMENTED_REQUEST

In the SEGMENTED_REQUEST state, the device waits for segments of a BACnet-Confirmed-Request-PDU.

IncorrectSecurityPdu_Received
If a PDU is received that is not secured with the same settings as the original PDU,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = INSUFFICIENT_SECURITY; and enter the IDLE state.

SecurityError_Received
If a security error is received via an N-REPORT.indication from the network layer,

then stop SegmentTimer and enter the IDLE state.

NewSegmentReceived
If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'more-follows' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; and whose 'sequence-number' parameter is not equal to InitialSequenceNumber plus ActualWindowSize, modulo 256,

then save the BACnet-Confirmed-Request-PDU segment; increment LastSequenceNumber, modulo 256; restart SegmentTimer; issue a N-RELEASE.request to the network layer entity specifying the source_address value of the segment for the destination_address parameter (in case the data link layer is waiting for a reply); and enter the SEGMENTED_REQUEST state to receive the remaining segments.

LastSegmentOfGroupReceived
If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'sequence-number' parameter is

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

equal to LastSequenceNumber plus 1, modulo 256; whose 'more-follows' parameter is TRUE; and whose 'sequence-number' parameter is equal to InitialSequenceNumber plus ActualWindowSize, modulo 256,

then save the BACnet-Confirmed-Request-PDU segment; increment LastSequenceNumber, modulo 256; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server' = TRUE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; set InitialSequenceNumber = LastSequenceNumber; set DuplicateCount to zero; and enter the SEGMENTED_REQUEST state to receive the remaining segments.

LastSegmentOfMessageReceived

If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU is received from the network layer whose 'segmented-message' parameter is TRUE; whose 'sequence-number' parameter is equal to LastSequenceNumber plus 1, modulo 256; and whose 'more-follows' parameter is FALSE (i.e. the final segment),

then save the BACnet-Confirmed-Request-PDU segment; increment LastSequenceNumber, modulo 256; stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server' = TRUE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; set InitialSequenceNumber = LastSequenceNumber; send CONF_SERV.indication(+) containing all of the received segments to the local application program; start RequestTimer; and enter the AWAIT_RESPONSE state.

DuplicateSegmentReceived

If a BACnet-Confirmed-Request-PDU is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of TRUE and DuplicateCount is less than Ndup,

then discard the BACnet-Confirmed-Request-PDU segment; restart SegmentTimer; increment DuplicateCount; and enter the SEGMENTED_REQUEST state to receive the remaining segments.

TooManyDuplicateSegmentsReceived

If a BACnet-Confirmed-Request-PDU is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of TRUE and DuplicateCount is equal to Ndup,

then discard the BACnet-Confirmed-Request-PDU segment; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = TRUE, 'server' = TRUE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; set InitialSequenceNumber = LastSequenceNumber; set DuplicateCount to zero; and enter the SEGMENTED_REQUEST state to receive the remaining segments.

SegmentReceivedOutOfOrder

If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU is received from the network layer whose 'segmented-message' parameter is TRUE and whose 'sequence-number' parameter is not equal to LastSequenceNumber plus 1, modulo 256, and DuplicateInWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber+1 modulo 256, LastSequenceNumber) returns a value of FALSE,

then discard the BACnet-Confirmed-Request-PDU segment; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = TRUE, 'server' = TRUE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; restart SegmentTimer; set InitialSequenceNumber = LastSequenceNumber; set DuplicateCount to zero; and enter the SEGMENTED_REQUEST state to receive the remaining segments.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

AbortPDU_Received
If a BACnet-Abort-PDU that is secured with the same settings as the original PDU and whose server parameter is FALSE is received from the network layer,

then stop SegmentTimer and enter the IDLE state.

UnexpectedPDU_Received
If an unexpected PDU (BACnet-Confirmed-Request-PDU with 'segmented-message' = FALSE or BACnet-SegmentACK-PDU with 'server' = FALSE) that is secured with the same settings as the original PDU is received from the network layer,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and 'abort-reason' = INVALID_APDU_IN_THIS_STATE and enter the IDLE state.

Timeout
If SegmentTimer becomes greater than $\mathbf{T_{seg}}$ times four,

then stop SegmentTimer and enter the IDLE state.

SendAbort
If ABORT.request is received from the local application program,

then stop SegmentTimer, issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE, and enter the IDLE state.

5.4.5.3 AWAIT_RESPONSE

In the AWAIT_RESPONSE state, the device waits for the local application program to respond to a BACnet-Confirmed-Request-PDU. See Clause 9.8 for specific considerations in MS/TP networks.

SecurityError_Received
If a security error is received via an N-REPORT.indication from the network layer,

then send SEC_ERR.indication with the appropriate security error information to the local application program; and enter the IDLE state.

IncorrectSecurityPdu_Received
If a PDU is received that is not secured with the same settings as the original PDU,

then discard the PDU, and re-enter the current state.

SendSimpleACK
If a CONF_SERV.response(+) is received from the local application program, which is to be conveyed via a BACnet-SimpleACK-PDU,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SimpleACK-PDU and enter the IDLE state.

SendUnsegmentedComplexACK
If a CONF_SERV.response(+) is received from the local application program, which is to be conveyed via a BACnet-ComplexACK-PDU, and the length of the APDU is less than or equal to maximum-transmittable-length as determined according to Clause 5.2.1,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-ComplexACK-PDU with 'segmented-message' = FALSE and enter the IDLE state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

CannotSendSegmentedComplexACK

If a CONF_SERV.response(+) is received from the local application program, which is to be conveyed via a BACnet-ComplexACK-PDU, and the length of the APDU is greater than maximum-transmittable-length as determined according to Clause 5.2.1, and either

(a) this device does not support the transmission of segmented messages or
(b) the client will not accept a segmented response (the 'segmented-response-accepted' parameter in BACnet-ConfirmedRequest-PDU is FALSE), or
(c) the client's max-segments-accepted parameter in the BACnet-ConfirmedRequest-PDU is fewer than required to transmit the total APDU when total size is known or,
(d) the number of segments transmittable by this device is fewer than required to transmit the total APDU when total size is known,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = SEGMENTATION_NOT_SUPPORTED for case (a) and (b), or BUFFER_OVERFLOW for case (c) and (d), and enter the IDLE state.

SendSegmentedComplexACK

If a CONF_SERV.response(+) is received from the local application program that is to be conveyed via a BACnet-ComplexACK-PDU, and the length of the APDU is greater than maximum-transmittable-length as determined according to Clause 5.2.1, and the device supports the transmission of segmented messages, and the client will accept a segmented response ('segmented-response-accepted' parameter in BACnet-ConfirmedRequest-PDU is TRUE),

then set SegmentRetryCount to zero; set InitialSequenceNumber to zero; set ProposedWindowSize to whatever value is desired; set ActualWindowSize to 1; start SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit a BACnet-ComplexACK-PDU containing the first segment of the message, with 'segmented-message' = TRUE, 'more-follows' = TRUE, 'sequence-number' = zero, and 'proposed-window-size' = ProposedWindowSize; and enter the SEGMENTED_RESPONSE state to await an acknowledgment.

SendErrorPDU

If a CONF_SERV.response(-) is received from the local application program,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Error-PDU and enter the IDLE state.

SendAbort

If ABORT.request is received from the local application program,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and enter the IDLE state.

SendReject

If REJECT.request is received from the local application program,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Reject-PDU; and enter the IDLE state.

AbortPDU_Received

If a BACnet-Abort-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE is received from the network layer,

then send ABORT.indication to the local application program; and enter the IDLE state.

DuplicateRequestReceived

If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU and whose 'segmented-message' parameter is FALSE is received from the network layer,

then discard the PDU as a duplicate request, and re-enter the current state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

DuplicateSegmentReceived

If a BACnet-Confirmed-Request-PDU that is secured with the same settings as the original PDU and whose 'segmented-message' parameter is TRUE is received from the network layer,

then discard the PDU as a duplicate segment; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-SegmentACK-PDU with 'negative-ack' = FALSE, 'server' = TRUE, 'sequence-number' = LastSequenceNumber, and 'actual-window-size' = ActualWindowSize; and re-enter the current state.

UnexpectedPDU_Received

If an unexpected PDU (BACnet-SegmentACK-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE) is received from the network layer,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; send ABORT.indication with 'server' = TRUE and 'abort-reason' = INVALID_APDU_IN_THIS_STATE to the local application program; and enter the IDLE state.

Timeout

If RequestTimer becomes greater than $\mathbf{T_{out}}$,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = APPLICATION_EXCEEDED_REPLY_TIME; send ABORT.indication with 'server' = TRUE and 'abort-reason' = APPLICATION_EXCEEDED_REPLY_TIME to the local application program; and enter the IDLE state.

5.4.5.4 SEGMENTED_RESPONSE

In the SEGMENTED_RESPONSE state, the device waits for a BACnet-SegmentACK-PDU for a segment or segments of a BACnet-ComplexACK-PDU.

SecurityError_Received

If a security error is received via an N-REPORT.indication from the network layer,

then stop SegmentTimer; send SEC_ERR.indication with the appropriate security error information to the local application program; and enter the IDLE state.

IncorrectSecurityPdu_Received

If a PDU is received that is not secured with the same settings as the original PDU,

then issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and enter the IDLE state.

DuplicateACK_Received

If a BACnet-SegmentACK-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE is received from the network layer and InWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of FALSE,

then restart SegmentTimer and enter the SEGMENTED_RESPONSE state to await an acknowledgment or timeout.

NewACK_Received

If a BACnet-SegmentACK-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE is received from the network layer and InWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of TRUE and there is at least one segment remaining to send,

then set InitialSequenceNumber equal to the 'sequence-number' parameter of the BACnet-SegmentACK-PDU plus one, modulo 256; set ActualWindowSize equal to the 'actual-window-size' parameter of the BACnet-SegmentACK-PDU; set SegmentRetryCount to zero; call FillWindow(InitialSequenceNumber) to issue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit one or more BACnet-ComplexACK-PDUs

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

containing the next ActualWindowSize segments of the message; restart SegmentTimer; and enter the SEGMENTED_RESPONSE state to await an acknowledgment.

FinalACK_Received

If a BACnet-SegmentACK-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE is received from the network layer and InWindow('sequence-number' parameter of the BACnet-SegmentACK-PDU, InitialSequenceNumber) returns a value of TRUE and there are no more segments to send,

then stop SegmentTimer and enter the IDLE state.

Timeout

If SegmentTimer becomes greater than $\mathbf{T_{seg}}$ and SegmentRetryCount is less than Number_Of_APDU_Retries,

then increment SegmentRetryCount; call FillWindow(InitialSequenceNumber) to reissue an N-UNITDATA.request with 'data_expecting_reply' = TRUE to transmit one or more BACnet-ComplexACK-PDUs containing the next ActualWindowSize segments of the message; restart SegmentTimer; and enter the SEGMENTED_RESPONSE state to await an acknowledgment.

FinalTimeout

If SegmentTimer becomes greater than $\mathbf{T_{seg}}$ and SegmentRetryCount is greater than or equal to Number_Of_APDU_Retries,

then stop the SegmentTimer, and enter the IDLE state.

AbortPDU_Received

If a BACnet-Abort-PDU that is secured with the same settings as the original PDU and whose 'server' parameter is FALSE is received from the network layer,

then stop SegmentTimer; send ABORT.indication to the local application program; and enter the IDLE state.

UnexpectedPDU_Received

If an unexpected PDU (BACnet-Confirmed-Request-PDU) that is secured with the same settings as the original PDU is received from the network layer,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and enter the IDLE state.

SendAbort

If ABORT.request is received from the local application program,

then stop SegmentTimer; issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE; and enter the IDLE state.

5.5 Application Protocol Time Sequence Diagrams

The flow sequence of service primitives can be represented by time-sequence diagrams. Each diagram is partitioned into three or four fields. The field labeled "Provider" represents the service-provider and the two fields labeled "User" represent the two service-users. The fourth field, if present, represents an application program. For the application layer, the vertical lines between user and provider represent the interface between the BACnet User Element and the BACnet ASE. For lower layers these vertical lines represent the service-access-points between the service-users and the service-provider. Moving from top to bottom in the diagram represents the passage of time. Arrows, placed in the areas representing the service-user, indicate the main flow of information during the execution of an interaction described by a service-primitive (i.e. to or from the service-user). Figures 5-4 through 5-13 illustrate the various sequences of application service primitives defined in BACnet.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

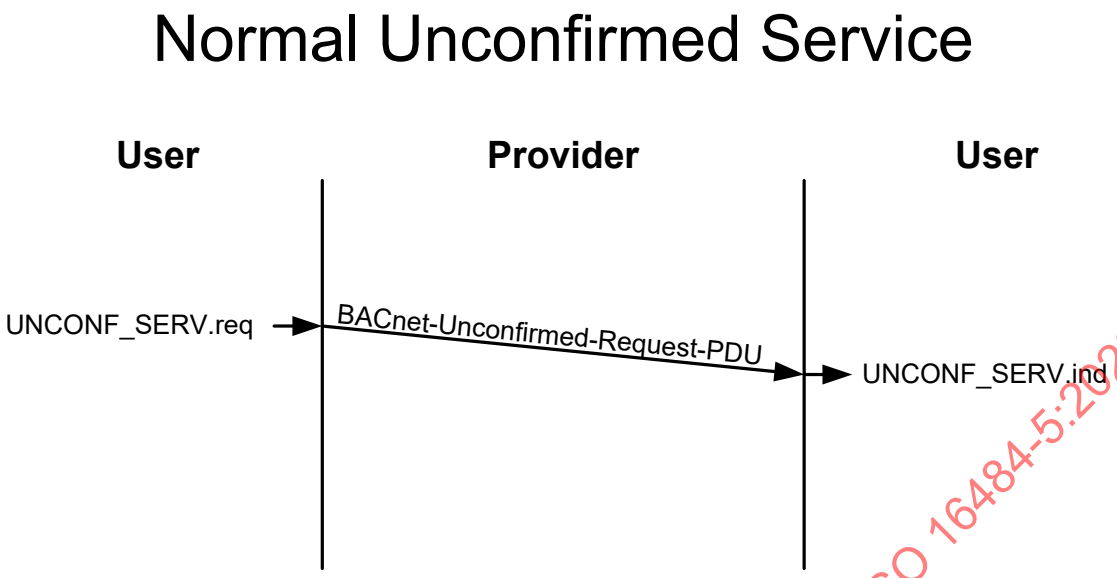


Figure 5-4. Time sequence diagram for a normal unconfirmed service.

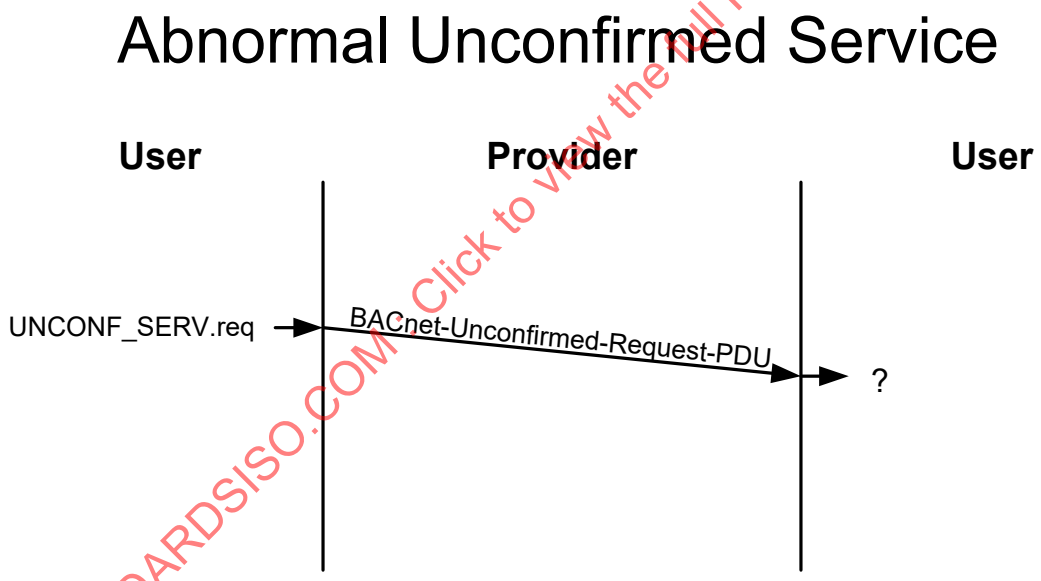


Figure 5-5. Time sequence diagram for an abnormal unconfirmed service. Unconfirmed service requests that are in some way flawed are ignored by the receiving user as indicated by the symbol "?".

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

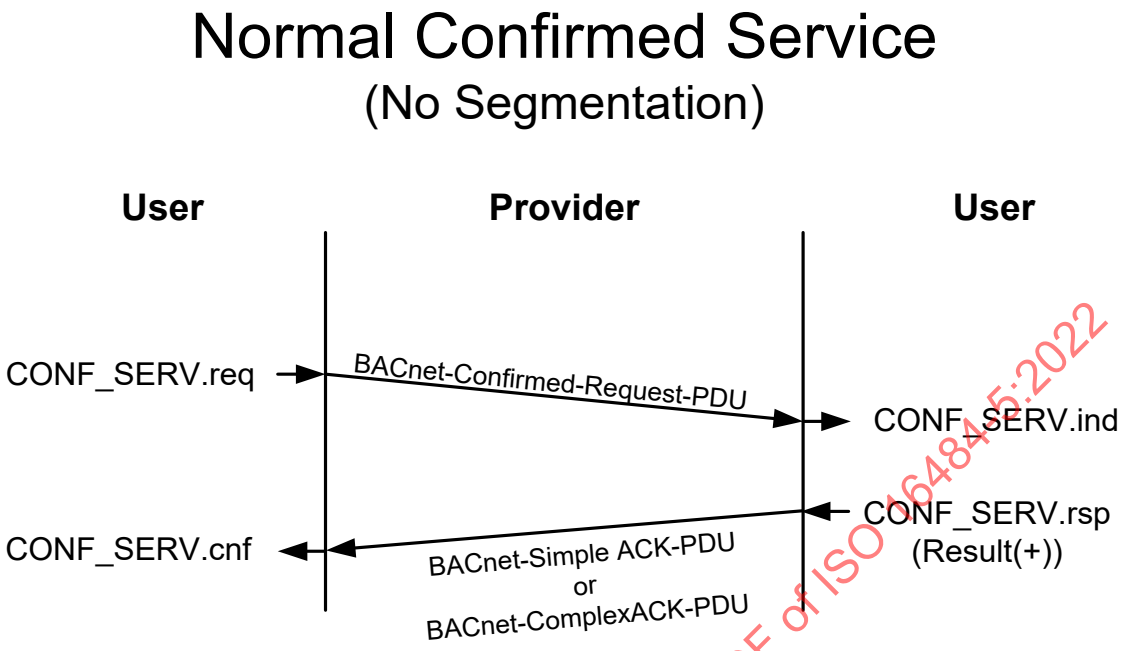


Figure 5-6. Time sequence diagram for normal confirmed services.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

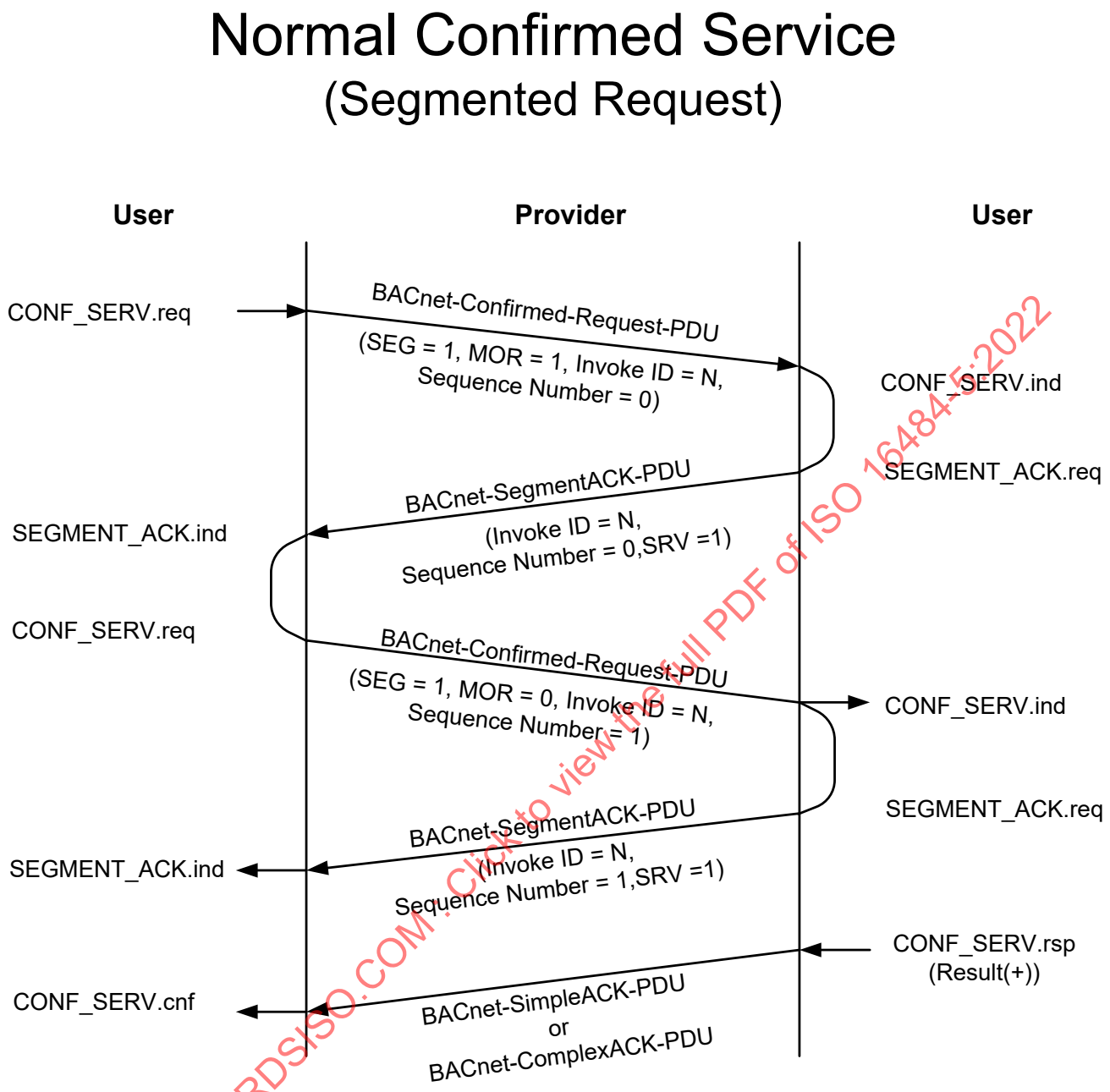


Figure 5-7. Time sequence diagram for a normal confirmed service with a segmented request.

Figure 5-7 illustrates two separate, interleaved exchanges of service primitives. One exchange is the usual confirmed service request, indication, response, and confirm sequence. Because the request is segmented it takes several CONF_SERV.request primitives to convey the entire request. The segment acknowledge service primitives, which are an independent exchange, are used to signal the client that the server is ready for the next segment.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

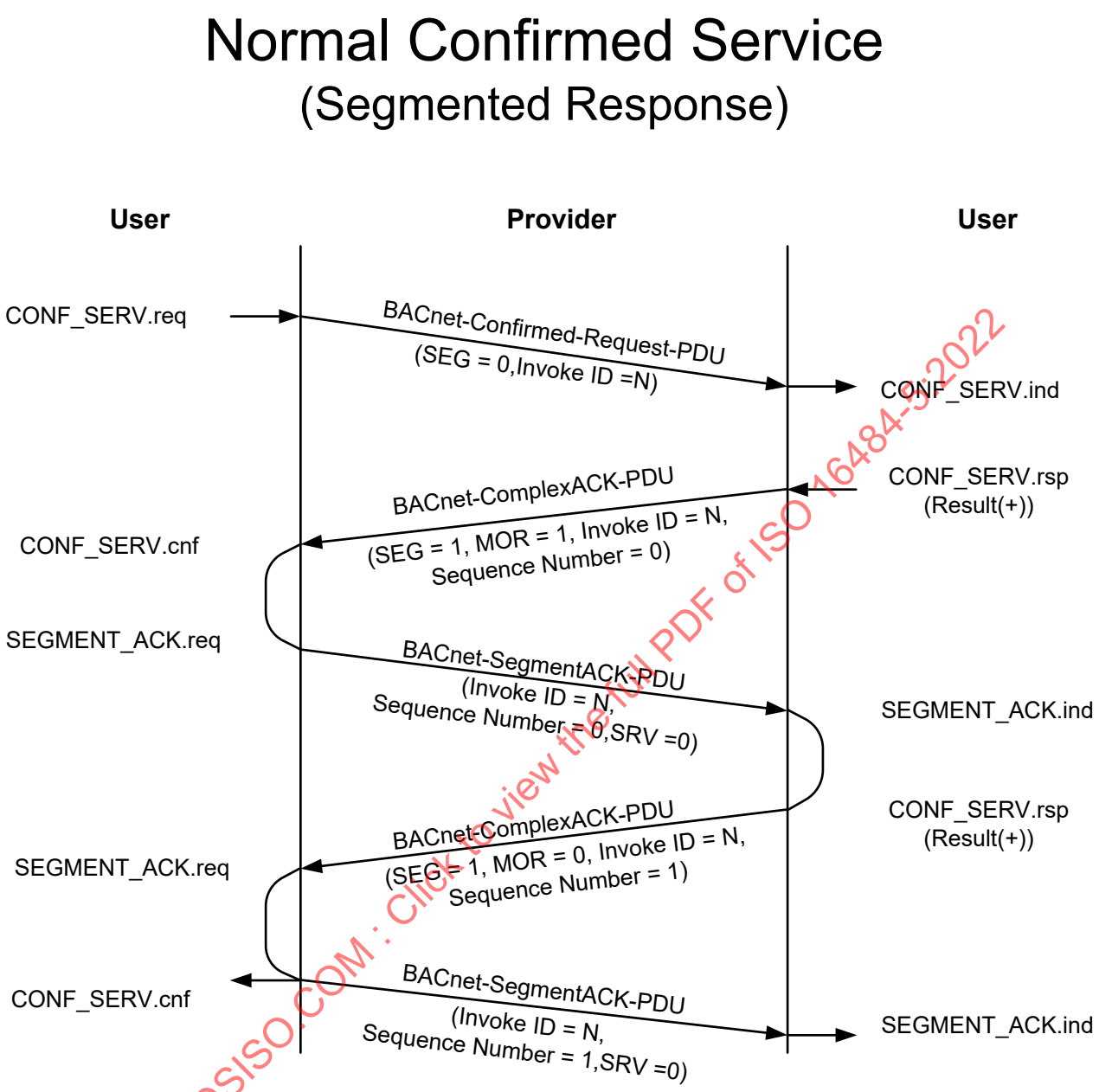
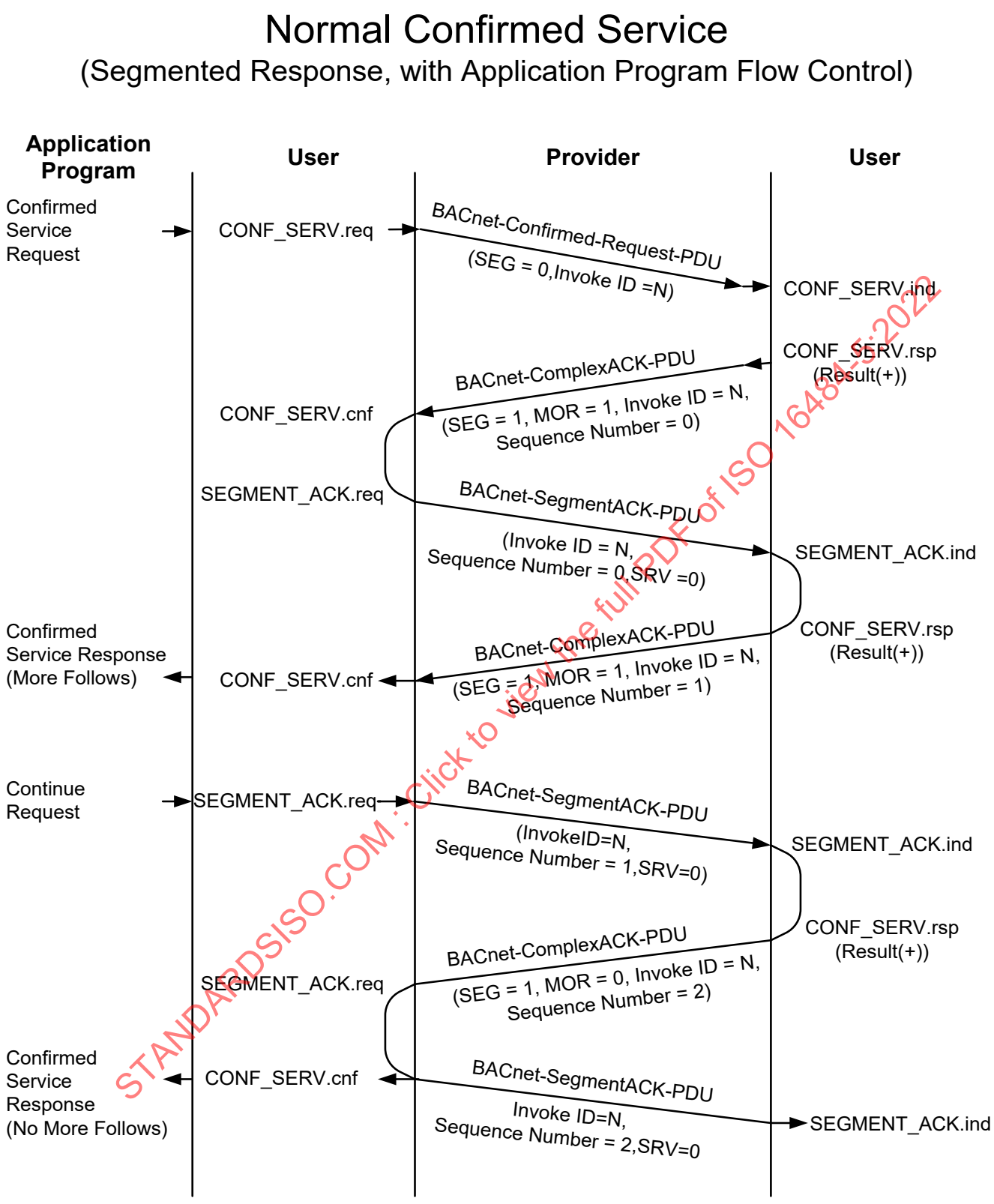


Figure 5-8. Time sequence diagram for a normal confirmed service with segmented response.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved



STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Figure 5-9. Time sequence diagram for a normal confirmed service with application flow control.

© ISO 2022 – All rights reserved

Normal Confirmed Service

(Segmented Response, with Application Program Flow Control and Requester Abort)

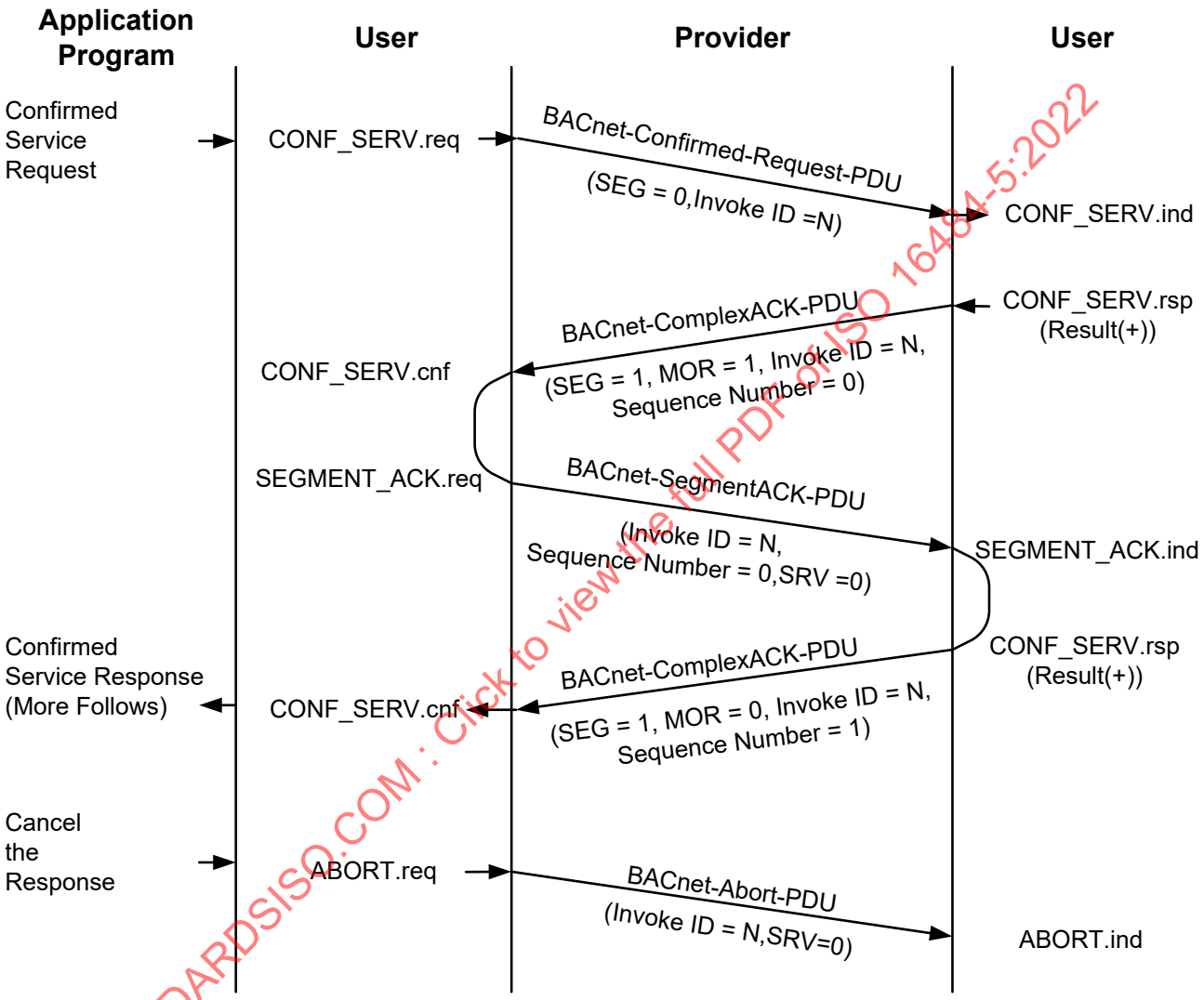


Figure 5-10. Time sequence diagram for a normal confirmed service with segmented response, application program flow control, and response cancellation.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

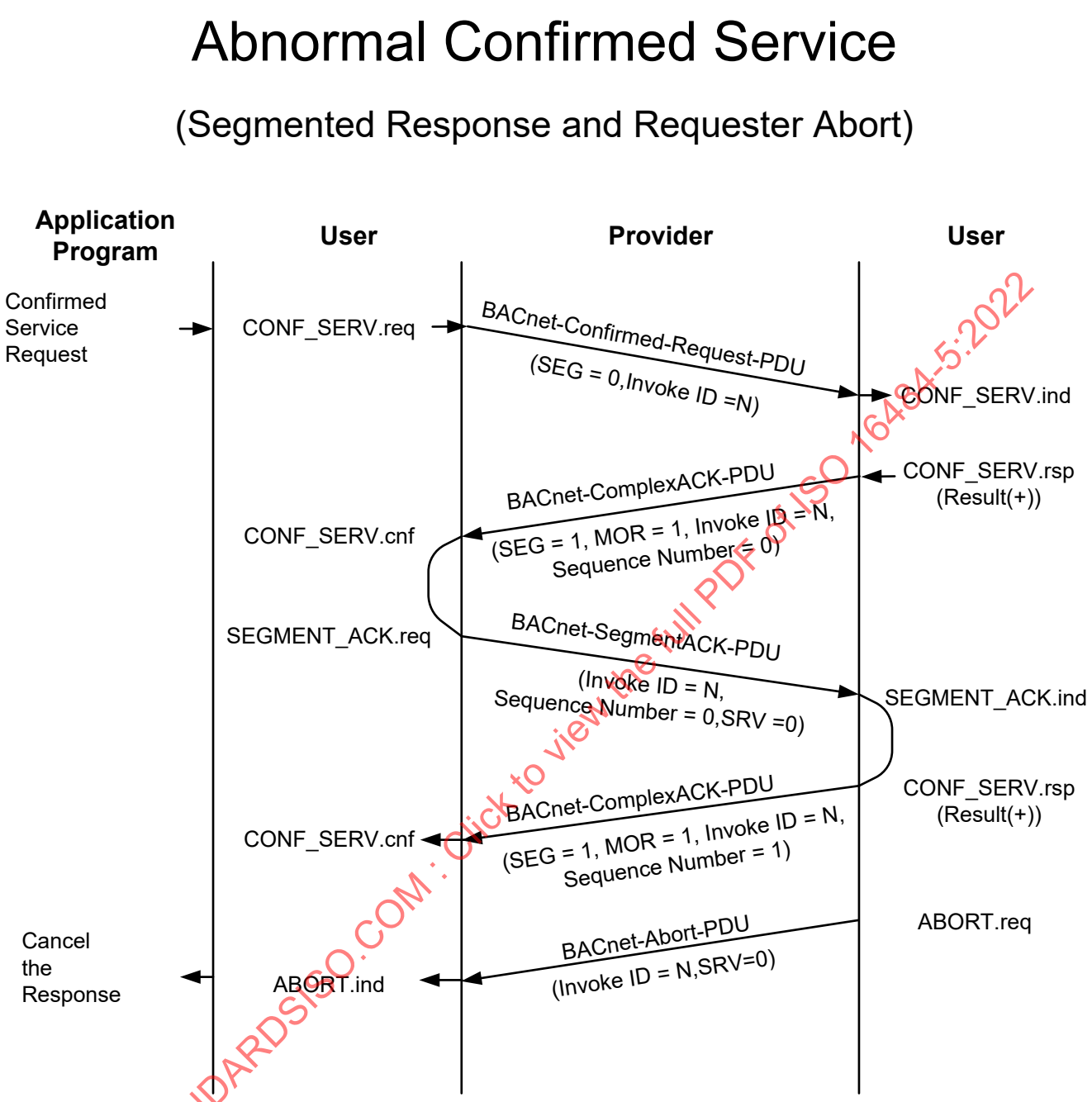


Figure 5-11. Time sequence diagram for an abnormal confirmed service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

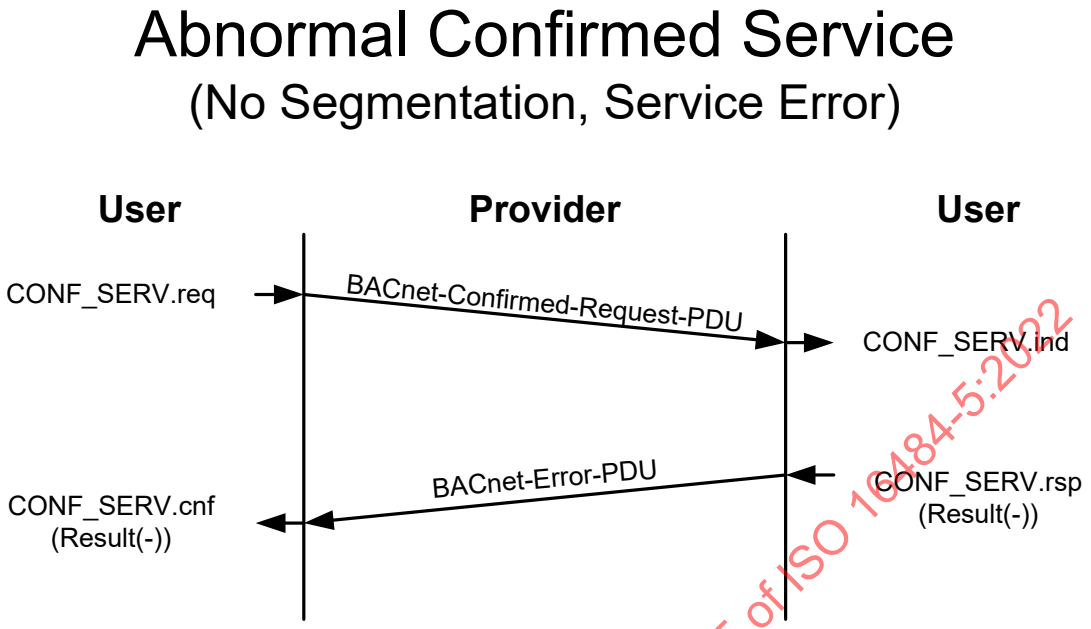


Figure 5-12. Time sequence diagram for an abnormal confirmed service.

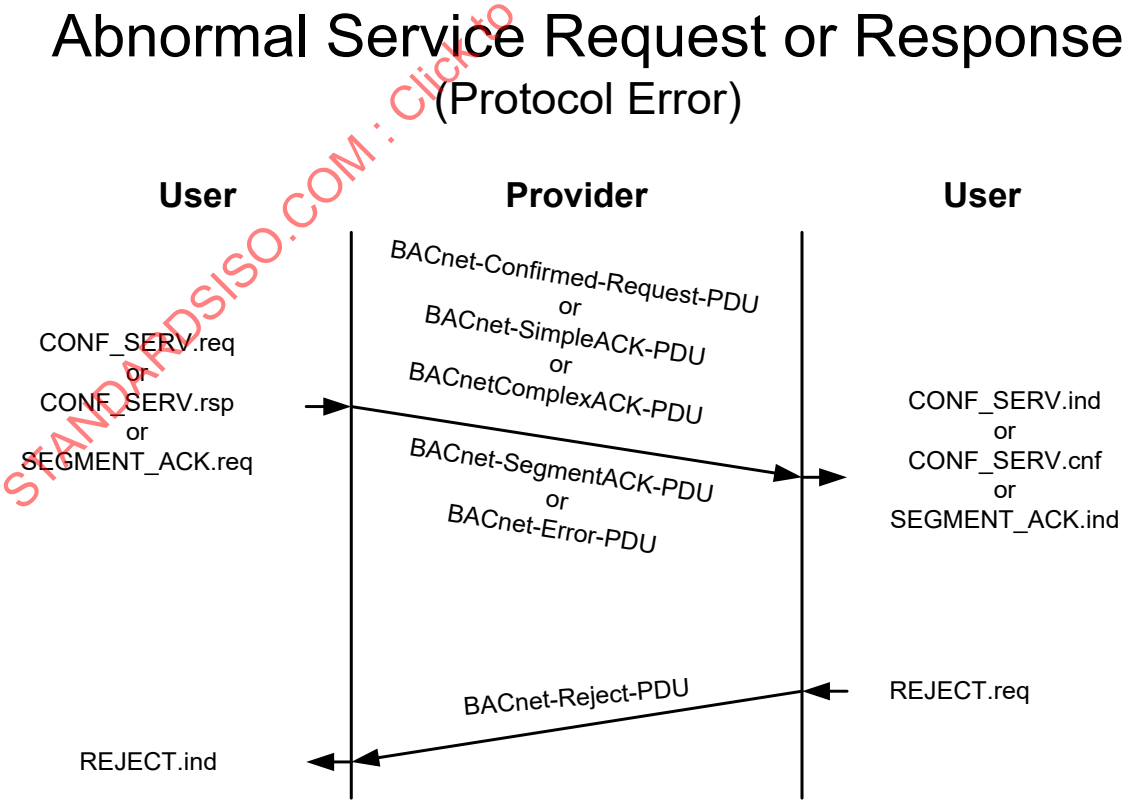


Figure 5-13. Time sequence diagram for an abnormal service request or response with a protocol error.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

5.6 Application Layer Service Conventions

This standard uses the descriptive conventions contained in the OSI technical report on service conventions, ISO TR 8509. The OSI conventions define the interactions between a protocol service user and a protocol service provider. Information passed between the protocol service user and the protocol service provider is represented abstractly as an exchange of "service primitives." The service primitives are an abstraction of the functional specification and the user-layer interaction. The abstract definition does not contain local detail of the user/provider interaction. Each primitive has a set of zero or more parameters, representing data elements that are passed to qualify the functions invoked by the primitive. Parameters indicate information available in a user/provider interaction; in any particular interface, some parameters may be explicitly stated (even though not explicitly defined in the primitive) or implicitly associated with the service access point. Similarly, in any particular protocol specification, functions corresponding to a service primitive may be explicitly defined or implicitly available.

Clauses 13 through 17 and 24 use a tabular format to describe the component parameters of the BACnet service primitives. Each table consists of five columns, containing the name of the service parameter and a column each for the request ("Req"), indication ("Ind"), response ("Rsp"), and confirm ("Cnf") primitives. The "Rsp" and "Cnf" columns are absent for unconfirmed services. Each row of the table contains one parameter or subparameter. Under the appropriate service primitive columns, a code is used to specify the type of use of the parameter on the primitive specified in the vertical column. These codes follow the conventions suggested in the ISO technical report on conventions, ISO TR 8509, namely:

M - parameter is Mandatory for the primitive.
U - parameter is a User option and may not be provided.
C - parameter is Conditional upon other parameters.
S - parameter is a Selection from a collection of two or more possible parameters. The parameters that make up this collection are indicated in the table as follows:

(a) each parameter in the collection is specified with the code "S";
(b) the name of each parameter in the collection is at the same table indentation from the beginning of the parameter column in the table; and
(c) either
    a. each parameter is at the leftmost (outer) indentation in the table or
    b. each parameter is part of the same parameter group. A parameter group is a collection of parameters where each member has a common parent parameter. The parent parameter for any group member is the first parameter above the member that is not indented as far as that member. In the following example, ParameterA and ParameterB form a parameter group:

```
ParameterX
        ParameterA
        ParameterB
ParameterY
        ParameterC
```

Informally, for parameters involved in a selection, the indentation in the service tables signifies which parameters are involved in a selection. All parameters at the same level of indentation under a common "higher level" parameter are part of the same selection.

The code "(=)" following one of the codes M, U, C, or S indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table. For instance, an "M(=)" code in the indication service primitive column and "M" in the request service primitive column means that the parameter in the indication primitive is semantically equivalent to that in the request primitive.

Some parameters may contain subparameters. Subparameters are indicated by indenting them with respect to the parent parameter. The presence of subparameters is always dependent on the presence of the parent parameter. In the example above, ParameterA and ParameterB are subparameters of ParameterX and ParameterC is a subparameter of ParameterY. If ParameterX is optional and is not supplied in a service primitive, then the subparameters (ParameterA and ParameterB) shall not be supplied.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Some service parameters are named using a "List of ..." convention. Unless otherwise noted, all parameters whose name begins with "List of ..." specify a list of zero or more of the item specified after the "List of" keyword phrase.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

6 THE NETWORK LAYER

The purpose of the BACnet network layer is to provide the means by which messages can be relayed from one BACnet network to another, regardless of the BACnet data link technology in use on that network. Whereas the data link layer provides the capability to address messages to a single device or broadcast them to all devices on the local network, the network layer allows messages to be directed to a single remote device, broadcast on a remote network, or broadcast globally to all devices on all networks. A BACnet device is uniquely located by a network number and a MAC address.

Devices that interconnect two disparate BACnet LANs, e.g., Ethernet and ARCNET, and provide the relay function described in this clause are called "BACnet routers." Devices that interconnect two disparate BACnet networks through a point-to-point (PTP) connection (see Clause 10) are also BACnet routers. BACnet routers build and maintain their routing tables automatically using the network layer protocol messages defined in this clause. Network layer protocol messages facilitate both the auto-configuration of routers and the flow of messages to, and between, routers. BACnet routing capability may be implemented in stand-alone devices or, alternatively, in devices that carry out other building automation and control functions.

Some functions assigned to the network layer by the OSI Basic Reference Model are not required in BACnet. One such function involves selecting a communications path between source and destination machines based on an optimization algorithm. This is not required because BACnet internetworks shall be designed and installed with at most a single, active path between any two devices, a constraint that greatly reduces the complexity of the network layer. Another common network layer function is message segmentation and reassembly. To obviate the need for these capabilities at the network layer, BACnet imposes a limitation on the length of the NPDU in messages passed through a BACnet router. The maximum NPDU length shall not exceed the capability of any data link technology encountered along the path from source to destination. A list of the maximum NPDU lengths for BACnet data link technologies and the minimum routed NPDU length between any two data link types supported by a BACnet router is given in Table 6-1.

Table 6-1. NPDU Lengths of BACnet Data Link Layers

Data Link Technology	Maximum NPDU Length on Data Link	Minimum Routed NPDU Length
ARCNET (ATA 878.1), as defined in Clause 8	501 octets	501 octets
BACnet/IP, as defined in Annex J	1497 octets	1497 octets
BACnet/IPv6, as defined in Annex U	1497 octets	1497 octets
BACnet/SC, as defined in Annex AB[1]	61327 octets[2]	1497 octets
Ethernet (ISO 8802-3), as defined in Clause 7	1497 octets	1497 octets
LonTalk (ISO/IEC 14908.1), as defined in Clause 11	228 octets	228 octets
MS/TP, as defined in Clause 9	1497 octets	1497 octets
Point-To-Point, as defined in Clause 10	501 octets	501 octets
ZigBee, as defined in Annex O	501 octets	501 octets

[1] BACnet routing between network ports of this data link technology shall include forwarding additional data attributes along with the NPDU being forwarded. See Clause 6.6.

[2] This number is the result of subtracting the BACnet/SC BVLC message header size (16 octets) and the minimally supported data options size (4192 octets) from the maximum BVLC message size of 65535 octets.

6.1 Network Layer Service Specification

Conceptually, the BACnet network layer provides an unacknowledged connectionless form of data unit transfer service to the application layer. The primitives associated with the interaction are the N-UNITDATA.request, the N-UNITDATA.indication, the N-RELEASE.request, and the N-REPORT.indication. These primitives provide parameters as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

N-UNITDATA.request (
destination_address,
data,
network_priority,
data_expecting_reply,
data_attributes
)

N-UNITDATA.indication (
source_address,
destination_address,
data,
network_priority,
data_expecting_reply,
data_attributes
)

N-RELEASE.request (
destination_address
)

N-REPORT.indication (
peer_address,
error_condition,
error_parameters,
data_attributes
)

The 'destination_address' and 'source_address' parameters provide the logical concatenation of 1) an optional network number, 2) the MAC address appropriate to the underlying LAN technology, and the 3) the link service access point. A network number of X'FFFF' indicates that the message is to be broadcast "globally" to all devices on all currently reachable networks. Currently reachable networks are those networks to which an active connection is already established within the BACnet internet. In particular, a global broadcast shall not trigger any attempts to establish PTP connections. The 'data' parameter is the network service data unit (NSDU) passed down from the application layer and is composed of a fully encoded BACnet APDU. The 'network_priority' is a numeric value used by the network layer in BACnet routers to determine any possible deviations from a first-in-first-out approach to managing the queue of messages awaiting relay. The data_expecting_reply parameter indicates whether (TRUE) or not (FALSE) a reply data unit is expected for the data unit being transferred. The optional parameter 'data_attributes' contains extra information about the 'data' parameter and can include the optional 'security_parameters' information that contains the security information used to secure the request and context information required for security related N-REPORT.indication primitives to be related to application TSMs.

Upon receipt of an N-UNITDATA.request primitive from the application layer, the network layer shall attempt to send an NSDU using the procedures described in this clause. Upon receipt of an NSDU from a peer network entity, a network entity shall either 1) send the NSDU to its destination on a directly connected network, 2) send the NSDU to the next BACnet router en route to its destination, and/or 3) if the destination address matches that of one of its own application entities, issue an N-UNITDATA.indication primitive to the appropriate entity in its own application layer to signal the arrival of the NSDU.

Upon receipt of an N-RELEASE.request primitive from the application layer, the network entity shall attempt to issue a DL-RELEASE.request primitive to the data link entity specified by the destination_address parameter of the N-RELEASE.request. If there is only one data link entity available, then the destination_address may be ignored.

The N-REPORT.indication primitive is used by the local network layer to indicate failures to transmit N-UNITDATA.requests to peer devices. The errors may be locally detected error conditions, or error conditions reported by a peer device via a network layer message. This primitive is used extensively by the network security wrapper to indicate security errors up the stack. The 'peer_address' parameter is of the same form as the 'destination_address' or 'source_address' parameters of the N-UNITDATA primitives and indicates the peer with which the error condition arose. The optional parameter 'data_attributes' contains extra information about the error including the optional

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'security_parameters' portion that conveys information describing the security failure and context required to relate the error to a previous N-UNITDATA.request or N-UNITDATA.indication primitive.

6.2 Network Layer PDU Structure

6.2.1 Protocol Version Number

Each NPDU shall begin with a single octet that indicates the version number of the BACnet protocol, encoded as an 8-bit unsigned integer. The present version number of the BACnet protocol is one (1).

6.2.2 Network Layer Protocol Control Information

The second octet in an NPDU shall be a control octet that indicates the presence or absence of particular NPCI fields. Figure 6-1 shows the order of the NPCI fields in an encoded NPDU. Use of the bits in the control octet is as follows.

Bit 7: 1 indicates that the NSDU conveys a network layer message. Message Type field is present.
0 indicates that the NSDU contains a BACnet APDU. Message Type field is absent.

Bit 6: Reserved. Shall be zero.

Bit 5: Destination specifier where:
0 = DNET, DLEN, DADR, and Hop Count absent
1 = DNET, DLEN, and Hop Count present
DLEN = 0 denotes broadcast MAC DADR and DADR field is absent
DLEN > 0 specifies length of DADR field

Bit 4: Reserved. Shall be zero.

Bit 3: Source specifier where:
0 = SNET, SLEN, and SADR absent
1 = SNET, SLEN, and SADR present
SLEN = 0 Invalid
SLEN > 0 specifies length of SADR field

Bit 2: The value of this bit corresponds to the data_expecting_reply parameter in the N-UNITDATA primitives.
0 = indicates that other than a BACnet-Confirmed-Request-PDU, a segment of a BACnet-ComplexACK-PDU, or a network layer message expecting a reply is present.
1 = indicates that a BACnet-Confirmed-Request-PDU, a segment of a BACnet-ComplexACK-PDU, or a network layer message expecting a reply is present.

Bits 1,0: Network priority where:
B'11' = Life Safety message
B'10' = Critical Equipment message
B'01' = Urgent message
B'00' = Normal message

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

In this standard:

DNET = 2-octet ultimate destination network number.
DLEN = 1-octet length of ultimate destination MAC layer address
(A value of 0 indicates a broadcast on the destination network.)
DADR = Ultimate destination MAC layer address.
DA = Local network destination MAC layer address.
SNET = 2-octet original source network number.
SLEN = 1-octet length of original source MAC layer address.
SADR = Original source MAC layer address.
SA = Local network source MAC layer address.

Version	1 octet
Control	1 octet
DNET	2 octets
DLEN	1 octet
DADR	variable
SNET	2 octets
SLEN	1 octet
SADR	variable
Hop Count	1 octet
Message Type	1 octet
Vendor ID	2 octets
APDU	N octets

Figure 6-1. NPDU field format. Which fields are present is determined by the bits in the control octet.

Figures 6-2(a) - 6-2(e) provide examples of NPDUs containing APDUs for various combinations of addressing information.

Version = X'01'	1 octet
Control = X'04'	1 octet
APDU	N octets

Figure 6-2(a). Example of a typical "Local" BACnet NPDU for which a reply is expected.

Version = X'01'	1 octet
Control = X'24'	1 octet
DNET	2 octets
DLEN = M	1 octet
DADR	M octets
Hop Count	1 octet
APDU	N octets

Figure 6-2(b). Example of a typical "Remote" BACnet NPDU directed to a router. Network Priority is NORMAL and a reply is expected.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Version = X'01'	1 octet
Control = X'29'	1 octet
DNET	2 octets
DLEN	1 octet
DADR	1 octet
SNET	2 octets
SLEN = 6	1 octet
SADR	6 octets
Hop Count	1 octet
APDU	N octets

Figure 6-2(c). Example of a typical BACnet NPDU as passed between routers. Network Priority is URGENT, original MAC Address is 6 octets, and the ultimate Destination MAC Address is 1 octet.

Version = X'01'	1 octet
Control = X'0B'	1 octet
SNET	2 octets
SLEN = 6	1 octet
SADR	6 octets
APDU	N octets

Figure 6-2(d). Example of a typical "Remote" BACnet NPDU as sent from a Router to its ultimate destination on a directly connected network. Network Priority is LIFE SAFETY.

Version =X'01'	1 octet
Control = X'28'	1 octet
DNET = X'FFFF'	2 octets
DLEN = 0	1 octet
SNET	2 octets
SLEN	1 octet
SADR	1 octet
Hop Count	1 octet
APDU	N octets

Figure 6-2(e). Example of a typical Broadcast message of NORMAL Network Priority as broadcast by a Router.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

6.2.2.1 DNET, SNET, and Vendor ID Encoding

The multi-octet fields, DNET, SNET, and Vendor ID, shall be conveyed with the most significant octet first. Allowable network number values for DNET shall be from 1 to 65535 and for SNET from 1 to 65534.

6.2.2.2 DADR and SADR Encoding

The DADR and SADR fields are encoded as shown in Table 6-2, Figure 6-3, and Figure 6-4.

© ISO 2022 – All rights reserved

Table 6-2. BACnet DADR and SADR Encoding Rules Based Upon Data Link Layer Technology

BACnet Data Link Layer	DLEN	SLEN	Encoding Rules
ARCNET, as defined in Clause 8	1	1	Encoded as in their MAC layer representations
BACnet/IP, as defined in Annex J	6	6	Encoded as specified in Clause J.1.2
BACnet/IPv6, as defined in Annex U	3	3	Encoded as specified in Clause H.7.2
BACnet/SC, as defined in Annex AB	6	6	Encoded as specified in Clause H.7.3
Ethernet, as defined in Clause 7	6	6	Encoded as in their MAC layer representations
LonTalk domain wide broadcast	2	2	The encoding for the SADR is shown in Figure 6-3 The encoding for the DADR is shown in Figure 6-4
LonTalk multicast	2	2	
LonTalk unicast	2	2	
LonTalk, unique Neuron_ID	7	2	
MS/TP, as defined in Clause 9	1	1	Encoded as in their MAC layer representations
ZigBee, as defined in Annex O	3	3	Encoded as specified in Clause H.7.2

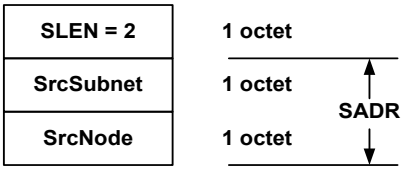


Figure 6-3. Encoding of the SLEN and SADR for NPDUs originating from LonTalk devices being routed through BACnet.

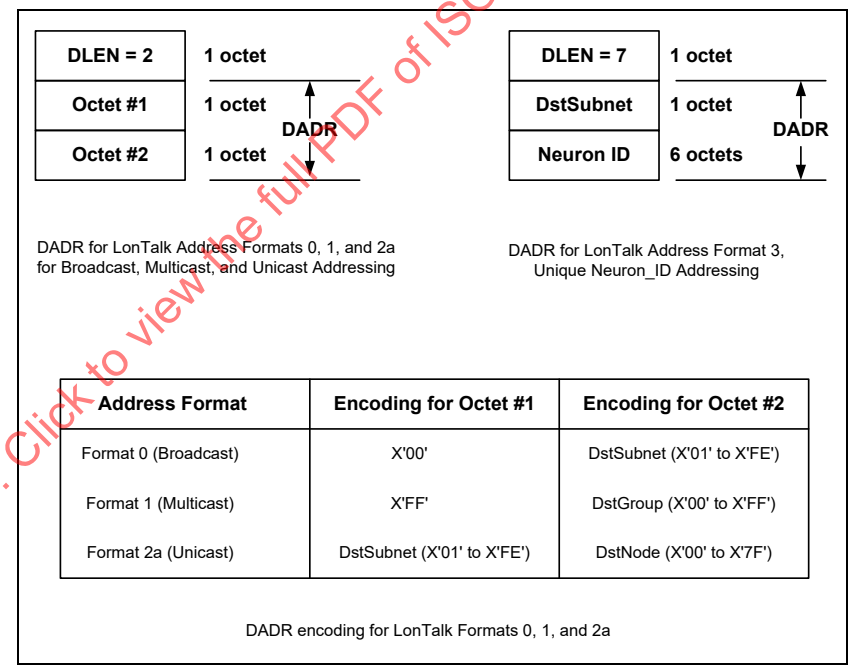


Figure 6-4. Encoding of the DLEN and DADR for NPDUs destined for LonTalk devices being routed through BACnet. The different LonTalk address formats are encoded as shown.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

6.2.3 Hop Count

The Hop Count is a decrementing counter value used to ensure that a message cannot be routed in a circular path indefinitely. Such a circular path can only occur if the configuration rule that allows only a single path between any two BACnet nodes is violated. See Clause 4.2.

The Hop Count field shall be present only if the message is destined for a remote network, i.e. if DNET is present. This is a one-octet field that is initialized to a value of X'FF'. Each router the message passes through shall decrement the Hop Count by at least one but not more than the current value of Hop Count. If the Hop Count reaches a value of zero, the router shall discard the message and not forward it to the next router.

6.2.4 Network Layer Message Type

If Bit 7 of the control octet described in Clause 6.2.2 is 1, a message type octet shall be present as shown in Figure 6-1. The following message types are indicated:

© ISO 2022 – All rights reserved

X'00': Who-Is-Router-To-Network
X'01': I-Am-Router-To-Network
X'02': I-Could-Be-Router-To-Network
X'03': Reject-Message-To-Network
X'04': Router-Busy-To-Network
X'05': Router-Available-To-Network
X'06': Initialize-Routing-Table
X'07': Initialize-Routing-Table-Ack
X'08': Establish-Connection-To-Network
X'09': Disconnect-Connection-To-Network
X'0A': removed
X'0B': removed
X'0C': removed
X'0D': removed
X'0E': removed
X'0F': removed
X'10': removed
X'11': removed
X'12': What-Is-Network-Number
X'13': Network-Number-Is
X'14' to X'7F': Reserved for use by ASHRAE
X'80' to X'FF': Available for vendor proprietary messages

Figures 6-5 through 6-10 provide examples of NPDUs containing network layer messages.

6.2.5 Vendor Proprietary Network Layer Messages

If Bit 7 of the control octet is 1 and the Message Type field contains a value in the range X'80' - X'FF', then a Vendor ID field shall be present as shown in Figure 6-1. Otherwise, the Vendor ID shall be omitted. The Vendor ID is defined in Clause 23. The Vendor ID shall be encoded in two octets.

Version = X'01'	1 octet
Control = X'80'	1 octet
Message Type = X'00'	1 octet
DNET	2 octets (optional)

Figure 6-5. Example of a Who-Is-Router-To-Network message. If DNET is omitted, a router receiving this message shall return a list of all reachable DNETs.

Version = X'01'	1 octet
Control = X'80'	1 octet
Message Type = X'08'	1 octet
DNET	2 octets
Termination Time Value	1 octet

Figure 6-6. Example of an Establish-Connection-To-Network message directed to a local router.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Version = X'01'	1 octet
NPCI = X'80'	1 octet
Message Type = X'03'	1 octet
Rejection Reason	1 octet
DNET	2 octets

Figure 6-7. Example of a Reject-Message-To-Network DNET.

Version = X'01'	1 octet
Control = X'80'	1 octet
Message Type = X'05'	1 octet
List of DNETs	2N octets (optional)

Figure 6-8. Example of a Router-Available-To-Network for one or more DNETs. If the list of DNETs is not present, flow control is being eased on all networks normally reached through the router.

Version = X'01'	1 octet
Control = X'80'	1 octet
Message Type = X'04'	1 octet
List of DNETs	2N octets (optional)

Figure 6-9. Example of a Router-Busy-To-Network for one or more DNETs. If the list of DNETs is not present, flow control is being imposed on all networks normally reachable through the router.

Version = X'01'	1 octet
Control = X'80'	1 octet
Message Type = X'02'	1 octet
DNET	2 octets
Performance Index	1 octet

Figure 6-10. Example of an I-Could-Be-Router-To-Network Message.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

6.2.6 Network Layer Messages Conveying Data

If there are data octets to be conveyed for the message type specified in Clause 6.2.4, these data octets shall follow the message type octet in the manner prescribed for each message type.

6.3 Messages for Multiple Recipients

BACnet supports the transmission of messages to multiple recipients through the use of multicast and broadcast addresses. Multicasting results in a message being processed by a group of recipients. Broadcasting results in a message being processed by all of the BACnet devices on the local network, a remote network, or all networks. The use of broadcast or multicast addressing for network layer protocol messages is described in Clause 6.5. Of the BACnet APDUs, only the BACnet-Unconfirmed-Request-PDU may be transmitted using a multicast or broadcast network layer address (note that a MAC layer multicast or broadcast address may be used for other PDU types when the network layer address restricts the destination to a single device).

6.3.1 Multicast Messages

At present, only Ethernet, LonTalk, ZigBee (as defined in Annex O), and BACnet/IP (as defined in Annex J) support multicast addresses. The method by which a BACnet device is assigned to a specific multicast group shall be a local matter.

© ISO 2022 – All rights reserved

6.3.2 Broadcast Messages

Three forms of broadcast transmission are provided by BACnet: local, remote, and global. A local broadcast is received by all stations on the local network. A remote broadcast is received by all stations on a single remote network. A global broadcast is received by all stations on all networks comprising the BACnet internetwork.

A local broadcast makes use of the broadcast MAC address appropriate to the local network's LAN technology, i.e. X'FFFFFFFFFFFF' for Ethernet, X'00' for ARCNET, X'FF' for MS/TP, or X'00' in the DstSubnet field of Address Format 0 in LonTalk, X'FFFF' for ZigBee, and an IP address with all ones in the host portion for BACnet/IP.

A remote broadcast is made on behalf of the source device on a specific distant network by a router directly connected to that network. In this case, DNET shall specify the network number of the remote network and DLEN shall be set to zero.

A global broadcast, indicated by a DNET of X'FFFF', is sent to all networks through all routers. Upon receipt of a message with the global broadcast DNET network number, a router shall decrement the Hop Count. If the Hop Count is still greater than zero, then the router shall broadcast the message on all directly connected networks except the network of origin, using the broadcast MAC address appropriate for each destination network. If the Hop Count is zero, then the router shall discard the message. In order for the message to be disseminated globally, the originating device shall use a broadcast MAC address on the originating network so that all attached routers may receive the message and propagate it further.

If a router has one or more ports that represent PTP connections as defined in Clause 10, global broadcasts shall be processed as follows. If the PTP connection is currently established, that is, the Connection State Machine is in the Connected state (see Clause 10.4.9), then the global broadcast message shall be transmitted through the PTP connection. If the PTP connection is not currently established, then no action shall be taken by the router to transmit the broadcast message through the PTP connection.

6.4 Network Layer Protocol Messages

This clause describes the format and purpose of the ten BACnet network layer protocol messages. These messages provide the basis for router auto-configuration, router table maintenance, and network layer congestion control.

6.4.1 Who-Is-Router-To-Network

This message is indicated by a Message Type of X'00' optionally followed by a 2-octet network number. Who-Is-Router-To-Network is used by both routing and non-routing nodes to ascertain the next router to a specific destination network or, in the case of routers, as an aid in building an up-to-date routing table. See Figure 6-5.

6.4.2 I-Am-Router-To-Network

This message is indicated by a Message Type of X'01' followed by one or more 2-octet network numbers. It is used to indicate the network numbers of the networks accessible through the router generating the message. It shall always be transmitted with a broadcast MAC address.

6.4.3 I-Could-Be-Router-To-Network

This message is used to respond to a Who-Is-Router-To-Network message containing a specific 2-octet network number when the responding half-router has the capability of establishing a PTP connection that can be used to reach the desired network but this PTP connection is not currently established.

This message is indicated by a Message Type of X'02'. The complete format of the NPDU is shown in Figure 6-10. The 2-octet network number indicates the DNET that could be reached by this half-router. The 1-octet "Performance Index" is a locally determined number that gives an indication of the quality and performance of this proposed connection. A low value in this field indicates a high performance index. Typically, the Performance Index would be established at installation time and set relative to the performance of other PTP half-routers in the system.

6.4.4 Reject-Message-To-Network

This message is indicated by a Message Type of X'03' followed by an octet indicating the reason for the rejection and a 2-octet network number (see Figure 6-7). It is directed to the node that originated the message being rejected, as

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

indicated by the source address information in that message. The rejection reason octet shall contain an unsigned integer with one of the following values:

0: Other error.

1: The router is not directly connected to DNET and cannot find a router to DNET on any directly connected network using Who-Is-Router-To-Network messages.

2: The router is busy and unable to accept messages for the specified DNET at the present time.

3: It is an unknown network layer message type. The DNET returned in this case is a local matter.

4: The message is too long to be routed to this DNET.

5: This rejection reason value has been removed.

6: The source message was rejected due to errors in the addressing. The length of the DADR or SADR was determined to be invalid.

6.4.5 Router-Busy-To-Network

This message is indicated by a Message Type of X'04' optionally followed by a list of 2-octet network numbers. It shall always be transmitted with a broadcast MAC address appropriate to the network on which it is broadcast. Router-Busy-To-Network is used by a router to curtail the receipt of messages for specific DNETs or all DNETs. See Figure 6-9.

6.4.6 Router-Available-To-Network

This message is indicated by a Message Type of X'05' optionally followed by a list of 2-octet network numbers. It shall always be transmitted with a broadcast MAC address. Router-Available-To-Network is used by a router to enable or re-enable the receipt of messages for a specific list of DNETs or all DNETs. See Figure 6-8.

6.4.7 Initialize-Routing-Table

This message is indicated by a Message Type of X'06'. It is used to initialize the routing table of a router or to query the contents of the current routing table.

The format of the data portion of the Initialize-Routing-Table message is shown in Figure 6-11.

The Number of Ports field of this NPDU indicates how many port mappings are being provided in this NPDU. This field permits routing tables to be incrementally updated as the network changes. Valid entries in this field are 0-255. Following this field are sets of data indicating the DNET directly connected to this port or accessible through a dial-up PTP connection, Port ID, Port Info Length, and, in the case Port Info Length is non-zero, Port Info. If an Initialize-Routing-Table message is sent with the Number of Ports equal to zero, the responding device shall return its complete routing table in an Initialize-Routing-Table-Ack message without updating its routing table. If the Port ID field has a value of zero, then all table entries for the specified DNET shall be purged from the table. If the Port ID field has a non-zero value, then the routing information for this DNET shall either replace any previous entry for this DNET in the routing table or, if no such entry exists, be appended to the routing table.

The Port Info Length is an unsigned integer indicating the length of the Port Info field.

The Port Info field, if present, shall contain an octet string. A typical use would be to convey modem control and dial information for accessing a remote network via a dial-up PTP connection.

The Initialize-Routing-Table message shall be transmitted with the DER = TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Number of Ports	1 octet
Connected DNET	2 octets
Port ID	1 octet
Port Info Length	1 octet
Port Info	J octets
. . .	. . .
Connected DNET	2 octets
Port ID	1 octet
Port Info Length	1 octet
Port Info	K octets

Figure 6-11. Format of the Data Portion of an Initialize-Routing-Table or Initialize-Routing-Table-Ack.

6.4.8 Initialize-Routing-Table-Ack

This message is indicated by a Message Type of X'07'. It is used to indicate that the routing table of a router has been changed or the table has been queried through the receipt of an Initialize-Routing-Table message with the Number of Ports field set equal to zero. The data portion of this message, returned only in response to a routing table query, conveys the routing table information, and it has the same format as the data portion of an Initialize-Routing-Table message. See Clause 6.4.7 and Figure 6-11.

6.4.9 Establish-Connection-To-Network

This message is used to instruct a half-router to establish a new PTP connection that creates a path to the indicated network.

This message is indicated by a Message Type of X'08'. The complete format of the NPDU is shown in Figure 6-6. The 2-octet network number indicates the DNET that should be connected to by this half-router. The 1-octet "Termination Time Value" specifies the time, in seconds, that the connection shall remain established in the absence of NPDUs being sent on this connection. A value of 0 indicates that the connection should be considered to be permanent. See Clause 6.7.1.4.

6.4.10 Disconnect-Connection-To-Network

This message is indicated by a Message Type of X'09' followed by a 2-octet network number. This message is used to instruct a router to disconnect an established PTP connection. The disconnection process shall follow the procedures described in Clause 10.

6.4.11 Deleted Clause

This clause has been removed.

6.4.12 Deleted Clause

This clause has been removed.

6.4.13 Deleted Clause

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

6.4.14 Deleted Clause

This clause has been removed.

6.4.15 Deleted Clause

This clause has been removed.

6.4.16 Deleted Clause

This clause has been removed.

6.4.17 Deleted Clause

This clause has been removed.

6.4.18 Deleted Clause

This clause has been removed.

6.4.19 What-Is-Network-Number

This message is indicated by a Message Type of X'12'. It is used to request the local network number from other devices on the local network. This message may be transmitted with a local broadcast or a local unicast address. This message shall never be routed. Devices shall ignore What-Is-Network-Number messages that contain SNET/SADR or DNET/DADR information in the NPCI.

Upon receipt of a What-Is-Network-Number message, a device that knows the local network number shall transmit a local broadcast Network-Number-Is message back to the source device. If the What-Is-Network-Number message was broadcast, then a non-routing device may optionally wait for up to 10 seconds before sending the Network-Number-Is message. If during that time, a different device broadcasts the Network-Number-Is message, the non-routing device may choose to not send a Network-Number-Is message.

Upon receipt of a What-Is-Network-Number message, a device that does not know the local network number shall discard the message.

A device shall cache its local network number, and not repeatedly issue this service.

6.4.20 Network-Number-Is

This message is indicated by a Message Type of X'13' followed by a 2-octet network number (most significant octet first), followed by a 1-octet flag, where a value of 1 indicates that the network number was configured, and a value of 0 indicates that the network number was learned by receipt of a previous Network-Number-Is message. This message is used to indicate the local network number to other devices on the local network. It shall be transmitted with a local broadcast address, and shall never be routed. Devices shall ignore Network-Number-Is messages that contain SNET/SADR or DNET/DADR information in the NPCI or that are sent with a local unicast address.

For a device that has not been configured to know its network number, when it receives this message indicating a configured network number, it shall set its network number from this message. If it receives this message indicating a learned network number, then it shall set its network number only if it has not previously received a message indicating a configured network number. If a device resets, it shall reduce the quality of its last received network number to "learned".

For a device that has been configured to know its network number, when it receives this message indicating a configured network number that is in conflict with its configuration, it should report the conflict to a local or remote management entity.

Upon startup, routers that have been configured to know their network numbers shall broadcast out each port a Network-Number-Is message containing the network number for the port and indicate that this number is configured, not learned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

6.5 Network Layer Procedures

This clause describes the network layer procedures to be followed by BACnet router and non-router nodes for both local and remote data transfer. "Local" means that the source and destination devices are on the same BACnet network. "Remote" means that the source and destination devices are on different BACnet networks. The source and destination networks are interconnected by zero or more intervening networks joined by BACnet routers to form a BACnet internetwork. See Figure 4-3.

If a BACnet device supports a data link that supports the 'data_attributes' parameter, the network layer shall support the forwarding of data attributes with a size of 4192 octets at a minimum. The insertion of 'data_attributes' for network layer messages originating at the local network layer is a local matter.

6.5.1 Network Layer Procedures for the Transmission of Local Traffic

Upon receipt of an N-UNITDATA.request primitive, the network entity (NE) shall inspect the DNET portion of the 'destination_address' parameter. The absence of DNET indicates that the destination device resides on the same BACnet network as the device issuing this transmission request. The value of the 'network_priority' parameter shall be included in the NPCI control octet although its use by receiving non-router entities is unspecified. The NE shall prepare a control NPCI octet indicating the absence of DNET, DADR, HOP COUNT, SNET, and SADR, concatenate it with the 'data' parameter conveyed in the N-UNITDATA.request primitive, and issue a DL-UNITDATA data link request primitive. The concatenation of the NPCI and the NSDU (the 'data' parameter from the N-UNITDATA.request), the NPDU, is passed as the 'data' parameter of the data link primitive. If the N-UNITDATA.request provided the 'data_attributes' parameter, it shall be passed to the data link if the data link supports this parameter.

6.5.2 Network Layer Procedures for the Receipt of Local Traffic

Upon receipt of an NPDU from the data link layer (conveyed by the 'data' parameter of the DL-UNITDATA data link indication primitive) whose first octet indicates BACnet version one, the destination NE shall interpret the second octet of the NPDU as control NPCI. If bit 7 of the control NPCI indicates that the message contains an APDU, then the procedure in Clause 6.5.2.1 is followed. Otherwise, a network layer message is being conveyed and the procedure in Clause 6.5.2.2 applies.

6.5.2.1 Receipt of Local APDUs

If the control NPCI octet indicates the absence of a DNET field or a DNET field is present and contains the global broadcast address X'FFFF', the NE shall attempt to locate a BACnet application entity. If a BACnet application entity is found, the NE shall issue an N-UNITDATA.indication primitive with the portion of the data link data following the NPCI as the 'data' parameter. If the 'data_attributes' parameter was received with the data link indication primitive, it shall be passed to the N-UNITDATA.indication primitive. If the application entity is not found and the NE resides in a non-routing node, the data link data shall be discarded. If the DNET is present and not equal to the global broadcast address X'FFFF' and the NE resides in a non-routing node, the data link data shall likewise be discarded and no further action taken. If the DNET is present and the NE resides in a BACnet router, the NE shall take the actions specified in Clause 6.5.4.

6.5.2.2 Receipt of Local Network Layer Messages

If the control NPCI octet indicates the absence of a DNET field or a DNET field is present and contains the global broadcast address X'FFFF', the NE shall attempt to interpret the network layer message. If the DNET field is absent and the message cannot be interpreted and the message was directed specifically at the router, a Reject-Message-To-Network shall be returned to the device that sent the message.

If the DNET is present and not equal to the global broadcast address X'FFFF' and the NE resides in a non-routing node, the data link data shall be discarded and no further action taken. If the DNET is present and the NE resides in a BACnet router, the NE shall take the actions specified in Clause 6.5.4.

6.5.3 Network Layer Procedures for the Transmission of Remote Traffic

Upon receipt of an N-UNITDATA.request primitive, the NE shall inspect the DNET portion of the 'destination_address' parameter. The presence of a DNET signifies that the destination device resides on a different BACnet network than the device issuing this transmission. A DNET value of X'FFFF' signifies a global broadcast and indicates that the message is to be directed to all local routers via a broadcast message on the local network. The NE shall prepare an NPCI control octet indicating the presence of DNET, DADR, and Hop Count but the absence of SNET and SADR. The NE shall also fill

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

in the network priority field using the supplied parameter. The resulting control, priority, and address information shall then be concatenated with the 'data' parameter conveyed in the N-UNITDATA.request primitive and issued as a DL-UNITDATA data link request primitive. The concatenation of the NPCI and the 'data' parameter from the N-UNITDATA.request (the NSDU), the NPDU, is passed as the 'data' parameter of the data link primitive. If the N-UNITDATA.request provided the 'data_attributes' parameter, it shall be passed to the data link if the data link supports that parameter. The DA portion of the 'destination_address' parameter passed to the data link layer shall be the MAC address of the BACnet router corresponding to the DNET parameter or the appropriate broadcast DA if the address of the router is initially unknown. The broadcast DA is also to be used if the DNET global broadcast network number is present.

Note that five methods exist for establishing the address of a BACnet router for a particular DNET: 1) the address may be established manually at the time a device is configured, 2) the address may be learned by issuing a Who-Is request and noting the SA associated with the subsequent I-Am message (assuming the device specified in the Who-Is is located on a remote DNET and the I-Am message was handled by a router on the local network), 3) by using the network layer message Who-Is-Router-To-Network, 4) by using the local broadcast MAC address in the initial transmission to a device on a remote DNET and noting the SA associated with any subsequent responses from the remote device, and 5) by noting the SA associated with any requests received from the remote DNET. Which method is used shall be a local matter; however, devices shall not rely solely on method 1.

The local broadcast MAC address may be used in response messages, although it is discouraged. It is preferable that a device note the SA associated with the original request and reuse that SA in the response. For MS/TP networks, in order for MS/TP master devices to use the local broadcast MAC address in a response, a Reply Postponed MAC frame shall be sent in response to the BACnet Data Expecting Reply frame and the response may then be sent when the MS/TP master device receives the token. MS/TP slave devices are unable to use the local broadcast MAC address for responses because they never receive the token.

6.5.4 Network Layer Procedures for the Receipt of Remote Traffic

Upon receipt of an NPDU from the data link layer (conveyed by the 'data' parameter of the DL-UNITDATA indication primitive) whose first octet indicates BACnet version one, the NE shall interpret the second octet of the NPDU as control NPCI. If the NPCI control octet indicates the presence of a DNET field whose value is not X'FFFF' and the NE resides in a BACnet device that is not a router, the message shall be discarded. If the NPCI control octet indicates the presence of a DNET field and the NE resides in a BACnet router, it shall place the NPDU and the data attributes received (conveyed by the 'data_attributes' parameter of the DL-UNITDATA.indication primitive), if any, in its message queue (or queues, if separate queues are maintained for each DNET), arranged in order by priority. Within each priority, the messages shall be arranged in first-in-first-out order. If the NPCI control octet indicates that the NPDU contains a network layer message, the NE shall, in addition, inspect the Message Type field. If this field indicates the presence of a Reject-Message-To-Network message, the NE shall carry out the processing specified in Clause 6.6.3.5. If the SNET and SADR fields are present, the message has arrived from a peer router. If the SNET and SADR fields are absent, the message originated on a network directly connected to the router. In the latter case, the router shall add the SNET and SADR to the NPCI based on the router's knowledge of the network number of the network from which the message arrived, alleviating the requirement that the originating station know its own network number. The SADR field shall be set equal to the SA of the incoming NPDU.

If the NPCI control octet indicates the presence of a DNET field, the NE resides in a BACnet router, the NPDU is to be routed to a different device and the NPDU requires a reply (conveyed by the 'data_expecting_reply' parameter of the DL-UNITDATA indication primitive), then a DL-RELEASE request shall be issued to the data link layer entity specified by the source_address value of the DL-UNITDATA indication.

Three possibilities exist: either the router is directly connected to the network referred to by DNET, the message must be relayed to another router for further transmission, or a global broadcast is required. In the first case, DNET, DADR, and Hop Count shall be removed from the NPCI and the message shall be sent directly to the destination device with DA set equal to DADR. The control octet shall be adjusted accordingly to indicate only the presence of SNET and SADR. In the second case, if the Hop Count is greater than zero, the message shall be sent to the next router on the path to the destination network. If the next router is unknown, an attempt shall be made to identify it using a Who-Is-Router-To-Network message. If the Hop Count is zero, then the message shall be discarded. If the DNET global broadcast network number is present and the Hop Count is greater than zero, the router shall broadcast the message on each network to which the router is directly connected, except the network of origin, using the broadcast address appropriate to each data link. If the DNET global broadcast network number is present and the Hop Count is zero, then the message shall be discarded.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If data attributes were received with the NPDU, these data attributes shall be forwarded with the NPDU to the data links through the 'data_attributes' parameter of the DL-UNITDATA.request, if supported. If this parameter is not supported by the DL-UNITDATA.request primitive, the data attributes shall be ignored.

6.6 BACnet Routers

BACnet routers are devices that interconnect two or more BACnet networks to form a BACnet internetwork. BACnet routers shall, at a minimum, implement the device requirements as specified in Clause 22.1.5. Table 6-1 specifies the maximum NPDU length and the minimum NPDU length being routed of the different data link layer types. Routers shall be capable of routing the NPDUs between any two of those data link layers supported by the router based on theminimum routed NPDU length defined in Table 6-1.

BACnet/SC to BACnet/SC BACnet routers, referred to as secure connect BACnet routers, shall support at least two BACnet/SC network ports and may support routing and forwarding of NPDUs exceeding 1497 octets, up to 61327 octets maximum between these ports. Secure connect BACnet routers shall support forwarding of the 'data_attributes' parameter content with the NPDU. Secure connect BACnet routers shall support forwarding of a minimum of 4192 octets of 'data_attributes' content between all BACnet/SC ports. See Clause AB.1.4

BACnet routers make use of BACnet network layer protocol messages to maintain their routing tables. Routers perform the routing tasks described in Clause 6.5. See Figure 6-12 for a flow chart of router operation.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

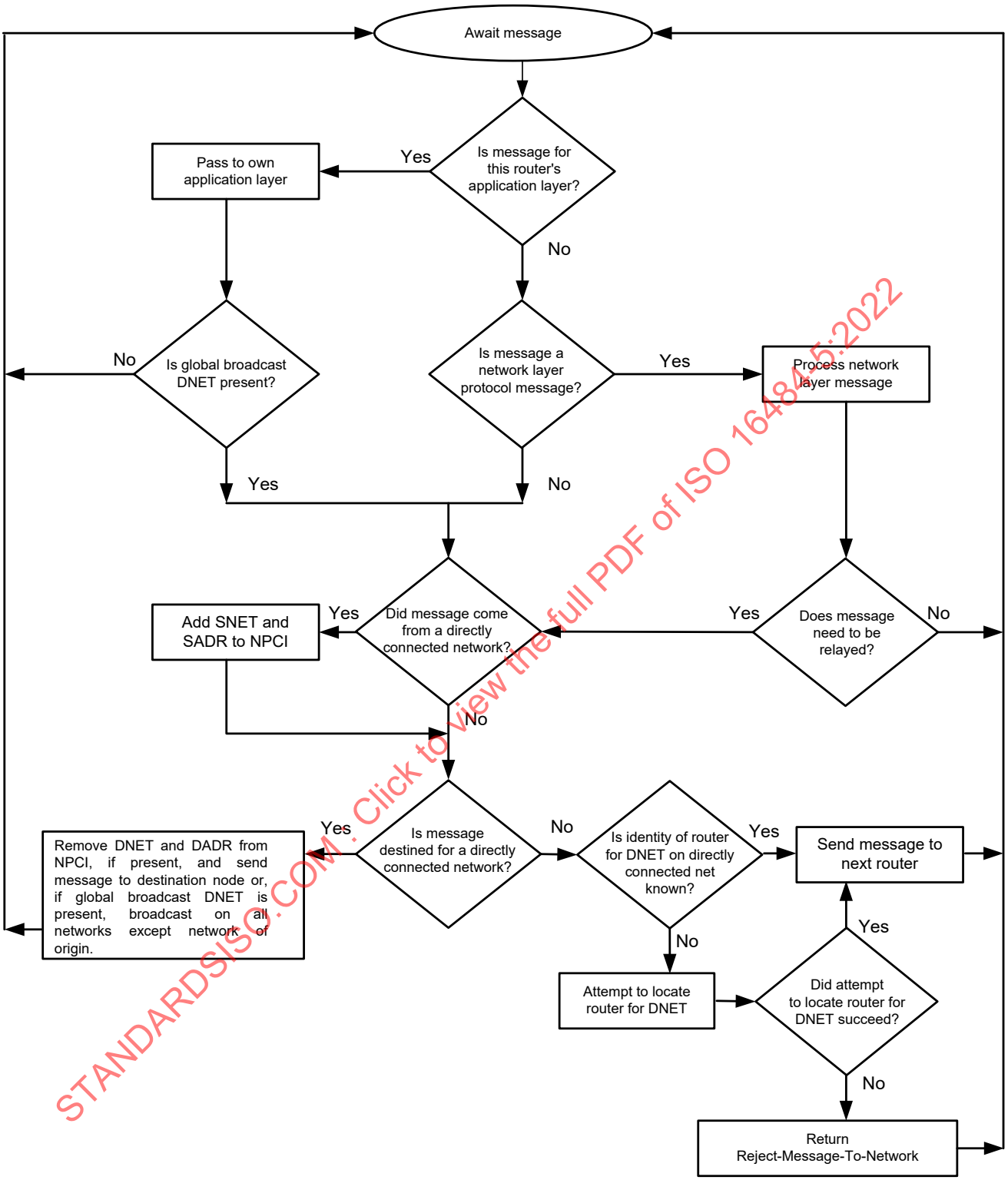


Figure 6-12. BACnet Message Routing.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

6.6.1 Routing Tables

By definition, a router is a device that is connected to at least two BACnet networks. Each attachment is through a "port." A "routing table" consists of the following information for each port:

(a) the MAC address of the port's connection to its network;
(b) the 2-octet network number of the directly connected network;
(c) a list of network numbers reachable through the port along with the MAC address of the next router on the path to each network number and the reachability status of each such network.

© ISO 2022 – All rights reserved

The "reachability status" is an implementation-dependent value that indicates whether the associated network is able to receive traffic. The reachability status shall be able to distinguish, at a minimum, between "permanent" failures of a route, such as might result from the failure of a router, and "temporary" unreachability due to the imposition of a congestion control restriction.

6.6.2 Start-up Procedures

Upon start-up, each router shall broadcast out each port an I-Am-Router-To-Network message containing the network numbers of each accessible network except the networks reachable via the network on which the broadcast is being made. This enables routers to build or update their routing table entries for each of the network numbers contained in the message.

6.6.3 Router Operation

This clause describes the operation of BACnet routers.

6.6.3.1 BACnet NPDUs - General

If a BACnet NPDU is received with NPCI indicating that the message should be relayed by virtue of the presence of a non-broadcast DNET, the router shall search its routing table for the indicated network number. Normal routing procedures are described in Clause 6.5. If, however, the network number cannot be found in the routing table or through the use of the Who-Is-Router-To-Network message, the router shall generate a Reject-Message-To-Network message and send it to the node that originated the BACnet NPDU. If the NPCI indicates either a remote or global broadcast, the message shall be processed as described in Clause 6.3.2.

6.6.3.2 Who-Is-Router-To-Network

This message may be generated by a non-routing BACnet node or by a BACnet router. If the message is broadcast with a specific network number, one I-Am-Router-To-Network message should be returned at most, originating at the router on the local network that is the next router to the specified destination network. If the 2-octet network number is omitted, each responding router shall reply with an I-Am-Router-To-Network message containing all networks reachable through it, including those that may be temporarily unreachable due to the imposition of a congestion control restriction and excluding the networks reachable through the port from which the Who-Is-Router-To-Network message was received. Who-Is-Router-To-Network will generally be broadcast but may be directed to a specific router to learn the contents of its router table. In the event a router receives multiple I-Am-Router-To-Network messages pertaining to the same network, the router shall assume that each new I-Am-Router-To-Network message represents a modification in the system configuration and shall update its routing information. If the router has an established PTP connection (see Clause 10) that conflicts with this new information, the PTP connection shall be terminated using the disconnect procedures defined in Clause 10. Thus the last message received shall take precedence over all previous messages.

When a router receives a Who-Is-Router-To-Network message specifying a particular network number, it shall search its routing table for the network number contained in the message. If the specified network number is found in its table and the port through which it is reachable is not the port from which the Who-Is-Router-To-Network message was received, the router shall construct an I-Am-Router-To-Network message containing the specified network number and send it to the node that generated the request using a broadcast MAC address, thus allowing other nodes on this network to take advantage of the routing information.

If the network number is not found in the routing table, the router shall attempt to discover the next router on the path to the indicated destination network by generating a Who-Is-Router-To-Network message containing the specified destination network number and broadcasting it out all its ports other than the one from which the Who-Is-Router-To-Network message arrived. Two cases are possible. In case one the received Who-Is-Router-To-Network message was from the originating device. For this case, the router shall add SNET and SADR fields before broadcasting the subsequent Who-Is-Router-To-Network. This permits an I-Could-Be-Router-To-Network message to be directed to the originating device. The second case is that the received Who-Is-Router-To-Network message came from another router and it already contains SNET and SADR fields. For this case, the SNET and SADR shall be retained in the newly generated Who-Is-Router-To-Network message.

If the Who-Is-Router-To-Network message does not specify a particular destination network number, the router shall construct an I-Am-Router-To-Network message containing a list of all the networks it is able to reach through other than the port from which the Who-Is-Router-To-Network message was received and transmit it in the same manner as described above. The message shall list all networks not flagged as permanently unreachable, including those that are

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

temporarily unreachable due to the imposition of congestion control restrictions. Networks that may be reachable through a PTP connection shall be listed only if the connection is currently established.

6.6.3.3 I-Am-Router-To-Network

At router start-up, each router shall broadcast locally an I-Am-Router-To-Network message on each directly connected network as specified in Clause 6.6.2. Each such message shall list each accessible network number except the number of the network on which the broadcast is being made. This broadcast allows other routers to update their routing tables whenever a new router joins the internetwork. In addition, an I-Am-Router-To-Network message shall be broadcast locally upon the receipt of a Who-Is-Router-To-Network message containing a network number matching a network number contained in the router's routing table, provided that the port through which it is reachable is not the port from which the Who-Is-Router-To-Network message was received.

If one or more of the reachable networks listed in the I-Am-Router-To-Network message is reached through a directly connected PTP connection, transmitting the I-Am-Router-To-Network message shall start or restart a connection termination delay timer. The PTP connection shall not be terminated before this delay timer expires. The connection termination delay timer shall be configurable with a default value of sixty seconds.

Upon receipt of an I-Am-Router-To-Network message, the router shall search its routing table for entries corresponding to each network number contained in the message. If no entry is found for a particular network number, a new entry shall be created. If an entry is found but the MAC address or port of the next router on the path to the indicated network differs from that found in the table, the MAC address in the table shall be replaced with that of the router originating the I-Am-Router-To-Network message. This ensures that all routers will have the most current information in their tables. Whether the router table was updated or not, the router shall then generate an I-Am-Router-To-Network message for all the network numbers contained in the received I-Am-Router-To-Network message and broadcast the new message, using the local broadcast MAC address, out all ports other than the one from which the previous message was received.

6.6.3.4 I-Could-Be-Router-To-Network

This message is generated by a half-router in response to a Who-Is-Router-To-Network message containing a specific 2-octet network number when the responding half-router has the capability of establishing a PTP connection that can be used to reach the desired network but this PTP connection is not currently established. In the event that a Who-Is-Router-To-Network message is received in which the 2-octet network number field is absent, such as is used to determine lists of networks reachable through active routers, the I-Could-Be-Router-To-Network message shall not be returned. The I-Could-Be-Router-To-Network message shall be directed to the device that originated the Who-Is-Router-To-Network message. The procedures to be used to establish a PTP connection are described in Clause 6.7 and Clause 10.

6.6.3.5 Reject-Message-To-Network

Reject-Message-To-Network is generated by a router when it receives a message that it is unable to relay to the DNET specified in the NPCI or if it receives an unknown network layer message directed specifically to that router. The reasons for rejecting the message are set forth in Clause 6.4.4.

When a router receives a Reject-Message-To-Network message with a rejection reason octet containing a value of 1 or 2, it shall search its routing table for the network number specified in the Reject-Message-To-Network message. If the network number is found, the status information for this network number shall be updated to indicate that the network is permanently unreachable if the reject reason was 1 or unreachable due to flow control if the reject reason was 2. In addition, regardless of the contents of the rejection reason octet, the router shall relay the message in the normal manner to the originating node specified in the NPCI using the procedures of Clause 6.5. A rejection reason of 1 is to be considered a serious error condition and should be reported to a local or remote network management entity. The nature of this reporting procedure is a local matter.

6.6.3.6 Router-Busy-To-Network

If a router wishes to curtail the receipt of messages for specific DNETs or all DNETs, it shall generate a Router-Busy-To-Network message.

If a router temporarily wishes to receive no more traffic for one or more specific DNETs, it shall broadcast a Router-Busy-To-Network message with a list of the 2-octet network numbers corresponding to these DNETs. If the 2-octet network numbers are omitted, it means the router wishes to stop the flow of messages to all the networks it normally serves.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each router receiving a Router-Busy-To-Network message shall update its routing table to indicate that the specified DNETs are not reachable, set or reset a 30-second timer for this status, and broadcast a Router-Busy-To-Network message out each port other than the one on which it was received so that all routers may learn of the congestion control restriction. The congestion control indication shall be cleared upon expiration of the 30-second timer. Upon receiving a message whose destination is on one of the temporarily unreachable DNETs, a router shall send a Reject-Message-To-Network message with a reject reason of 2 to the originating node.

Normally, a Router-Busy-To-Network message should be followed in a short time by a Router-Available-To-Network message indicating that the congestion control restriction has been lifted. In the event that a router receives a message while it is still requiring congestion control and the router is able to accept the message, it shall do so and, at its discretion, again broadcast a Router-Busy-To-Network message for the benefit of this node and any others that may not have received the previous transmission. If the router is unable to accept the message, it shall immediately return a Reject-Message-To-Network to the sender. It may then also broadcast another Router-Busy-To-Network message for the reasons cited above.

6.6.3.7 Router-Available-To-Network

When a router wishes to re-enable the receipt of messages for a specific list of DNETs, or all DNETs, previously curtailed by a Router-Busy-To-Network message, it shall broadcast a Router-Available-To-Network message. If the message is broadcast with a list of 2-octet network numbers, it means that the router is now able to receive traffic for these specific DNETs. If the 2-octet network numbers are omitted, the router wishes to re-enable the flow of messages to all the networks it serves.

Each router receiving a Router-Available-To-Network message shall update its routing table to indicate that the specified DNETs are now reachable and broadcast a Router-Available-To-Network message out each port other than the one on which it arrived so that all routers may learn of the lifting of the congestion control restriction.

6.6.3.8 Initialize-Routing-Table

The Initialize-Routing-Table message is generated by any node that has been programmed to provide the initial routing table information to one or more BACnet routers or wishes to query the contents of the current routing tables. The establishment of the contents of the routing table and the circumstances under which Initialize-Routing-Table messages are generated are local matters. In addition, an Initialize-Routing-Table message with Number of Ports set equal to zero shall cause the responding device to return its complete routing table in an Initialize-Routing-Table-Ack message without updating its routing table.

When a router receives this message containing a routing table, indicated by a non-zero value in the Number of Ports field, it shall update its current port-to-network-number mappings for each network specified in the NPDU with the information contained in the NPDU and return an Initialize-Routing-Table-Ack message without any routing table data to the source. When a router receives this message in the form of a routing table query, indicated by a zero value in the Number of Ports field, it shall return an Initialize-Routing-Table-Ack message to the source containing a complete copy of its routing table as described in Clause 6.6.3.9.

6.6.3.9 Initialize-Routing-Table-Ack

This message is sent by a router after the reception and servicing of an Initialize-Routing-Table message. If the router is acknowledging a table update message, signified by a non-zero value in the Number of Ports field, it shall return an Initialize-Routing-Table-Ack without data. If the router is acknowledging a table query message, indicated by a zero value in the Number of Ports field, it shall return a complete copy of its routing table. If a complete copy of the table cannot be returned in a single acknowledgment, the router shall send multiple acknowledgments, each containing a portion of the routing table until the entire table has been sent.

6.6.3.10 Establish-Connection-To-Network

Upon receipt of an Establish-Connection-To-Network message, a half-router shall attempt to establish a PTP connection using the procedures described in Clause 6.7 and Clause 10.

6.6.3.11 Disconnect-Connection-To-Network

Upon receipt of a Disconnect-Connection-To-Network message, a half-router shall terminate an established PTP connection using the procedures described in Clause 6.7 and Clause 10.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

6.6.4 Router Congestion Control

Routers may wish to temporarily suspend the receipt of messages destined for a specific network or, possibly, all networks. Normally, this would be the result of impending buffer overflow in the router itself but could also occur because of a buffer problem with a downstream router on the path to a particular network. The messages used to impose and remove congestion control restrictions are Router-Busy-To-Network and Router-Available-To-Network. The algorithm for determining that congestion control should be imposed or removed is not specified in this standard but would most likely involve such factors as the percentage of buffer space currently occupied and, possibly, the rate at which new messages have been arriving at the router.

6.7 Point-To-Point Half-Routers

In BACnet networks that are interconnected across PTP connections (as defined in Clause 10), the procedures for half-router establishment and synchronization are different from those for normal routers. This is due to two unique characteristics of this type of connection. First, since a PTP connection may be established over a wide area network, such as the public telephone network, it is sometimes advantageous to limit the duration of these connections. This causes temporary half-router connections that must be controlled by BACnet. Secondly, PTP connections are always established between two half-routers that together form a single router. A diagram of this router architecture is shown in Figure 6-13. When a connection is established, both half-routers also need to update their routing tables to reflect any new or updated routing information stored by the partner half-router.

To control the link establishment, link termination, and route-learning functions of a PTP half-router, BACnet has defined five network layer messages. The I-Could-Be-Router-To-Network message announces that a half-router has the capability to connect to a requested network but does not have an active connection. The Establish-Connection-To-Network message requests that a connection be established. The Disconnect-Connection-To-Network message requests that an active connection be disconnected. Routing table initialization may be performed using the Initialize-Routing-Table and Initialize-Routing-Table-ACK messages. Thereafter, the half-router maintains its table using the same procedures as other active routers regardless of whether any active PTP connections exist.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

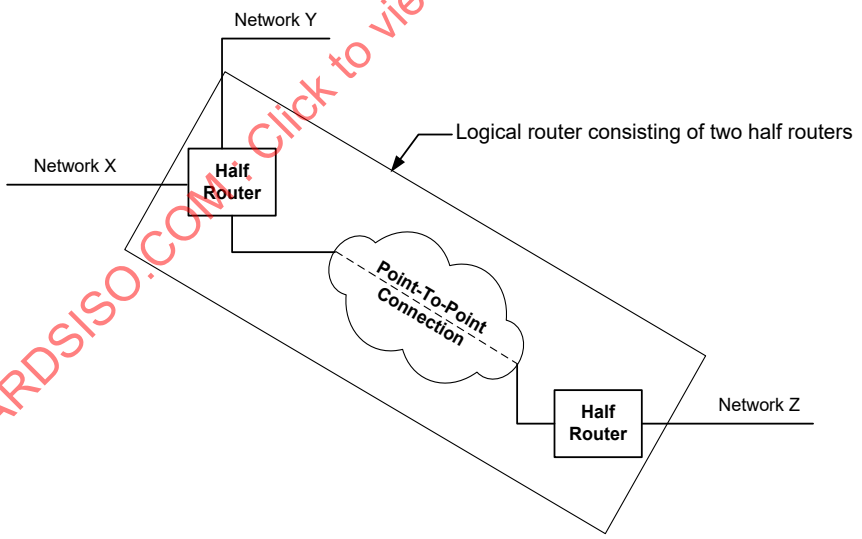


Figure 6-13. Upon a PTP connection, two half-routers combine to become a router.

6.7.1 Procedures for Establishing a New PTP Connection Between Two Half-Routers

As specified in Clause 6.5.3, one of the methods of establishing the address of a BACnet router for a particular DNET is by having the initiating Network Entity (NE) send a Who-Is-Router-To-Network message for the DNET. A router that has an active connection to the DNET, either directly connected or over an established PTP connection, shall respond with an I-Am-Router-To-Network message. A half-router that does not have an active connection but could initiate a PTP connection to the requested DNET shall respond with an I-Could-Be-Router-To-Network message.

© ISO 2022 – All rights reserved

6.7.1.1 Initiating Network Entity (NE) Procedure

To determine a new route, the NE shall issue a Who-Is-Router-To-Network message for the unknown DNET. After a locally specified time period, the initiating NE shall determine the most suitable half-router to the DNET. The algorithm for selecting the most suitable half-router is a local matter. As an aid to help select the most suitable half-router, each I-Could-Be-Router-To-Network message has a 1-octet field to indicate the expected performance of the half router's PTP connection. Upon selection of a suitable half-router, the initiating NE shall send an Establish-Connection-To-Network message to this half-router and wait to receive an I-Am-Router-To-Network message for this DNET. When the I-Am-Router-To-Network message is received, the initiating NE may send NPDUs for this DNET. If, after a locally specified time period, an I-Am-Router-To-Network message is not received by the initiating NE, the initiating NE shall send a Disconnect-Connection-To-Network to the selected half-router and may try this procedure again to find another half-router.

6.7.1.2 Initiating Half-Router Procedure

Upon receipt of an Establish-Connection-To-Network message, a half-router shall try to establish the requested connection. If the connection is established, the initiating half-router shall forward the Establish-Connection-To-Network message to the answering half-router, synchronize its routing table with the routing table of the answering half-router partner using the procedures in Clause 6.7.3, broadcast an I-Am-Router-To-Network message containing all of the DNETs accessible through the answering half-router to all directly connected networks, and start an activity timer (T_{active}).

When the connection is established, the initiating half-router shall adjust its routing table to indicate that any DNETs accessible from the answering half-router have a "reachability status" that is "reachable" and continue with other normal operations.

If the connection cannot be established, the initiating half-router shall adjust its routing table to indicate that any DNETs accessible from the answering half-router have a "reachability status" that is "temporarily unreachable" and continue with other normal operations. If a Disconnect-Connection-To-Network message is received, the half-router shall ignore the message.

If the connection is in the process of being established and a Disconnect-Connection-To-Network message is received, the half-router shall immediately end the connection establishment procedure.

If the connection is in the process of being established and a Who-Is-Router-To-Network message is received for a DNET accessible through the PTP connection, the half-router shall respond with an I-Am-Router-To-Network, followed by a Router-Busy-To-Network message. If a message is received for the DNET before the connection is established, it shall be rejected with a rejection reason = 2. When the connection is established, the initiating half-router shall issue a Router-Available-To-Network message. If the connection cannot be established, the half-router shall issue a Router-Available-To-Network message but reject, with a rejection reason = 1, any message received for the DNET. The reason for rejecting the message is to expedite error handling.

If the connection is in the process of being established and a I-Am-Router-To-Network message is received for the DNET to which the initiating half-router is attempting a connection, the connection establishment shall be immediately terminated.

If the connection is in the process of being established and an Establish-Connection-To-Network message to the same DNET is received, the Termination Time Value shall be evaluated. If the new Termination Time Value is greater than the Termination Time Value of the original Establish-Connection-To-Network message, the new Termination Time Value shall be used by the Activity Timer. The connection process shall then proceed normally.

6.7.1.3 Answering Half-Router Procedure

Upon connection establishment from a PTP half-router, the answering half-router shall set an activity timer (T_{active}) based upon a received Establish-Connection-To-Network message from the initiating half-router, synchronize its routing table with the routing table of the initiating half-router partner using the procedures in Clause 6.7.3, and broadcast an I-Am-Router-To-Network message containing all of the DNETs accessible through the initiating half-router to all directly connected networks. If the connection is terminated, the answering half-router's routing table shall be adjusted to indicate that any DNET accessible from the initiating half-router has a "reachability status" that is "temporarily unreachable," if the answering half-router is able to re-establish the connection or "permanently unreachable" if the answering half-router is unable to re-establish the connection. If the connection is established, the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

answering half-router's routing table shall be adjusted to indicate that any DNET accessible from the initiating half-router has a "reachability status" that is "reachable."

6.7.1.4 Activity Timer (T_{active})

The activity timer (T_{active}) is the time that a half-router shall wait for the absence of any messages being routed over its PTP connection before it attempts to automatically disconnect the connection. This timer shall be set to the Termination Time Value field from the Establish-Connection-To-Network message. If the Termination Time Value is set to zero, the activity time shall be considered infinite.

6.7.1.4.1 Initiating Half-Router Procedure

Upon receipt of an Establish-Connection-To-Network message, an initiating half-router shall set the activity timer (T_{active}) to the Termination Time Value field from the Establish-Connection-To-Network message. If the Termination Time Value is set to zero, the activity time shall be considered infinite.

6.7.1.4.2 Answering Half-Router Procedure

Upon receipt of an Establish-Connection-To-Network message from the initiating half-router, the answering half-router shall set the activity timer (T_{active}) to the Termination Time Value field from the Establish-Connection-To-Network message. If the Termination Time Value is set to zero, the activity time shall be considered infinite.

6.7.2 Procedures for Disconnecting a PTP Connection in a Half-Router

There are three bases for disconnecting a PTP connection established by a half-router. The first is by a Network Entity (NE) initiating a Disconnect-Connection-To-Network message. The second is by a timer expiration indicating that the connection has been inactive for an abnormal period of time. The third basis for disconnecting a connection is to compensate for a configuration error. The specification of this procedure is given in Clause 6.7.4.

6.7.2.1 Active Disconnection of a PTP Connection

6.7.2.1.1 Initiating Network Entity (NE) Procedure

If the initiating NE determines that the half-router connection is no longer needed, it may send a Disconnect-Connection-To-Network message to the half-router. The routing table entry for this DNET shall be immediately set to "disconnected."

6.7.2.1.2 Initiating/Answering Half-Router Procedure

Upon receipt of a Disconnect-Connection-To-Network message, a half-router shall disconnect the PTP connection as specified in Clause 10. When the connection is terminated, the half-router shall adjust its routing table to indicate that any DNETs accessible from the previously connected half-router have a "reachability status" that is "temporarily unreachable" if the half-router is able to re-establish the connection, or "permanently unreachable" if the half-router is unable to re-establish the connection.

6.7.2.2 Timed Disconnection of a PTP Connection

If the activity timer (T_{active}) expires, a half-router shall disconnect the PTP connection as specified in Clause 10. When the connection is terminated, the half-router shall adjust its routing table to indicate that any DNETs accessible from the previously connected half-router have a "reachability status" that is "temporarily unreachable" if the half-router is able to re-establish the connection, or "permanently unreachable" if the half-router is unable to re-establish the connection.

6.7.2.3 Restarting of the Activity Timer (T_{active})

The Activity Timer (T_{active}) in each half-router shall be restarted to its original value contained in the initiating Establish-Connection-To-Network whenever an NPDU is transferred over the PTP link.

6.7.3 Procedures for Synchronizing Half-Router Routing Tables

Upon the establishment of a PTP connection between two half-routers, the routing tables of the half-routers shall be synchronized. This is accomplished using the I-Am-Router-To-Network message.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Upon connection establishment, the two half-routers shall exchange I-Am-Router-To-Network messages. Each message shall contain all of the reachable (before the connection was established) DNETs connected through this router. In the event that a duplicate network connection is discovered by the procedure specified in Clause 6.7.4.2, synchronization of routing tables shall fail, causing the routing table entry for any DNET accessible from the peer half-router to have a reachability status of "temporarily unreachable."

6.7.4 Error Recovery Procedures

6.7.4.1 Recovering from Routing Requests to Unconnected Networks

Since PTP connections may be temporary in nature, there is a possibility that a half-router may receive a message bound for a DNET connection that has been disconnected. If another route is not in place through a different port than the one from which the message was received, this is considered an error. To recover from this situation, the receiving half-router shall reject this message with a Reject-Message-To-Network message using rejection reason = 1. The initiating Network Entity shall recover from this error by initiating the procedure for establishing a new PTP connection through a half-router as described in Clause 6.7.1.

6.7.4.1.1 Disconnected Half-Router Procedure

Upon receipt of a message that is requested to be routed across a PTP connection that is disconnected, the half-router shall determine if another route is in place. If no other route is in place or if the next hop of this route is identical to the path from which the message was received, the half-router shall issue a Reject-Message-To-Network for this message with a rejection reason = 1 and discard the message. If another acceptable route is in place, the message shall be forwarded on this route.

6.7.4.1.2 Initiating Network Entity (NE) Procedure

If the initiating NE receives a Reject-Message-To-Network, it shall attempt to determine a new route to the DNET after waiting for a random back-off period. The random back-off period, in seconds, is determined by the initiating NE through the generation of a random number of either 0 or 1 and then multiplying this number by 40. The initiating NE shall not try to re-establish the network connection until the back-off period has expired. If during the back-off period the initiating NE learns of a valid route to the required DNET, the initiating NE shall use this path and consider the network connection re-established. Upon expiration of the back-off period, if the network connection has not been re-established, the initiating NE shall attempt to determine a new route to the DNET using the procedure for establishing a new PTP connection through a half-router as described in Clause 6.7.1.

6.7.4.2 Recovering from Duplicate Network Connections

In the unlikely event that two or more PTP connections are made to single DNET, at least one of the connections shall be terminated and the routing tables in all routers shall be made consistent. The procedure to ensure that no loop exists consists of having every half-router examine each received I-Am-Router-To-Network message for another path to any of the half-router's directly connected networks. The existence of a second path to a directly connected network indicates that a loop is formed. If a loop is detected, the half-router shall disconnect its PTP connection thereby breaking the loop.

6.7.4.2.1 Half-Router Procedure for Receipt of Conflicting I-Am-Router-To-Network Messages

If during the initialization or lifetime of a PTP connection a half-router hears an I-Am-Router-To-Network message from the PTP connection containing a DNET to one of the half-router's directly connected networks, the half-router shall immediately terminate the connection.

6.7.4.2.2 Half-Router Procedure for Initiation of I-Am-Router-To-Network Messages

As an added safety measure to ensure that duplicate paths are discovered in a timely manner, a half-router shall broadcast one or more I-Am-Router-To-Network message(s) once every five minutes when a PTP connection is in place. The DNETs in this message shall be all of the DNETs accessible through the PTP connection. In the event this list of DNETs would exceed the maximum NPDU length of the network being utilized, the list shall be divided into segments that fit on the network and sent in consecutive I-Am-Router-To-Network messages.

6.7.4.2.3 Half-Router Procedure for Decrementing the Hop Count

To reduce the number of circularly routed messages in a misconfigured system, BACnet NPDUs contain a hop count that limits the number of routers that shall forward the NPDU. In routers that provide the capability to configure the amount that the Hop Count field shall be decremented when an NPDU is forwarded, a network administrator may optimize the damping of looping messages. One method to do this is to find the path in the network that requires the maximum

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

number of router hops. The amount to decrement the NPDU Hop Count field in every router on the network is then calculated as the integer division of 255/(maximum number of router hops). On a PTP connection, the half of the router that forwards the NPDU onto a non-PTP network shall decrement the Hop Count field.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

7 DATA LINK/PHYSICAL LAYERS: Ethernet (ISO 8802-3) LAN

This clause describes the transport of BACnet LSDUs using the services of the data link and physical mechanisms described in International Standards ISO 8802-2: "Information processing systems- Local area networks- Part 2: Logical link control" and ISO/IEC 8802-3: "Information processing systems- Local area networks- Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications." The clauses of ISO 8802-2 pertaining to Class I LLC and Type 1 Unacknowledged Connectionless-Mode Service as well as all of ISO/IEC 8802-3, as amended and extended by the International Organization for Standardization, are deemed to be included in this standard by reference.

7.1 The Use of ISO 8802-2 Logical Link Control (LLC)

Standard BACnet networks may pass BACnet link service data units (LSDUs) using the data link services of ISO 8802-2 Logical Link Control (LLC). A BACnet LSDU consists of an NPDU constructed as described in Clause 6. BACnet devices using ISO 8802-3 LAN technology shall conform to the requirements of LLC Class I, subject to the constraints specified in this clause. Class I LLC consists of Type 1 LLC - Unacknowledged Connectionless-Mode service. LLC parameters shall be conveyed using the DL-UNITDATA primitives as described in the referenced standards.

All BACnet devices conforming to this section shall be capable of accepting properly formed Unnumbered Information (UI) commands and responding to XID Exchange Identification and TEST commands.

7.2 Parameters Required by the LLC Primitives

The DL-UNITDATA primitive requires source address, destination address, data, and priority parameters. The source and destination addresses each consist of the logical concatenation of a medium access control (MAC) address and a link service access point (LSAP). The MAC address is a 6-octet value determined by the network interface hardware. The LSAP is the single-octet value X'82' and is used to indicate that an LSDU contains BACnet data. The data parameter is the NPDU from the network layer. Since the ISO 8802-3 MAC layer only operates at a single priority with only one class of service, the value of the priority parameter is not specified in this standard.

7.3 Parameters Required by the MAC Primitives

The ISO/IEC 8802-3 MAC layer primitives are the MA-DATA.request and MA-DATA.indication. These convey the encoded LLC data using the source and destination MAC addresses described above. Again, since only one class of service is provided, the value of the 'service_class' parameter is unspecified. See Figure 7-1.

7.4 Physical Media

The physical media specified by ISO 8802-3 and subsequent addenda are equally acceptable.

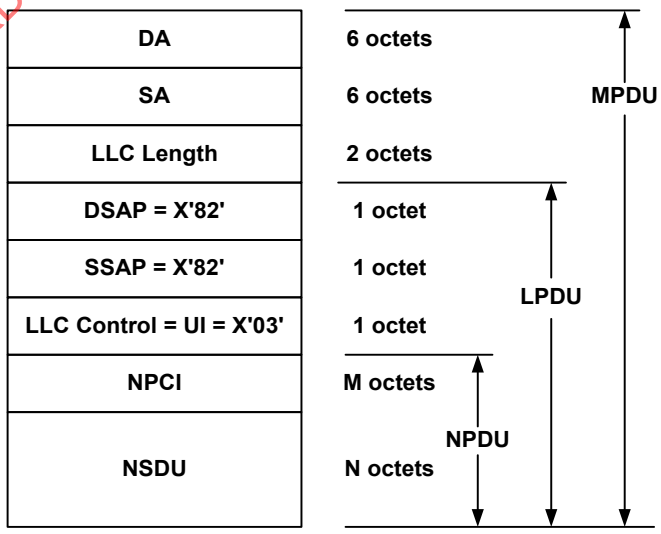


Figure 7-1. Format of an MPDU on an ISO 8802-3 LAN.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

8 DATA LINK/PHYSICAL LAYERS: ARCNET (ATA 878.1) LAN

This clause describes the transport of BACnet LSDUs using the services of the data link and physical mechanisms described in ATA 878.1, "ARCNET Local Area Network Standard." The ARCNET Local Area Network Standard, as amended and extended by the ARCNET Trade Association, is deemed to be included in this standard by reference.

8.1 The Use of ISO 8802-2 Logical Link Control (LLC)

Standard BACnet networks may pass BACnet link service data units (LSDUs) using the data link services of ISO 8802-2 LLC. A BACnet LSDU consists of an NPDU constructed as described in Clause 6. BACnet devices using ARCNET LAN technology shall conform to the requirements of LLC Class I, subject to the constraints specified in this clause. Class I LLC service consists of Type 1 LLC - Unacknowledged Connectionless-Mode service. LLC parameters shall be conveyed using the DL-UNITDATA primitives as described in the referenced standards.

The mapping of these primitives onto the ARCNET MAC layer primitives is described in Clause 8.3..

All BACnet devices conforming to this section shall be capable of accepting and responding to XID Exchange Identification and TEST commands.

8.2 Parameters Required by the LLC Primitives

The DL-UNITDATA primitive requires source address, destination address, data, and priority parameters. The source and destination addresses each consist of the logical concatenation of a medium access control (MAC) address, link service access point (LSAP), and a system code (SC). The MAC address is a 1-octet value determined by the network interface hardware; the LSAP used to indicate that an LSDU contains BACnet data is the single octet value X'82'; and the SC used to indicate a BACnet frame is the single-octet value X'CD'. The data parameter is the NPDU from the network layer. Since the ARCNET MAC sublayer only operates at a single priority with only one class of service, the value of the priority parameter is not specified in this standard.

BACnet ARCNET devices shall support a settable MAC address and shall be able to be set to any valid unicast MAC address. Where a device has multiple ARCNET ports, each port shall be settable to any valid value regardless of the MAC address settings of the other ARCNET ports.

8.3 Mapping the LLC Services to the ARCNET MAC Layer

The Type 1 Unacknowledged Connectionless LLC service shall map directly onto the ARCNET MA_DATA request primitive. Although a successful transmission results in an acknowledgment from the destination MAC sublayer, no indication is expected, or provided, to the LLC sublayer.

ARCNET does not permit MSDUs of length 253, 254, or 255 octets. A BACnet LPDU of length 0 to 252 octets shall be conveyed as the entire MSDU of an ARCNET MPDU (frame) with a single Information length (IL) octet. A BACnet LPDU of length 253 to 504 octets shall be conveyed as the initial octets of the MSDU of an ARCNET MPDU with two Information length (IL) octets. In this case, the LPDU shall be followed by three octets of unspecified value, such that the net length of the MSDU is 256 to 507 octets. When an ARCNET MPDU with two Information length octets is received, the final 3 octets of the MSDU shall be ignored.

An LPDU longer than 504 octets cannot be conveyed via ARCNET.

8.4 Parameters Required by the MAC Primitives

The ARCNET MAC layer primitives are MA-DATA.request, MA-DATA.indication, and MA-DATA.confirmation. These convey the encoded LLC data (MSDU) using the source and destination MAC addresses described above in conjunction with the BACnet system code. See Figure 8-1.

8.5 Physical Media

The physical media specified by the ARCNET standard are equally acceptable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

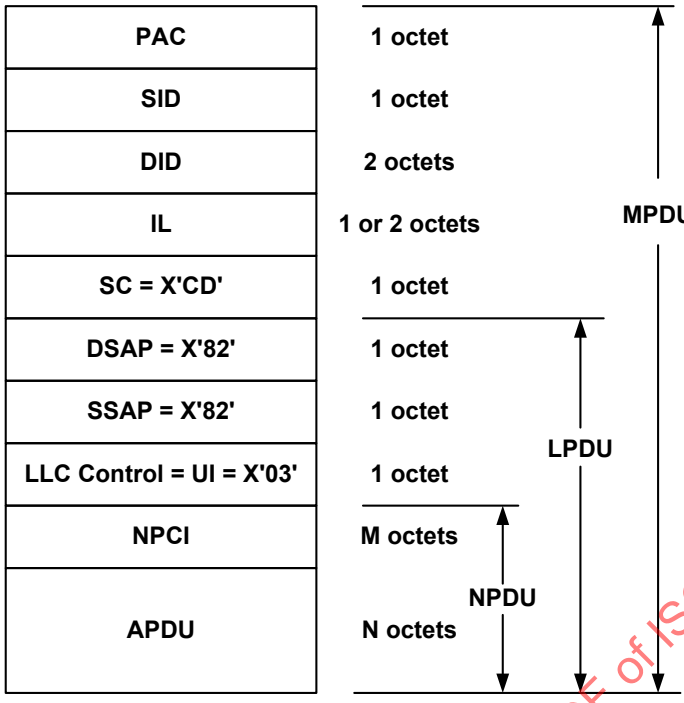


Figure 8-1. Format of an MPDU on an ARCNET LAN.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9 DATA LINK/PHYSICAL LAYERS: MASTER-SLAVE/TOKEN-PASSING (MS/TP) LAN

This clause describes a Master-Slave/Token-Passing (MS/TP) data link protocol, which provides the same services to the network layer as ISO 8802-2 Logical Link Control. It uses services provided by the EIA-485 physical layer. Relevant clauses of EIA-485 are deemed to be included in this standard by reference. The following hardware is assumed:

(a) A UART (Universal Asynchronous Receiver/Transmitter) capable of transmitting and receiving eight data bits with one stop bit and no parity.
(b) An EIA-485 transceiver whose driver may be disabled.
(c) A timer with a resolution of five milliseconds or less.

9.1 Service Specification

MS/TP is not intended to be a general purpose LAN under ISO 8802-2. Instead, MS/TP includes a data link layer sufficient to provide to the BACnet network layer the same services as are offered by ISO 8802-2 Type 1.

This clause describes the primitives and parameters associated with the provided services. The parameters are described in an abstract sense, which does not constrain the implementation method. Primitives and their parameters are described in a form that echoes their specification in ISO 8802-2. This is intended to provide a consistent interface to the BACnet network layer.

9.1.1 DL-UNITDATA.request

9.1.1.1 Function

This primitive is the service request primitive for the unacknowledged connectionless-mode data transfer service.

9.1.1.2 Semantics of the Service Primitive

The primitive shall provide parameters as follows:

```
DL-UNITDATA.request (
        source_address,
        destination_address,
        data,
        priority,
        data_expecting_reply
        )
```

Each source and destination address consists of the logical concatenation of a medium access control (MAC) address and a link service access point (LSAP). For the case of MS/TP devices, since the data link interface supports only the BACnet network layer, the LSAP is omitted and these parameters consist of only the device MAC address.

The 'data' parameter specifies the link service data unit (LSDU) to be transferred by the MS/TP entity. There is sufficient information associated with the LSDU for the MS/TP entity to determine the length of the data unit.

The 'priority' parameter specifies the priority desired for the data unit transfer. The priority parameter is ignored by MS/TP.

The 'data_expecting_reply' parameter specifies whether or not the data unit to be transferred expects a reply.

9.1.1.3 When Generated

This primitive is passed from the network layer to the MS/TP entity to request that a network protocol data unit (NPDU) be sent to one or more remote LSAPs using unacknowledged connectionless-mode procedures.

9.1.1.4 Effect on Receipt

Receipt of this primitive causes the MS/TP entity to attempt to send the NPDU using unacknowledged connectionless-mode procedures. BACnet NPDUs less than 502 octets in length are conveyed using BACnet Data Expecting Reply or BACnet Data Not Expecting Reply frames (see Clause 9.3). BACnet NPDUs between 502 and 1497 octets in length,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

inclusive, are conveyed using BACnet Extended Data Expecting Reply or BACnet Extended Data Not Expecting Reply frames.

9.1.2 DL-UNITDATA.indication

9.1.2.1 Function

This primitive is the service indication primitive for the unacknowledged connectionless-mode data transfer service.

9.1.2.2 Semantics of the Service Primitive

DL-UNITDATA.indication (
source_address,
destination_address,
data,
priority,
data_expecting_reply
)

Each source and destination address consists of the logical concatenation of a medium access control (MAC) address and a link service access point (LSAP). For the case of MS/TP devices, since the data link interface supports only the BACnet network layer, the LSAP is omitted and these parameters consist of only the device MAC address.

The 'data' parameter specifies the link service data unit that has been received by the MS/TP entity. There is sufficient information associated with the LSDU for the MS/TP entity to determine the length of the data unit. A data unit larger than the maximum supported NPDU shall be silently dropped.

The 'priority' parameter specifies the priority desired for the data unit transfer. The priority parameter is ignored by MS/TP.

The 'data_expecting_reply' parameter specifies whether or not the data unit that has been received expects a reply.

9.1.2.3 When Generated

This primitive is passed from the MS/TP entity to the network layer to indicate the arrival of an NPDU from the specified remote entity.

9.1.2.4 Effect on Receipt

The effect of receipt of this primitive by the network layer is unspecified.

9.1.3 Test_Request and Test_Response

ISO 8802-2 Type 1 defines XID and TEST PDUs and procedures but does not define an interface to invoke them from the network layer. Test_Request and Test_Response PDUs and procedures have been defined for MS/TP to accomplish the same functions. Because MS/TP supports only the equivalent of a single LSAP, these PDUs are sufficient to implement the relevant aspects of XID as well.

The response with Test_Response to a received Test_Request PDU is mandatory for all MS/TP nodes. The origination of a Test_Request PDU is optional.

9.1.3.1 Use of Test_Request and Test_Response for ISO 8802-2 TEST Functions

The TEST function provides a facility to conduct loopback tests of the MS/TP to MS/TP transmission path. Successful completion of the test consists of sending a Test_Request PDU with a particular information field to the designated destination and receiving, in return, the identical information field in a Test_Response PDU.

If a receiving node can successfully receive and return the information field, it shall do so. If it cannot receive and return the entire information field but can detect the reception of a valid Test_Request frame (for example, by computing the CRC on octets as they are received), then the receiving node shall discard the information field and return a Test_Response containing no information field. If the receiving node cannot detect the valid reception of frames with overlength information fields, then no response shall be returned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.1.3.2 Use of Test_Request and Test_Response for ISO 8802-2 XID Functions

ISO 8802-2 describes seven possible uses of XID:

(a) XID can be used with a null DSAP and null SSAP as an "Are You There" test. Since MS/TP supports only the equivalent of a single LSAP, the Test_Request PDU with no data can perform this function.
(b) XID can be used with a group or global DSAP to identify group members or all active stations. Since MS/TP supports only the equivalent of a single LSAP, the Test_Request PDU with no data can perform this function.
(c) XID can be used for a duplicate address check. This function is not applicable to MS/TP. EIA-485 token bus networks such as MS/TP will generally not achieve reliable operation if multiple nodes exist with the same address, since collisions will occur during token passing.
(d) Class II LLCs may use XID to determine window size. MS/TP does not support Class II operation.
(e) XID may be used to identify the class of each LLC. Since MS/TP supports only Class I operation, this is a trivial operation.
(f) XID may be used to identify the service types supported by each LSAP. Since MS/TP supports only Class I operation, this is a trivial operation.
(g) An LLC can announce its presence by broadcasting an XID with global DSAP. Since MS/TP supports only one LSAP, the equivalent may be accomplished by broadcasting a Test_Response PDU.

9.1.4 DL-RELEASE.request

9.1.4.1 Function

This primitive is the service request primitive for releasing a Master Node State Machine from the ANSWER_DATA_REQUEST state when no reply is available from the higher layers.

9.1.4.2 Semantics of the Service Primitive

The primitive shall not provide any parameters as follows:

DL-RELEASE.request( )

9.1.4.3 When Generated

This primitive is generated from the network layer to the MS/TP entity to indicate that no reply is available from the higher layers.

9.1.4.4 Effect on Receipt

Receipt of this primitive causes the MS/TP Master Node State Machine to leave the ANSWER_DATA_REQUEST state. If a Master Node State Machine is not in the ANSWER_DATA_REQUEST state or a Slave Node State Machine is present, then this primitive shall be ignored.

9.2 Physical Layer

9.2.1 Medium

An MS/TP EIA-485 network shall use shielded, twisted-pair cable for data signaling with characteristic impedance between 100 and 130 ohms. Distributed capacitance between conductors shall be less than 100 pF per meter (30 pF per foot). Distributed capacitance between conductors and shield shall be less than 200 pF per meter (60 pF per foot). Foil or braided shields are acceptable. The maximum recommended length of an MS/TP segment with AWG 18 (0.82 mm^2 conductor area) cable is specified in Clause 9.2.3. The use of greater distances and/or different wire gauges shall comply with the electrical specifications of EIA-485.

9.2.2 Connections and Terminations

The maximum number of unit loads per segment shall be 32 (as specified by the EIA-485 standard). Additional unit loads may be accommodated by the use of repeaters, as described in Clause 9.9.

Because MS/TP uses NRZ encoding, the polarity of the connection to the cable is important. The non-inverting input of the EIA-485 transceiver is designated in this specification as "plus" or "+" and the inverting input as "minus" or "-". It is

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

recommended, but not required, that the black or red insulated wire of the twisted pair be designated as "plus" and the white, clear, or green insulated wire be designated as "minus." The method of connection between the interface and the cable is not part of this specification.

An MS/TP EIA-485 network shall have no T connections. A termination resistance of 120 ohms plus or minus 5% shall be connected at each of the two ends of the segment medium. No other termination resistors are allowed at intermediate nodes.

Each MS/TP segment shall be provided with network bias resistors, connected as shown in Figure 9-1, such that an undriven communications line will be held in a guaranteed logical one state. The bias provides a reliable way for stations to detect the presence or absence of signals on the line. An unbiased line will take an indeterminate state in the absence of any driving node. Under some conditions, noise or cross-talk might result in some nodes receiving spurious octets from the undriven idle line.

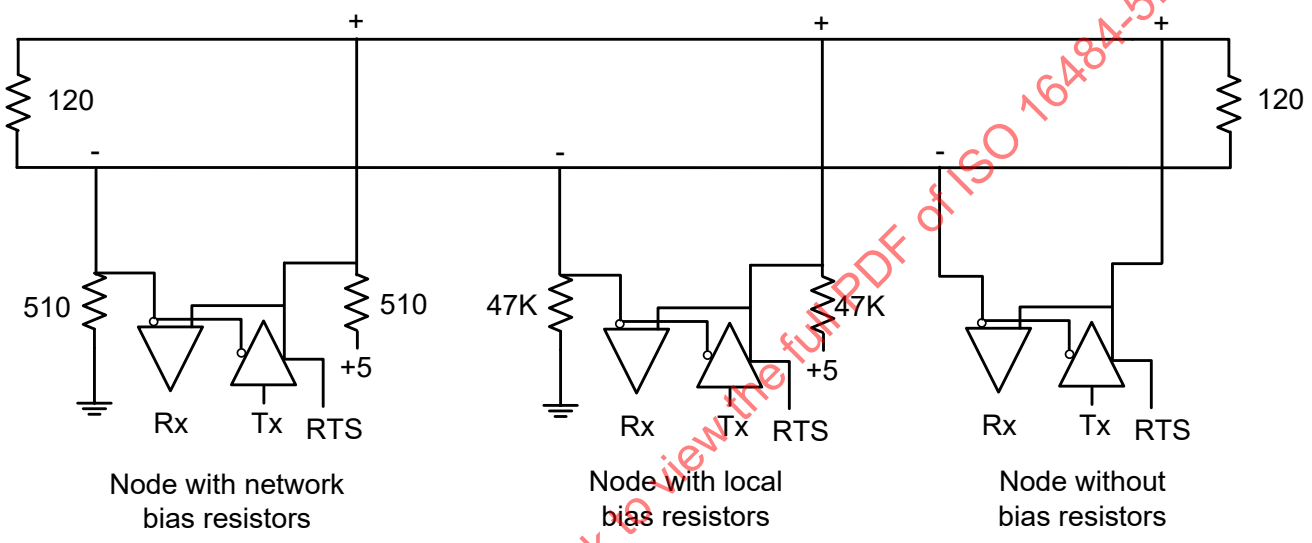


Figure 9-1. EIA-485 Network Showing Three Types of Nodes.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

At least one set, and no more than two sets, of network bias resistors shall exist for each segment. Each set of network bias resistors shall consist of two resistors, each having a value of 510 ohms, plus or minus 5%, connected as shown in Figure 9-1. If two sets of network bias resistors are provided, they shall be placed at two distinct nodes, preferably at the ends of the segment, so that proper bias levels can be maintained even if one of the bias nodes loses power. Other nodes may be provided with local bias resistors as long as each local bias resistor value is 47K ohms or greater. The use of local bias resistors is optional.

For any physical segment that runs between buildings there shall be at least 1500 V of electrical isolation between the EIA-485 signal conductors and the digital ground of any node on that physical segment.

The shield shall be grounded at one end only to prevent ground currents from being created.

9.2.2.1 Device Wiring

There are a variety of permitted device wiring arrangements, depending on the particular needs of the devices used and the installation requirements. Some MS/TP devices are designed with a third-wire Reference connection in addition to the signaling connections and some are two-wire only, depending on the particular application being addressed. All device wiring arrangements shall meet the Connections and Terminations restrictions and requirements described in Clause 9.2.2.

9.2.2.1.1 Single Buildings

Within a single building, there is generally a limited ground voltage offset from one MS/TP device to another, thus permitting a simple installation in most cases. The following clauses describe several common methods for wiring devices within a single building using different device wiring arrangements.

© ISO 2022 – All rights reserved

9.2.2.1.1.1 Twisted-pair Only with Non-isolated Devices

For many installations, a simple twisted pair wire with shield is sufficient to allow reliable communications. In Figure 9-1.1, all of the devices use two-wire connections with the reference level between devices established by an internal earth ground connection made through some impedance (Z) at each device. This is generally the lowest cost solution and is sufficient for installations where electrical noise, ground noise, and stray fields are low. EIA-485 is designed to operate with voltages on the signaling wires between -7 and +12 volts. If the voltage between any two earth ground connections combined with the noise picked up by the twisted pair signaling wire is well within this range, the EIA-485 requirements for signaling levels have been met.

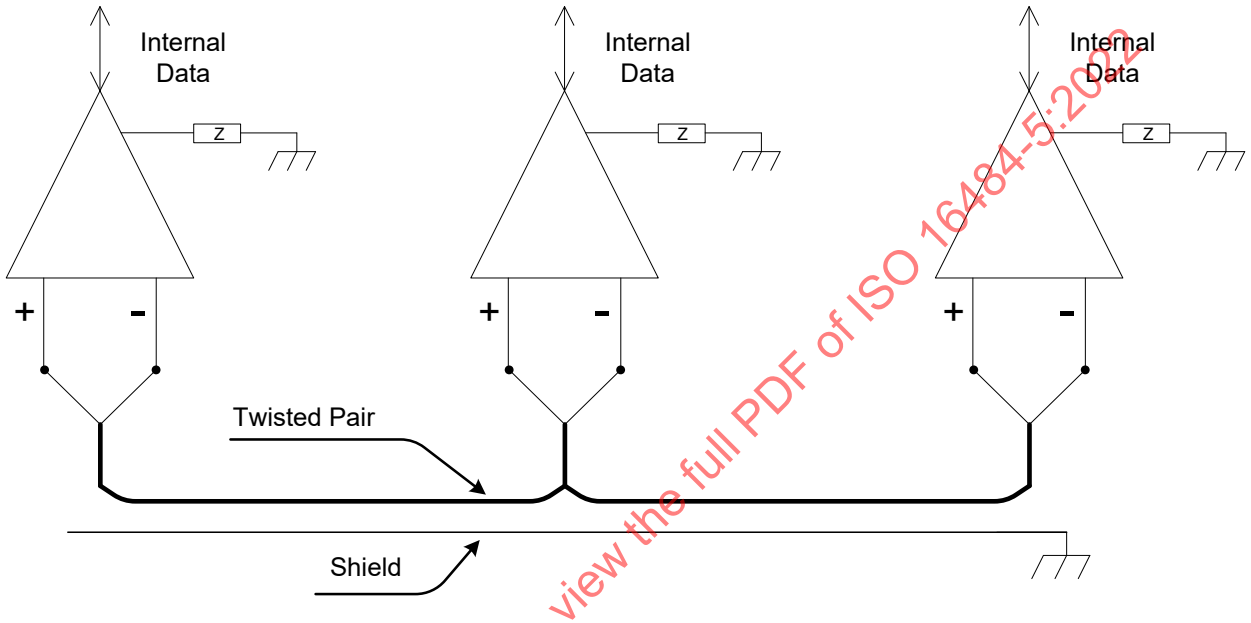


Figure 9-1.1. Simple Twisted Pair with Shield.

9.2.2.1.1.2 Twisted-pair Only with Mixed Devices

Some EIA-485 devices provide a third-wire reference connection and use internal isolation. The third-wire reference shall be electrically connected to the other devices' reference to meet EIA-485 requirements. When such a device is used in a low-electrical-noise installation, it is sufficient to connect the third-wire reference to earth using a 100 ohm current limiting resistor (R) as shown in Figure 9-1.2. The earth connection may be made using either the shield of the communication cable since it is tied to earth ground at one point (this is the preferred approach) or a local earth grounding point such as the device case. This type of connection does not take full advantage of the electrical noise rejection capability of the third-wire reference.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

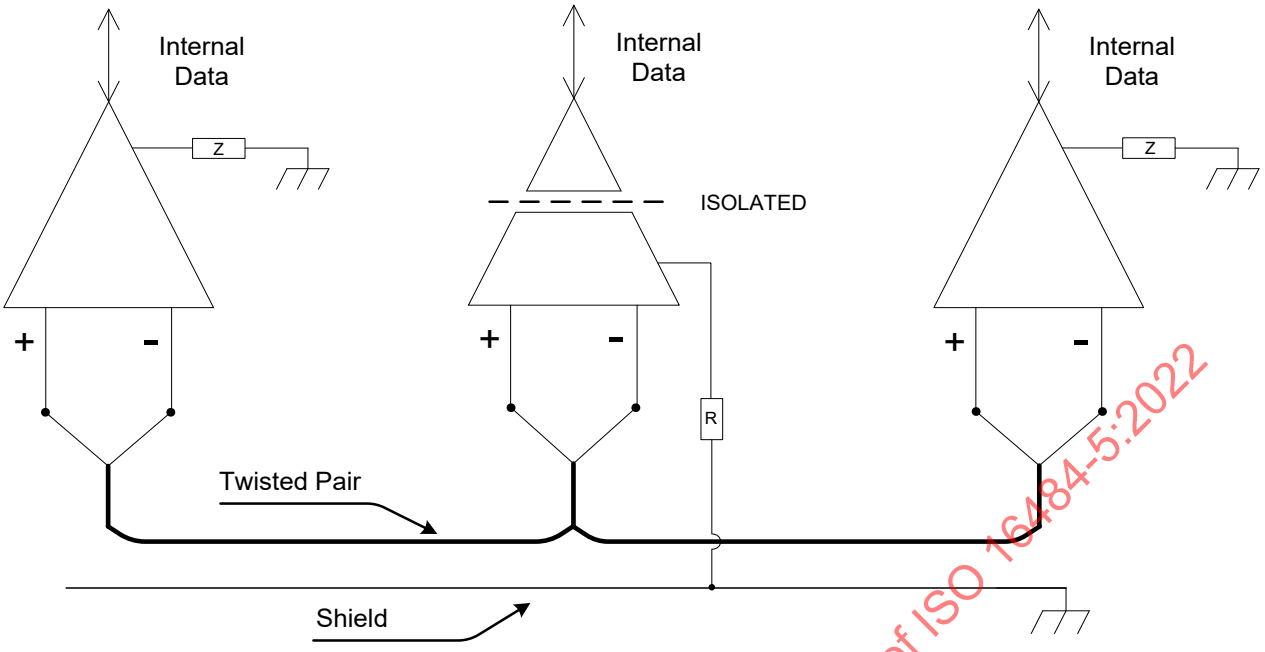


Figure 9-1.2. Mixed Devices on Twisted Pair with Shield.

9.2.2.1.1.3 Twisted-pair and Reference with Isolated Devices

If the installation exclusively uses EIA-485 devices with third-wire reference connections, electrical noise rejection is best if a third conductor in the same cable is used to connect all of the reference connections together as shown in Figure 9-1.3. This nearly eliminates earth-ground voltage differences and allows the differential input of each EIA-485 device to float with the electrical noise and stray fields picked up by the signal cable, resulting in better noise rejection. If there are more than three wires in the cable chosen, the third conductor shall be made up of all of the extra wires (outside of the twisted pair used for signaling) connected together. If desired, the third-wire reference conductor may be tied to earth ground at one point where electrical noise is low through a 100-ohm current-limiting resistor in order to limit voltage excursions and to simplify adding two-wire devices in the future.

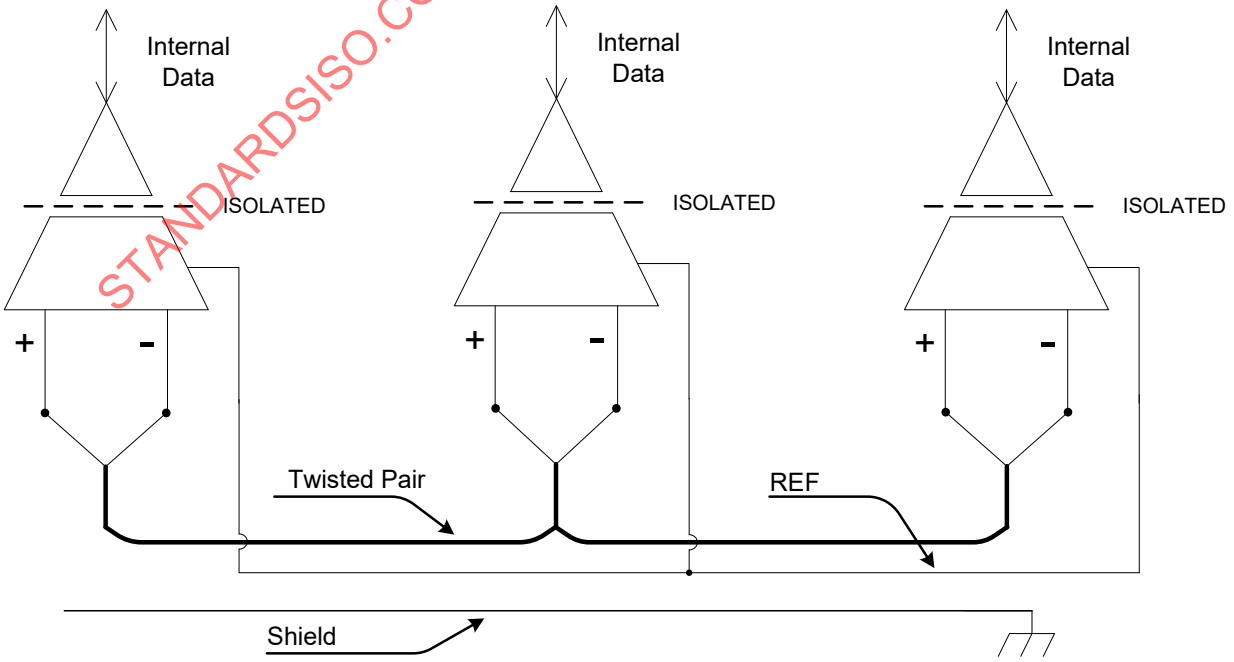


Figure 9-1.3. All Isolated Devices on 3-Conductor Cable with Shield.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.2.2.1.1.4 Twisted-pair and Reference with Mixed Devices

If the installation includes a mixture of two-wire non-isolated devices and three-wire isolated devices, they may be used together in the configuration of Figure 9-1.4 if the two-wire devices are installed in areas with low electrical noise or if high levels of electrical noise are generated locally at the three-wire isolated devices and the remainder of the installation is electrically quiet. In this installation, a third conductor in the same cable is used to connect all of the reference connections together. Three-wire devices with a reference connection shall be directly tied to the third conductor and two-wire device reference connections are made indirectly through a single 100-ohm current-limiting resistor tied between the reference conductor and earth ground in a low-noise area, preferably near the supervisory controller.

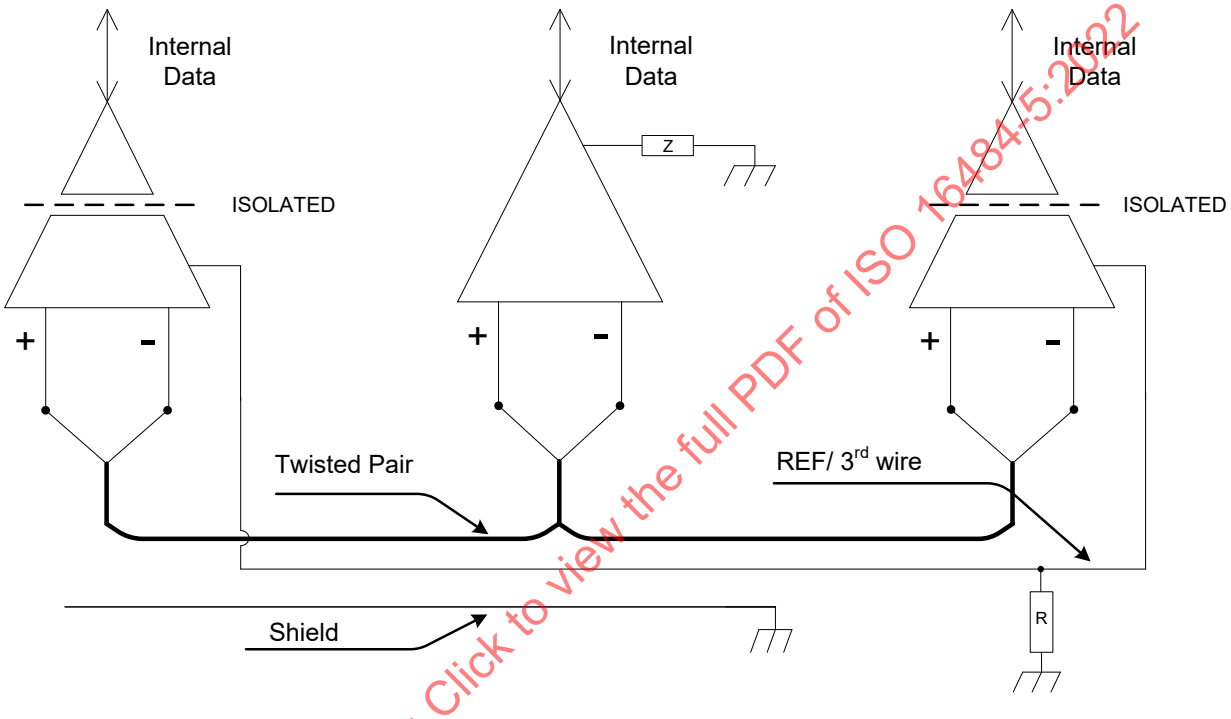


Figure 9-1.4. Mixed Devices on 3-Conductor Cable with Shield.

9.2.2.1.1.5 Extending Twisted-pair with Reference

If the installation includes existing two-wire non-isolated devices that are to be extended with three-wire isolated devices, they may be connected together in the configuration of Figure 9-1.5. In this installation, a third conductor in the extended cable is used to connect all of the reference connections together. Three-wire devices with a reference connection shall be directly tied to the third conductor and two-wire device reference connections are made indirectly through a single 100-ohm current-limiting resistor tied between the reference conductor and earth ground in a low noise area, preferably where the extended cable is connected to the existing twisted-pair cable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

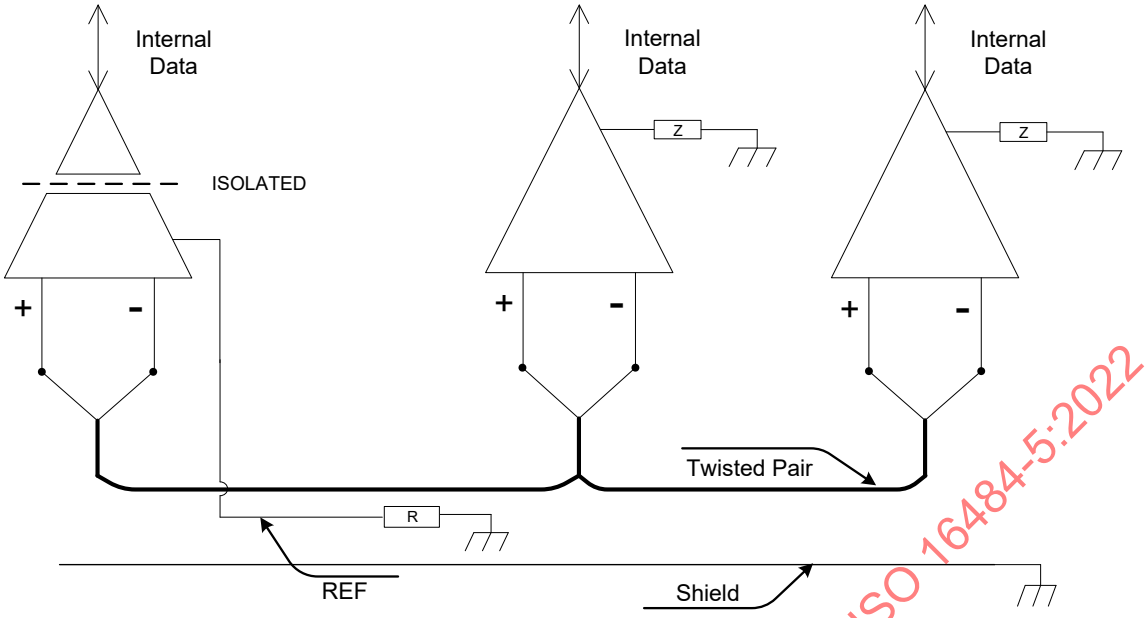


Figure 9-1.5. Extending Existing Twisted-pair with Isolated Devices.

9.2.2.1.2 Multiple Buildings

Connecting multiple buildings using MS/TP shall have at least 1500 V of electrical isolation as specified in Clause 9.2.2. The following clauses describe ways to provide the required isolation.

9.2.2.1.2.1 Isolated Devices

Installations that connect multiple buildings using a single cable are permitted if the secondary buildings contain only three-wire isolated devices with 1500-volt electrical isolation capability and there are no earth ground connections in the secondary buildings to the reference conductor or the shield as shown in Figure 9-1.6. When the primary building contains only two-wire non-isolated devices, the reference conductor shall be connected to earth ground at one location through a 100-ohm current-limiting resistor (R) in that building and the shield shall be tied to earth ground in a single location in the same building. The use of surge arrestors near each building's cable entrance to protect all of the conductors is recommended.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

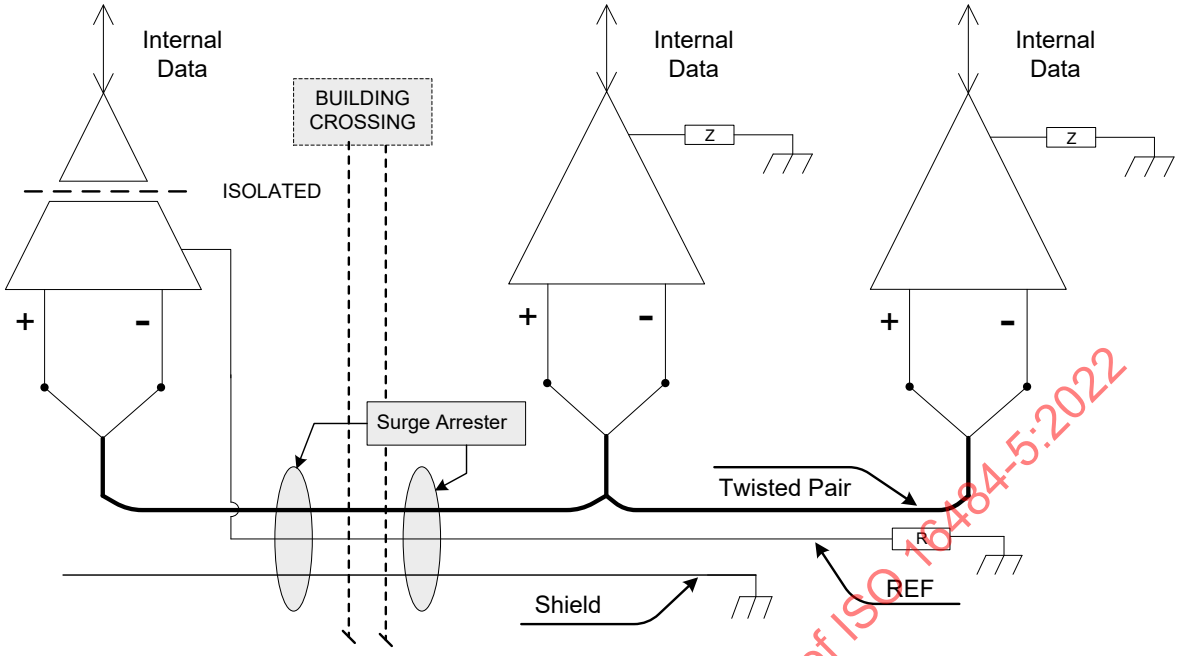


Figure 9-1.6. Two Buildings with All Isolated Devices in One Building.

9.2.2.1.2.2 Isolated Repeater

If the installation includes a mixture of three-wire isolated and two-wire non-isolated devices in each of two buildings on a single communications line, it is possible to connect the buildings using an isolated repeater so that each building is electrically isolated from the other building as shown in Figure 9-1.7. The isolated repeater must provide complete three-way electrical isolation between the wiring on either side and ground. In this case, the communication wiring within each building is configured like that of a single building since the isolated repeater provides the required 1500-volt electrical isolation. The cable connecting the buildings is an extension of the cable in one of the buildings and shall be electrically isolated from the other building by the isolated repeater and shall not have any connections to other devices or to ground within the other building.

An isolated repeater may also be used on both sides of the cable connecting the buildings. This may be needed for extra isolation or for cable length. In this case the cable connecting the buildings shall be separately shielded, terminated, and biased and shall not have any connections to other devices within either building. If the pair of isolated repeaters provides a reference connection, the two reference connections shall be joined by a third conductor within the cable connecting the buildings and shall not be connected to any other device or to ground.

The use of surge arrestors near each building's cable entrance to protect all of the conductors is recommended.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

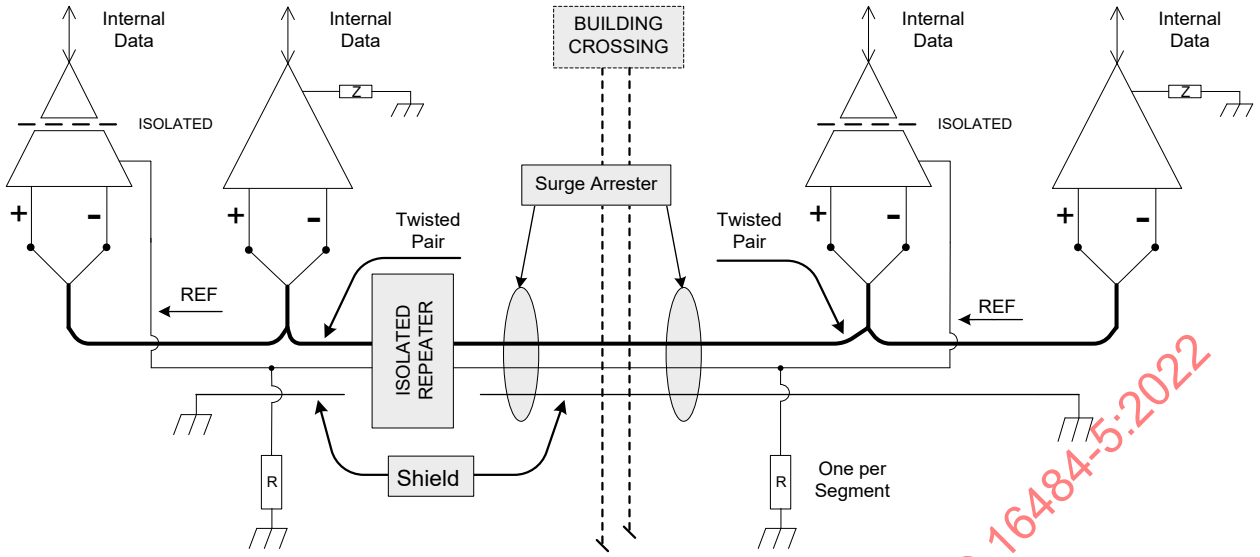


Figure 9-1.7. Isolated Repeater between Two Buildings with Surge Arresters.

9.2.2.1.2.3 Fiber Optic Isolation

If the installation includes a mixture of three-wire isolated and two-wire non-isolated devices in each of two buildings on a single communications line, it is best to connect the buildings using EIA-485 half-duplex compatible fiber optic modems so that each building is electrically isolated from the other building and there are no conductors outside the buildings as shown in Figure 9-1.8. In this case, the communication wiring within each building is configured like that of a single building since the fiber optic modems provide the required 1500-volt electrical isolation.

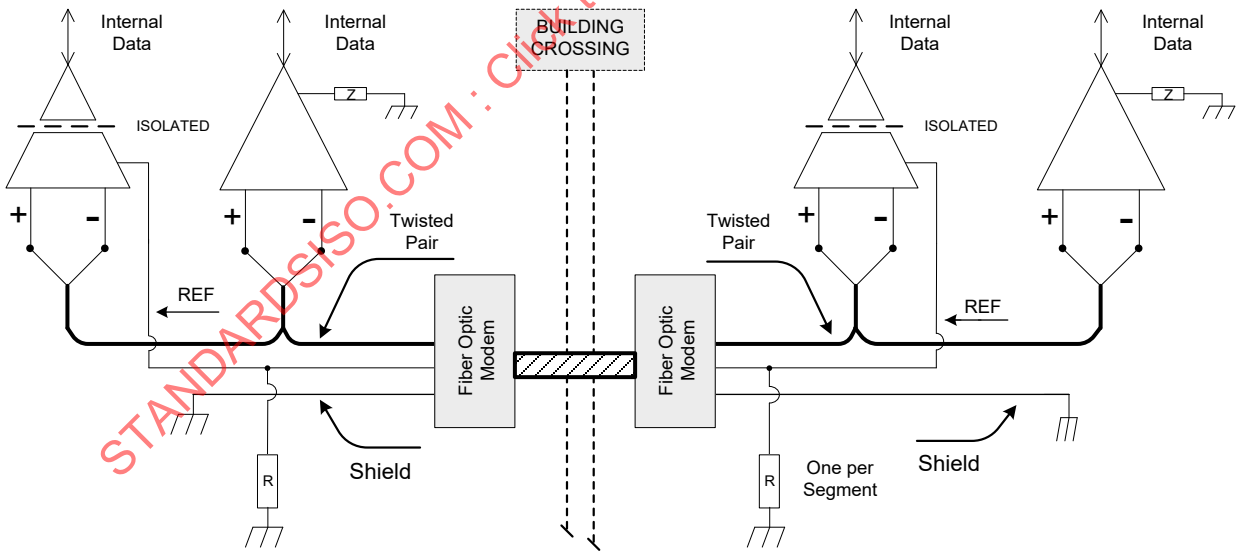


Figure 9-1.8. Fiber Optic Modems between Two Buildings.

9.2.2.1.2.4 No Isolation (not permitted)

Directly connecting two buildings with a single EIA-485 communications cable where there are two-wire non-isolated devices in each building shall not be permitted since the required 1500-V electrical isolation between signal conductors and digital ground cannot be maintained. Device wiring such as that shown in Figure 9-1.9 is not allowed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

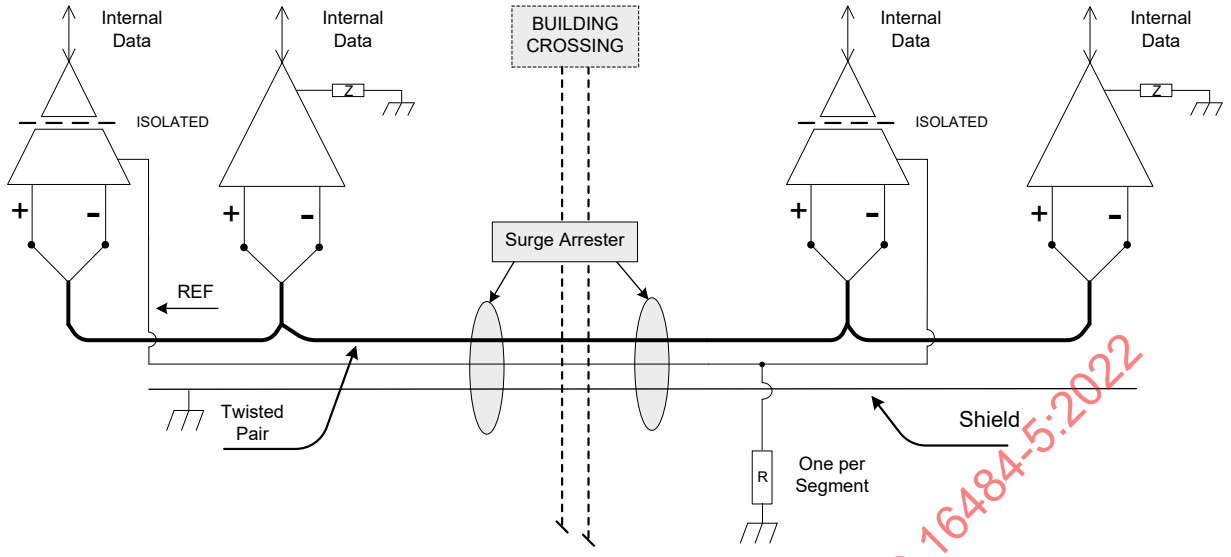
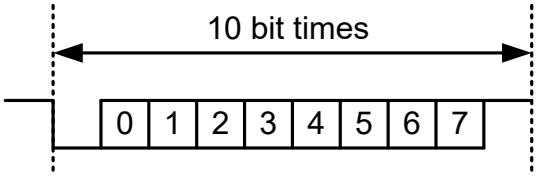


Figure 9-1.9. Two Buildings with Mixed Devices - No Isolation (Not Allowed).

9.2.3 Timing

Octets shall be transmitted using non-return to zero (NRZ) encoding with one start bit, eight data bits, no parity, and one stop bit. The start bit shall have a value of zero, while the stop bit shall have a value of one. The data bits shall be transmitted with the least significant bit first. This is illustrated in Figure 9-2.



Although asynchronous framing is used, there shall be no more than $\mathbf{T}_{\mathbf{frame_gap}}$ of idle line (logical ones or stop bits) between any two octets of a frame.

'The standard baud rates are shown in the table below. The required baud rates, plus or minus 1%, shall be supported. Any or all of the optional baud rates, plus or minus 1%, may be supported at the vendor's option.

Baud Rate	Requirement	Recommended Maximum Distance
9600	Required	1200 meters (4000 feet)
19200	Optional	1200 meters (4000 feet)
38400	Required	1200 meters (4000 feet)
57600	Optional	1200 meters (4000 feet)
76800	Optional	1200 meters (4000 feet)
115200	Optional	1000 meters (3280 feet)

Transmitter enable: A node shall enable its EIA-485 driver before it generates the leading edge of the first start bit of a frame. The node shall drive the line to the logical one state during the time between the enable and the leading edge of the first start bit of a frame.

Transmitter disable: A node shall not disable its EIA-485 driver until the stop bit of the final octet of a frame has been generated. The node shall disable its EIA-485 driver within $\mathbf{T}_{\mathbf{postdrive}}$ after the beginning of the stop bit of the final octet of a frame in order that it not interfere with any subsequent frame transmitted by another node. This specification allows, but does not encourage, the use of a "padding" octet after the final octet of a frame in order to facilitate the use of common UART transmit interrupts for driver disable control. If a "padding" octet is used, its value shall be X'FF'. The "padding" octet is not considered part of the frame, that is, it shall be included within $\mathbf{T}_{\mathbf{postdrive}}$.

Receive to Transmit turn-around: A node shall not enable its EIA-485 driver for at least $\mathbf{T}_{\mathbf{turnaround}}$ after the node receives the final stop bit of any octet.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.3 MS/TP Frame Format

All frames are of the following format:

Preamble	two octet preamble: X'55', X'FF'
Frame Type	one octet
Destination Address	one octet address
Source Address	one octet address
Length	two octets, most significant octet first
Header CRC	one octet
Data	(present only if Length is non-zero and frame is a non-encoded type)
Data CRC	(present only if Length is non-zero and frame is a non-encoded type) two octets, least significant octet first
Encoded Data	(present only if frame is a COBS-encoded type)
Encoded CRC-32K	(present only if frame is a COBS-encoded type) five octets; CRC-32K generated according to Clause 9.6 and COBS-encoded according to Clause 9.10
(pad)	(optional) at most one octet of padding: X'FF'

The Frame Type is used to distinguish between different types of MAC frames. Defined types are:

00	Token
01	Poll For Master
02	Reply To Poll For Master
03	Test_Request
04	Test_Response
05	BACnet Data Expecting Reply
06	BACnet Data Not Expecting Reply
07	Reply Postponed
32	BACnet Extended Data Expecting Reply
33	BACnet Extended Data Not Expecting Reply

Frame Types 8 through 31 and 34 through 127 are reserved by ASHRAE. Open standard development organizations wishing to convey network protocols other than BACnet over MS/TP may apply for a Frame Type allocation from the reserved range by contacting the ASHRAE Manager of Standards.

Frame Types 32 through 127 indicate COBS-encoded frames and shall convey Encoded Data and Encoded CRC-32K fields (see Clause 9.10). All other values for Frame Type indicate a non-encoded frame. Support for COBS-encoded BACnet MS/TP frames is required for BACnet routing nodes and optional for non-routing nodes. Devices that support COBS-encoded BACnet MS/TP frames shall support both BACnet Extended Data Expecting Reply and BACnet Extended Data Not Expecting Reply Frame Types.

Frame Types 128 through 255 are available to vendors for proprietary (non-BACnet) frames. Use of proprietary frames might allow a Brand-X controller, for example, to send proprietary frames to other Brand-X controllers that do not implement BACnet, while using the same medium to send BACnet frames to a Brand-Y panel that does implement BACnet. Token, Poll For Master, and Reply To Poll For Master frames shall be understood by all MS/TP master nodes.

The Destination and Source Addresses are one octet each. A Destination Address of 255 (X'FF') denotes broadcast. A Source Address of 255 is not allowed. Addresses 0 to 127 are valid for both master and slave nodes. Addresses 128 to 254 are valid only for slave nodes.

MS/TP devices shall support configurable MAC addresses, and each shall be able to be set to any valid unicast address (0..127 for masters and 0..254 for slaves). Where a device has multiple MS/TP ports, the MAC address of each port shall be settable to any valid value regardless of the MAC address settings of the other MS/TP ports.

For non-encoded frames, the Length field specifies the length in octets of the Data field and shall be between 0 and 501 octets.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

For non-encoded frames, the Data and Data CRC fields are conditional on the Frame Type and the Length, as specified in the description of each Frame Type. If the Length field is zero, that is, if both length octets are zero, then the Data and Data CRC fields shall not be present.

For COBS-encoded frames, the Length field specifies the combined length of the Encoded Data and Encoded CRC-32K fields in octets, minus two. (Two octets are subtracted to maintain backward compatibility with devices that implement the SKIP_DATA state in their Receive Frame state machine.)

For COBS-encoded frames, the Length field shall be in the range $\mathbf{N}_{\mathbf{min_COBS_length}}$ to $\mathbf{N}_{\mathbf{max_COBS_length}}$ (see Clause 9.5.3) and the Encoded Data and Encoded CRC-32K fields shall always be present.

Clause 9.6 and Annex G describe in detail the generation and checking of the Header, Data CRC, and CRC-32K octets.

Clause 9.10 and Annex T describe in detail the COBS-encoding of the Encoded Data and Encoded CRC-32K fields.

9.3.1 Frame Type 00: Token

The Token frame is used to pass network mastership to the destination node. The use of the Token frame is described in detail in Clause 9.5.

There are no data octets in Token frames.

9.3.2 Frame Type 01: Poll For Master

The Poll For Master frame is transmitted by master nodes during configuration and periodically during normal network operation. It is used to discover the presence of other master nodes on the network and to determine a successor node in the token ring. The use of the Poll For Master frame in the token network is described in detail in Clause 9.5.

There are no data octets in Poll For Master frames.

Both master and slave nodes shall expect to receive Poll For Master frames. Master nodes shall respond to Poll For Master Frames as described in Clause 9.5.6.2. Slave nodes shall ignore Poll For Master frames, as described in Clause 9.5.7.2.

9.3.3 Frame Type 02: Reply To Poll For Master

This frame is transmitted as a reply to the Poll For Master frame. It is used to indicate that the node sending the frame wishes to enter the token ring. The use of this frame in the token network is described in detail in Clause 9.5.

There are no data octets in Reply To Poll For Master frames.

9.3.4 Frame Type 03: Test_Request

This frame is used to initiate a loopback test of the MS/TP to MS/TP transmission path. The use of this frame in the token network is described in detail in Clause 9.1.3. The length of the data portion of a Test_Request frame may range from 0 to 501 octets.

9.3.5 Frame Type 04: Test_Response

This frame is used to reply to Test_Request frames. The use of this frame in the token network is described in detail in Clause 9.1.3. The length of the data portion of a Test_Response frame may range from 0 to 501 octets. The data, if present, shall be that which was present in the initiating Test_Request.

9.3.6 Frame Type 05: BACnet Data Expecting Reply

This frame is used by master nodes to convey the data parameter of a DL_UNITDATA.request whose DER parameter is TRUE. The length of the data portion of a BACnet Data Expecting Reply frame may range from 0 to 501 octets.

9.3.7 Frame Type 06: BACnet Data Not Expecting Reply

This frame is used to convey the data parameter of a DL_UNITDATA.request whose DER parameter is FALSE. The length of the data portion of a BACnet Data Not Expecting Reply frame may range from 0 to 501 octets.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.3.8 Frame Type 07: Reply Postponed

This frame is used by master nodes to defer sending a reply to a previously received BACnet Data Expecting Reply frame. The use of this frame in the token network is described in detail in Clause 9.5.6.

There are no data octets in Reply Postponed frames.

9.3.9 Frame Type 32: BACnet Extended Data Expecting Reply

This COBS-encoded frame is used by master nodes to convey the data parameter of a DL_UNITDATA.request whose data_expecting_reply parameter is TRUE and whose data parameter length is between 502 and 1497 octets, inclusive.

9.3.10 Frame Type 33: BACnet Extended Data Not Expecting Reply

This COBS-encoded frame is used by master nodes to convey the data parameter of a DL_UNITDATA.request whose data_expecting_reply parameter is FALSE and whose data parameter length is between 502 and 1497 octets, inclusive.

9.3.11 Frame Types 128 through 255: Proprietary Frames

These frames are available to vendors as proprietary (non-BACnet) frames. The first two octets of the Data field shall specify the unique vendor identification code, most significant octet first, for the type of vendor-proprietary frame to be conveyed. The length of the data portion of a proprietary frame shall be in the range of 2 to 501 octets.

9.4 Overview of the MS/TP Network

MS/TP uses a token to control access to a bus network. A master node may initiate the transmission of a data frame when it holds the token. Both master and slave nodes may transmit data frames in response to requests from master nodes. After sending at most $\mathbf{N}_{\mathbf{max_info_frames}}$ data frames (and awaiting any expected replies), a master node shall pass the token to the next master node.

It is generally easier to deal with a lost token than with the presence of two tokens in a ring: a simple timeout will detect token loss, and regeneration of the token and recovery of the ring may proceed in an orderly fashion. If more than one token exists, however, collisions are likely. These will disrupt communications and slow throughput but may not be severe enough to cause loss of the tokens. In such a case, a persistent reduction in throughput might result. For this reason, the ring maintenance rules in this clause favor the loss of the token over the creation of a second token.

Token frames are not acknowledged. If the acknowledgment of a token were lost, the token's sender might retry, resulting in the creation of two tokens. Instead, after a node passes the token, it listens to see if the intended receiver node begins using the token. Usage in this case is defined as the reception of $\mathbf{N}_{\mathbf{min_octets}}$ octets from the network within $\mathbf{T}_{\mathbf{usage_timeout}}$ after the final octet of the Token frame is transmitted.

Most token bus networks, such as ARCNET, do not distinguish between requests and replies: both are passed in the same type of frames, which are sent only when the sending node has the token. Since MS/TP defines slave nodes that never hold the token, a means must be provided to allow replies to be returned from slave devices. For simplicity, the same mechanism is used for replies returned from master nodes.

When a request that expects a reply is sent to an MS/TP node, the sender shall wait for the reply to be returned before passing the token. If the responding node is a master, it may return the reply or it may return a Reply Postponed frame, indicating that the actual reply will be returned later, when the replying node holds the token.

9.5 MS/TP Medium Access Control

The description that follows defines variables and procedures that may in some ways resemble the variables and procedures used in various computer languages. This description is in no way intended to prescribe the method of implementation. An implementation may be constructed in any fashion desired as long as it matches the behavior described by this standard. The description that follows is intended only to specify that behavior clearly and precisely.

9.5.1 UART Receiver Model

In this clause, we present a model of the receiver interface to a UART as a data register and two Boolean flags. These are intended to closely resemble the functions of commercial UART chips but in a generic and non-prescriptive fashion. The model is used by the procedural and state machine descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.5.1.1 DataRegister

The DataRegister holds the octet most recently received. The contents of this register after the occurrence of a framing or overrun error are not specified.

9.5.1.2 DataAvailable

The flag DataAvailable is TRUE if an octet is available in DataRegister. A means of setting this flag to FALSE when the associated data have been read from DataRegister shall be provided. Many common UARTs set DataAvailable FALSE automatically when DataRegister is read.

9.5.1.3 ReceiveError

The flag ReceiveError is TRUE if an error is detected during the reception of an octet. Many common UARTs detect several types of receive errors, in particular framing errors and overrun errors. ReceiveError shall be TRUE if any of these errors is detected.

A framing error occurs if a logical zero is received when a stop bit (logical one) is expected.

An overrun error occurs if an octet is received before an earlier octet is read from DataRegister. In general, the occurrence of overrun errors is evidence of improper design. However, it is recognized that critical system events may cause overrun errors to occur from time to time. The inclusion of this error in the state machine processing ensures that such errors are handled in a well-defined fashion.

A means of setting ReceiveError to FALSE when the associated error has been recognized shall be provided.

9.5.2 Variables

A number of variables and timers are used in the descriptions that follow:

DataCRC Used to accumulate the CRC on the data field of a frame.

DataLength Used to store the data length of a received frame.

DestinationAddress Used to store the destination address of a received frame.

EventCount Used to count the number of received octets or errors. This is used in the detection of link activity.

FrameType Used to store the frame type of a received frame.

FrameCount The number of frames sent by this node during a single token hold. When this counter reaches the value $\mathbf{N}_{\mathbf{max_info_frames}}$, the node must pass the token.

HeaderCRC Used to accumulate the CRC on the header of a frame.

Index Used as an index by the Receive State Machine, up to the value of DataLength+1.

InputBuffer[] An array of octets, used to store octets as they are received. InputBuffer is indexed from 0 to InputBufferSize-1.

InputBufferSize The number of elements in the array InputBuffer[]. Routing nodes shall support the maximum NPDU size. Non-routing nodes may support a smaller value. Devices that support COBS-encoded frames shall include additional octets to account for encoding overhead. The overhead is calculated as the maximum supported NPDU size divided by 254 (any fractional part is rounded up to the nearest integer) plus five octets for the Encoded CRC-32K. For example, the InputBufferSize required for a device that supports the maximum BACnet NPDU size is 1497 + 6 + 5 = 1508.

GoodHeader A Boolean flag set to TRUE or FALSE by the CheckHeader procedure (see Clause 9.5.8).

CRC32K Used by devices that support COBS-encoded frames to accumulate the CRC-32K on the Encoded Data field.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NS "Next Station," the MAC address of the node to which This Station passes the token. If the Next Station is unknown, NS shall be equal to TS.

PS "Poll Station," the MAC address of the node to which This Station last sent a Poll For Master. This is used during token maintenance.

ReceivedInvalidFrame A Boolean flag set to TRUE by the Receive State Machine if an error is detected during the reception of a frame. Set to FALSE by the main state machine.

ReceivedValidFrame A Boolean flag set to TRUE by the Receive State Machine if a valid frame is received. Set to FALSE by the main state machine.

RetryCount A counter of transmission retries used for Token and Poll For Master transmission.

SilenceTimer A timer with nominal 5 millisecond resolution used to measure and generate silence on the medium between octets. It is incremented by a timer process and is cleared by the Receive State Machine when activity is detected and by the SendFrame procedure as each octet is transmitted. Since the timer resolution is limited and the timer is not necessarily synchronized to other machine events, a timer value of N will actually denote intervals between N-1 and N.

SoleMaster A Boolean flag set to TRUE by the master machine if this node is the only known master node.

SourceAddress Used to store the Source Address of a received frame.

TokenCount The number of tokens received by this node. When this counter reaches the value $\mathbf{N_{poll}}$, the node polls the address range between **TS** and **NS** for additional master nodes. **TokenCount** is set to one at the end of the polling process.

TS "This Station," the MAC address of this node. This variable represents the value of the MAC_Address property of the node's Network Port object which represents this MS/TP port. Valid values for TS are 0 to 254. The value 255 is used to denote broadcast when used as a destination address but is not allowed as a value for TS.

9.5.3 Parameters

Parameter values used in the description:

$\mathbf{N_{max_info_frames}}$ This parameter represents the value of the Max_Info_Frames property of the node's Network Port object which represents this MS/TP port. The value of Max_Info_Frames specifies the maximum number of information frames the node may send before it must pass the token. Max_Info_Frames may have different values on different nodes but shall have a minimum value of 1 and a maximum value of 255. This may be used to allocate more or less of the available link bandwidth to particular nodes.

$\mathbf{N_{max_master}}$ This parameter represents the value of the Max_Master property of the node's Network Port object which represents this MS/TP port. The value of Max_Master specifies the highest allowable address for master nodes. The value of this parameter shall be less than or equal to 127.

$\mathbf{N_{poll}}$ The number of tokens received or used before a Poll For Master cycle is executed: 50.

$\mathbf{N_{retry_token}}$ The number of retries on sending Token: 1.

$\mathbf{N_{min_octets}}$ The minimum number of DataAvailable or ReceiveError events that must be seen by a receiving node in order to declare the line "active": 4.

$\mathbf{N_{min_COBS_type}}$ The first COBS-encoded Frame Type value: 32.

$\mathbf{N_{max_COBS_type}}$ The last COBS-encoded Frame Type value: 127.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

$N_{min_COBS_length}$ The minimum valid Length value of any COBS-encoded frame: 5. The theoretical minimum Length is calculated as follows: COBS-encoded frames must contain at least one data octet. The minimum COBS encoding overhead for the Encoded Data field is one octet. The size of the Encoded CRC-32K field is always five octets. Adding the lengths of these fields and subtracting two (for backward compatibility) results in a minimum Length value of five (1 + 1 + 5 - 2). In practice, the minimum Length value is determined by the network-layer client and is likely to be larger (e.g., for BACnet the minimum Length is 502 + 1 + 3 = 506.

$N_{max_COBS_length}$ The maximum valid Length value of any COBS-encoded frame: 2043. The theoretical maximum Length is calculated as follows: the largest data parameter that any future network client may specify is 2032 octets (this is near the limit of the CRC-32K's maximum error-detection capability). The worst-case COBS encoding overhead for the Encoded Data field would be 2032 / 254 = 8 octets. Adding in the size adjustment for the Encoded CRC-32K results in a maximum Length value of 2032 + 8 + 3 = 2043. In practice, the maximum Length value is determined by the network-layer client and is likely to be smaller (e.g., for BACnet the maximum Length is 1497 + 6 + 3 = 1506.

T_{frame_abort} The minimum time without a DataAvailable or ReceiveError event within a frame before a receiving node may discard the frame: 60 bit times. (Implementations may use larger values for this timeout, not to exceed 100 milliseconds.)

T_{frame_gap} The maximum idle time a sending node may allow to elapse between octets of a frame the node is transmitting: 20 bit times.

T_{no_token} The time without a DataAvailable or ReceiveError event before declaration of loss of token: 500 milliseconds.

$T_{postdrive}$ The maximum time after the end of the stop bit of the final octet of a transmitted frame before a node must disable its EIA-485 driver: 15 bit times.

T_{reply_delay} The maximum time a node may wait after reception of a frame that expects a reply before sending the first octet of a reply or Reply Postponed frame: 250 milliseconds.

$T_{reply_timeout}$ The minimum time without a DataAvailable or ReceiveError event that a node must wait for a station to begin replying to a confirmed request: 255 milliseconds. (Implementations may use larger values for this timeout, not to exceed 300 milliseconds.)

T_{roff} Repeater turnoff delay. The duration of a continuous logical one state at the active input port of an MS/TP repeater after which the repeater will enter the IDLE state: 29 bit times < T_{roff} < 40 bit times.

T_{slot} The width of the time slot within which a node may generate a token: 10 milliseconds.

$T_{turnaround}$ The minimum time after the end of the stop bit of the final octet of a received frame before a node may enable its EIA-485 driver: 40 bit times.

T_{usage_delay} The maximum time a node may wait after reception of the token or a Poll For Master frame before sending the first octet of a frame: 15 milliseconds.

$T_{usage_timeout}$ The time without a DataAvailable or ReceiveError event that a sending node must wait for a receiving node to begin using a token or replying to a Poll For Master frame: 20 milliseconds. Sending node implementations may use larger values for this timeout, not to exceed 35 milliseconds.

9.5.4 Receive Frame Finite State Machine

This section describes the reception of an MS/TP frame by a BACnet device. The description of operation is as a finite state machine. Figure 9-3 shows the Receive Frame state machine, which is described fully in this clause. Each state is given a name, specified in all capital letters. Transitions are also named, in mixed upper- and lowercase letters. Transitions are described as a series of conditions followed by a series of actions to be taken if the conditions are met. The final action in each transition is entry into a new state, which may be the same as the current state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

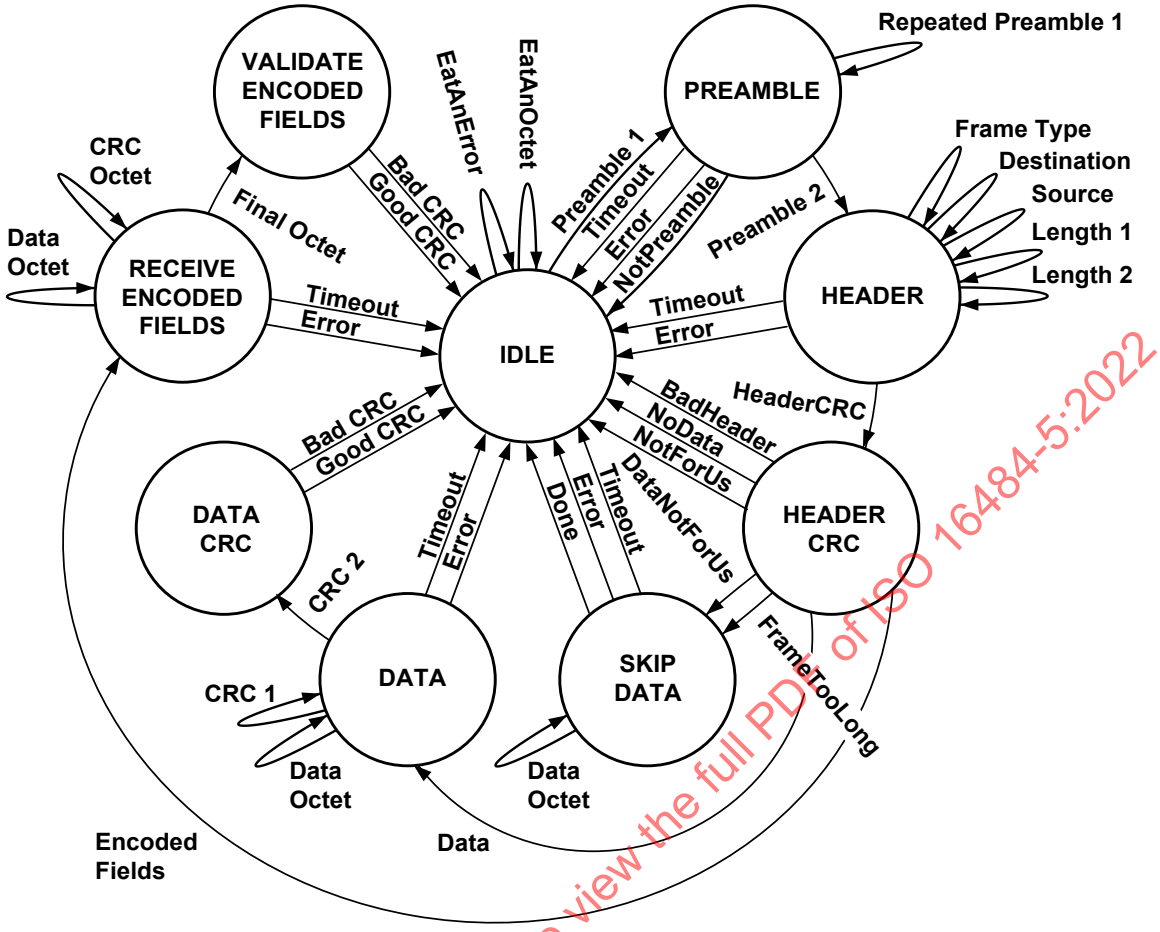


Figure 9-3. Receive Frame State Machine.

The Receive Frame state machine operates independently from the MS/TP Master Node or Slave Node machine, communicating with it by means of flags and other variables. The description assumes that the Master Node or Slave Node state machine can process received frames and other indications from the Receive Frame state machine before the next frame begins. The means by which this behavior is implemented are a local matter.

This description assumes that the node will not receive its own transmissions. If a given implementation does receive its own transmissions, then the implementation shall be constructed so that the Receive Frame machine will ignore the transmissions.

9.5.4.1 IDLE

In the IDLE state, the node waits for the beginning of a frame.

EatAnError

If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; increment EventCount; and enter the IDLE state to wait for the start of a frame.

EatAnOctet

If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is not X'55',

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; and enter the IDLE state to wait for the start of a frame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Preamble1
If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is X '55',

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; and enter the PREAMBLE state to receive the remainder of the frame.

9.5.4.2 PREAMBLE

In the PREAMBLE state, the node waits for the second octet of the preamble.

Timeout
If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

then a correct preamble has not been received. Enter the IDLE state to wait for the start of a frame.

Error
If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; increment EventCount; and enter the IDLE state to wait for the start of a frame.

RepeatedPreamble1
If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is X'55',

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; and enter the PREAMBLE state to wait for the second preamble octet.

NotPreamble
If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is not X'FF' or X '55',

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; and enter the IDLE state to wait for the start of a frame.

Preamble2
If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is X'FF',

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; set Index to zero; set HeaderCRC to X'FF'; and enter the HEADER state to receive the remainder of the frame.

9.5.4.3 HEADER

In the HEADER state, the node waits for the fixed message header.

Timeout
If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of a frame.

Error
If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; increment EventCount; set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame; and enter the IDLE state to wait for the start of a frame.

FrameType
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 0,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; save the contents of DataRegister as FrameType; set Index to 1; and enter the HEADER state.

Destination
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 1

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; save the contents of DataRegister as DestinationAddress; set Index to 2; and enter the HEADER state.

Source
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 2,

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; save the contents of DataRegister as SourceAddress; set Index to 3; and enter the HEADER state.

Length1
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 3,

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; multiply the contents of DataRegister by 256 and save the result as DataLength; set Index to 4; and enter the HEADER state.

Length2
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 4,

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; add the contents of DataRegister to DataLength and save the result as DataLength; set Index to 5; and enter the HEADER state.

HeaderCRC
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 5,

then set DataAvailable to FALSE; set SilenceTimer to zero; increment EventCount; accumulate the contents of DataRegister into HeaderCRC; call the CheckHeader procedure defined in Clause 9.5.8; and enter the HEADER_CRC state.

9.5.4.4 HEADER_CRC

In the HEADER_CRC state, the node validates the CRC on the fixed message header and tests for illegal values in other header fields.

BadHeader
If GoodHeader is FALSE,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

NotForUs
If GoodHeader is TRUE and DataLength is zero and the value of DestinationAddress is not equal to either TS (this station) or 255 (broadcast),

then enter the IDLE state to wait for the start of the next frame.

DataNotForUs
If GoodHeader is TRUE and DataLength is not zero and the value of DestinationAddress is not equal to either TS (this station) or 255 (broadcast),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then set Index to zero and enter the SKIP_DATA state to consume the Data and Data CRC or Encoded Data and Encoded CRC-32K portions of the frame.

FrameTooLong

If GoodHeader is TRUE and the value of DestinationAddress is equal to either TS (this station) or 255 (broadcast) and DataLength is greater than InputBufferSize,

then set ReceivedInvalidFrame to TRUE to indicate that a frame with an illegal or unacceptable DataLength has been received, set Index to zero, and enter the SKIP_DATA state to consume the Data and Data CRC or Encoded Data and Encoded CRC-32K portions of the frame.

NoData

If GoodHeader is TRUE and the value of DestinationAddress is equal to either TS (this station) or 255 (broadcast) and DataLength is zero,

then set ReceivedValidFrame to TRUE to indicate that a frame with no data has been received, and enter the IDLE state to wait for the start of the next frame.

Data

If GoodHeader is TRUE and the value of DestinationAddress is equal to either TS (this station) or 255 (broadcast) and DataLength is not zero and DataLength is less than or equal to InputBufferSize and either this device does not support COBS-encoded frames or FrameType indicates a non-encoded frame,

then set Index to zero; set DataCRC to X'FFFF'; and enter the DATA state to receive the data portion of the frame.

EncodedFields

If GoodHeader is TRUE and the value of DestinationAddress is equal to either TS (this station) or 255 (broadcast) and DataLength is less than or equal to InputBufferSize and this device supports COBS-encoded frames and FrameType indicates a COBS-encoded frame,

then set Index to zero; set CRC32K to X'FFFFFFFF'; and enter the RECEIVE_ENCODED_FIELDS state to receive the Encoded Data and Encoded CRC-32K fields of the frame.

9.5.4.5 DATA

In the DATA state, the node waits for the data portion of a frame.

Timeout

If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

Error

If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame; and enter the IDLE state to wait for the start of the next frame.

DataOctet

If ReceiveError is FALSE and DataAvailable is TRUE and Index is less than DataLength,

then set DataAvailable to FALSE; set SilenceTimer to zero; accumulate the contents of DataRegister into DataCRC; save the contents of DataRegister at InputBuffer[Index]; increment Index by 1; and enter the DATA state.

CRC1

If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then set DataAvailable to FALSE; set SilenceTimer to zero; accumulate the contents of DataRegister into DataCRC; increment Index by 1; and enter the DATA state.

CRC2
If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength plus 1,

then set DataAvailable to FALSE; set SilenceTimer to zero; accumulate the contents of DataRegister into DataCRC; and enter the DATA_CRC state.

9.5.4.6 DATA_CRC

In the DATA_CRC state, the node validates the CRC of the message data.

BadCRC
If the value of DataCRC is not X'F0B8',

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

GoodCRC
If the value of DataCRC is X'F0B8',

then set ReceivedValidFrame to TRUE to indicate the complete reception of a valid frame, and enter the IDLE state to wait for the start of the next frame.

9.5.4.7 SKIP_DATA

In the SKIP_DATA state, the node waits for the data portion of a frame to be received so that its contents can be ignored.

Timeout
If SilenceTimer is greater than Tframe_abort,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

Error
If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame; and enter the IDLE state to wait for the start of the next frame.

DataOctet
If ReceiveError is FALSE and DataAvailable is TRUE and Index is less than DataLength+1,

then set DataAvailable to FALSE; set SilenceTimer to zero; increment Index by 1; and enter the SKIP_DATA state.

Done
If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength+1,

then set DataAvailable to FALSE; set SilenceTimer to zero; and enter the IDLE state to wait for the start of the next frame.

9.5.4.8 RECEIVE_ENCODED_FIELDS

In the RECEIVE_ENCODED_FIELDS state, the node waits for and decodes the Encoded Data and Encoded CRC-32K fields of a frame according to the procedure described in Clause 9.10.3. If the device does not support COBS-encoded frames, this state is unreachable and should not be implemented.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Timeout

If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

Error

If ReceiveError is TRUE,

then set ReceiveError to FALSE; set SilenceTimer to zero; set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame; and enter the IDLE state to wait for the start of the next frame.

DataOctet

If ReceiveError is FALSE and DataAvailable is TRUE and Index is less than DataLength minus 3,

then set DataAvailable to FALSE; set SilenceTimer to zero; accumulate the contents of DataRegister into CRC32K; save the contents of DataRegister at InputBuffer[Index]; increment Index by 1; and enter the RECEIVE_ENCODED_FIELDS state.

CRCOctet

If ReceiveError is FALSE and DataAvailable is TRUE and Index is greater than or equal to DataLength minus 3 and Index is less than DataLength plus 1,

then set DataAvailable to FALSE; set SilenceTimer to zero; save the contents of DataRegister at InputBuffer[Index]; increment Index by 1; and enter the RECEIVE_ENCODED_FIELDS state.

FinalOctet

If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength plus 1,

then set DataAvailable to FALSE; set SilenceTimer to zero; save the contents of DataRegister at InputBuffer[Index]; decode the Encoded Data and Encoded CRC-32K fields according to Clause 9.10.3; and enter the VALIDATE_ENCODED_FIELDS state.

9.5.4.9 VALIDATE_ENCODED_FIELDS

In the VALIDATE_ENCODED_FIELDS state, the node validates the received CRC-32K of the frame. If the device does not support COBS-encoded frames, this state is unreachable and should not be implemented.

BadCRC

If the value of CRC32K is not X'0843323B',

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

GoodCRC

If the value of CRC32K is X'0843323B',

then set ReceivedValidFrame to TRUE to indicate the complete reception of a valid frame, and enter the IDLE state to wait for the start of the next frame.

9.5.5 The SendFrame Procedure

The transmission of an MS/TP frame proceeds as follows:

Procedure SendFrame

If Frame Type is less then Nmin_COBS_type or Frame Type is greater than Nmax_COBS_type ,
then call SendNonEncodedFrame,
else call SendCOBS_EncodedFrame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

9.5.5.1 SendNonEncodedFrame Procedure for Non-Encoded Frame Types

(1) If SilenceTimer is less than $\mathbf{T}_{\mathbf{turnaround}}$, wait ($\mathbf{T}_{\mathbf{turnaround}}$ - SilenceTimer).
(2) Disable the receiver, and enable the transmit line driver.
(3) Transmit the preamble octets X'55', X'FF'. As each octet is transmitted, set SilenceTimer to zero.
(4) Initialize HeaderCRC to X'FF'.
(5) Transmit the Frame Type, Destination Address, Source Address, and Data Length octets. Accumulate each octet into HeaderCRC. As each octet is transmitted, set SilenceTimer to zero.
(6) Transmit the ones-complement of HeaderCRC. Set SilenceTimer to zero.
(7) If there are data octets, initialize DataCRC to X'FFFF'.
(8) Transmit any data octets. Accumulate each octet into DataCRC. As each octet is transmitted, set SilenceTimer to zero.
(9) Transmit the ones-complement of DataCRC, least significant octet first. As each octet is transmitted, set SilenceTimer to zero.
(10) Wait until the final stop bit of the most significant CRC octet has been transmitted but not more than $\mathbf{T}_{\mathbf{postdrive}}$.
(11) Disable the transmit line driver and enable the receiver.
(12) Return.

9.5.5.2 SendCOBS_EncodedFrame Procedure for COBS-Encoded Frame Types

This procedure shall be implemented by devices that support COBS-encoded Frame Types.

(1) COBS-encode the NPDU according to Clause 9.10.2 to generate the Encoded Data field. Set DataLength equal to the length of the Encoded Data field in octets, plus three.
(2) If SilenceTimer is less than $\mathbf{T}_{\mathbf{turnaround}}$, wait ($\mathbf{T}_{\mathbf{turnaround}}$ - SilenceTimer).
(3) Disable the receiver and enable the transmit line driver.
(4) Transmit the preamble octets X'55', X'FF'. As each octet is transmitted, set SilenceTimer to zero.
(5) Initialize HeaderCRC to X'FF'.
(6) Transmit the Frame Type, Destination Address, Source Address, and Data Length octets. Accumulate each octet into HeaderCRC. As each octet is transmitted, set SilenceTimer to zero.
(7) Transmit the ones-complement of HeaderCRC. Set SilenceTimer to zero.
(8) Initialize CRC32K to X'FFFFFFFF'.
(9) Transmit the Encoded Data octets. Accumulate each octet into CRC32K. As each octet is transmitted, set SilenceTimer to zero.
(10) Compute the ones-complement of CRC32K and order it least significant octet first. COBS-encode the four octets according to Clause 9.10.2 to generate the five-octet Encoded CRC-32K field.
(11) Transmit the Encoded CRC-32K octets. As each octet is transmitted, set SilenceTimer to zero.
(12) Wait until the final stop bit of the last Encoded CRC-32K octet has been transmitted but not more than $\mathbf{T}_{\mathbf{postdrive}}$.
(13) Disable the transmit line driver and enable the receiver.
(14) Return.

9.5.6 Master Node Finite State Machine

The description of operation is as a finite state machine. Figure 9-4 shows the Master Node state machine, which is described fully in this clause. Each state is given a name, specified in all capital letters. Transitions are also named, in mixed upper- and lowercase letters. Transitions are described as a series of conditions followed by a series of actions to be taken if the conditions are met. The final action in each transition is entry into a new state, which may be the same as the current state.

A master node that supports segmentation shall respond to each BACnet Data Expecting Reply frame with either a Reply Postponed frame or a data frame. This response releases the MS/TP Master Node State Machine that is holding the token from the WAIT_FOR_REPLY state and allows it to send the next MS/TP frame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

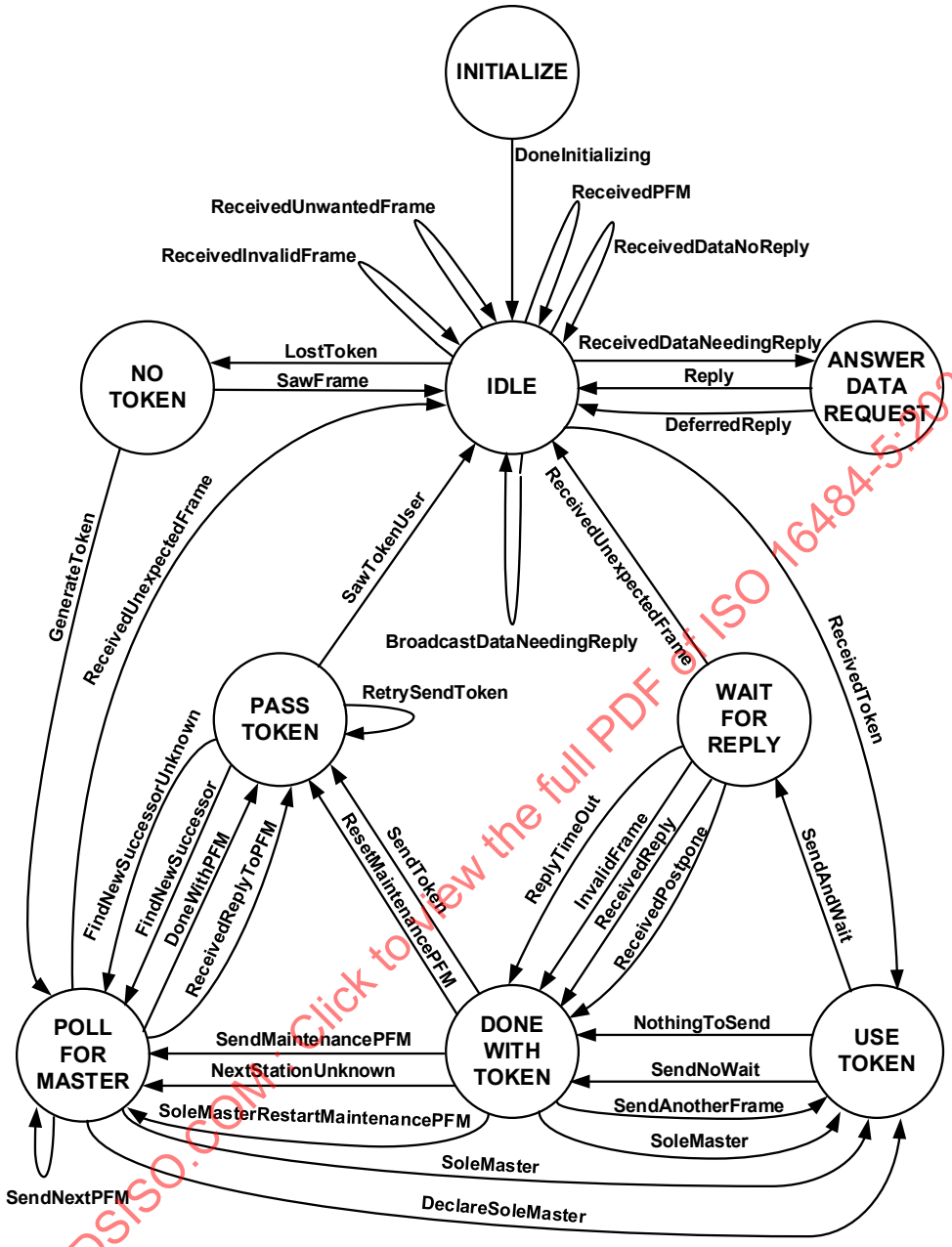


Figure 9-4. Master Node State Machine.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

9.5.6.1 INITIALIZE

When a master node is powered up or reset, it shall unconditionally enter the INITIALIZE state.

DoneInitializing
Unconditionally,

set TS to the node's station address, set NS equal to TS (indicating that the next station is unknown), set PS equal to TS, set TokenCount to $\mathbf{N_{poll}}$ (thus causing a Poll For Master to be sent when this node first receives the token), set SoleMaster to FALSE, set ReceivedValidFrame and ReceivedInvalidFrame to FALSE, and enter the IDLE state.

9.5.6.2 IDLE

In the IDLE state, the node waits for a frame.

© ISO 2022 – All rights reserved

LostToken

If SilenceTimer is greater than or equal to $\mathbf{T}_{\mathbf{no_token}}$,

then assume that the token has been lost. Set EventCount to zero and enter the NO_TOKEN state.

ReceivedInvalidFrame

If ReceivedInvalidFrame is TRUE,

then an invalid frame was received. Set ReceivedInvalidFrame to FALSE, and enter the IDLE state to wait for the next frame.

ReceivedUnwantedFrame

If ReceivedValidFrame is TRUE and either

(a) DestinationAddress is not equal to either TS (this station) or 255 (broadcast) or
(b) DestinationAddress is equal to 255 (broadcast) and FrameType has a value of Token, Test_Request, or a proprietary type known to this node that expects a reply (such frames may not be broadcast) or
(c) FrameType has a value that indicates a standard or proprietary type that is not known to this node or
(d) DestinationAddress is equal to TS and FrameType is equal to Reply To Poll For Master or Reply Postponed,

then an unexpected or unwanted frame was received. Set ReceivedValidFrame to FALSE, and enter the IDLE state to wait for the next frame.

ReceivedToken

If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to Token,

then set ReceivedValidFrame to FALSE; set FrameCount to zero; set SoleMaster to FALSE; and enter the USE_TOKEN state.

ReceivedPFM

If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to Poll For Master,

then call SendFrame to transmit a Reply To Poll For Master frame to the node whose address is specified by SourceAddress (Source Address of the Poll); set ReceivedValidFrame to FALSE; and enter the IDLE state.

ReceivedDataNoReply

If ReceivedValidFrame is TRUE and DestinationAddress is equal to either TS (this station) or 255 (broadcast) and FrameType is equal to BACnet Data Not Expecting Reply, Test_Response, BACnet Extended Data Not Expecting Reply, or a FrameType known to this node that does not expect a reply,

then indicate successful reception to the higher layers; set ReceivedValidFrame to FALSE; and enter the IDLE state.

ReceivedDataNeedingReply

If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to BACnet Data Expecting Reply, Test_Request, BACnet Extended Data Expecting Reply, or a FrameType known to this node that expects a reply,

then indicate successful reception to the higher layers (management entity in the case of Test_Request); set ReceivedValidFrame to FALSE; and enter the ANSWER_DATA_REQUEST state.

BroadcastDataNeedingReply

If ReceivedValidFrame is TRUE and DestinationAddress is equal to 255 (broadcast) and FrameType is equal to either BACnet Data Expecting Reply or BACnet Extended Data Expecting Reply,

then indicate successful reception to the higher layers; set ReceivedValidFrame to FALSE; and enter the IDLE state to wait for the next frame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.5.6.3 USE_TOKEN

In the USE_TOKEN state, the node is allowed to send one or more data frames. These may be BACnet Data frames or other standard or proprietary frames.

NothingToSend
: If there is no data frame awaiting transmission,

  then set FrameCount to $\mathbf{N}_{\mathbf{max_info_frames}}$ and enter the DONE_WITH_TOKEN state.

SendNoWait
: If the next frame awaiting transmission is of type Test_Response, BACnet Data Not Expecting Reply, BACnet Extended Data Not Expecting Reply, a FrameType known to this node that does not expect a reply, or has a DestinationAddress that is equal to 255 (broadcast) and a FrameType equal to either BACnet Data Expecting Reply or BACnet Extended Data Expecting Reply,

  then call SendFrame to transmit the frame; increment FrameCount; and enter the DONE_WITH_TOKEN state.

SendAndWait
: If the next frame awaiting transmission is of type Test_Request, a FrameType known to this node that expects a reply, or has a DestinationAddress that is not equal to 255 (broadcast) and a FrameType equal to either BACnet Data Expecting Reply or BACnet Extended Data Expecting Reply,

  then call SendFrame to transmit the data frame; increment FrameCount; and enter the WAIT_FOR_REPLY state.

9.5.6.4 WAIT_FOR_REPLY

In the WAIT_FOR_REPLY state, the node waits for a reply from another node.

ReplyTimeout
: If SilenceTimer is greater than or equal to $\mathbf{T}_{\mathbf{reply_timeout}}$,

  then assume that the request has failed. Set FrameCount to $\mathbf{N}_{\mathbf{max_info_frames}}$ and enter the DONE_WITH_TOKEN state. Any retry of the data frame shall await the next entry to the USE_TOKEN state. (Because of the length of the timeout, this transition will cause the token to be passed regardless of the initial value of FrameCount.)

InvalidFrame
: If SilenceTimer is less than $\mathbf{T}_{\mathbf{reply_timeout}}$ and ReceivedInvalidFrame is TRUE,

  then there was an error in frame reception. Set ReceivedInvalidFrame to FALSE and enter the DONE_WITH_TOKEN state.

ReceivedReply
: If SilenceTimer is less than $\mathbf{T}_{\mathbf{reply_timeout}}$ and ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to Test_Response, BACnet Data Not Expecting Reply, BACnet Extended Data Not Expecting Reply, or a FrameType known to this node that indicates a reply,

  then indicate successful reception to the higher layers; set ReceivedValidFrame to FALSE; and enter the DONE_WITH_TOKEN state.

ReceivedPostpone
: If SilenceTimer is less than $\mathbf{T}_{\mathbf{reply_timeout}}$ and ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to Reply Postponed,

  then the reply to the message has been postponed until a later time. Set ReceivedValidFrame to FALSE and enter the DONE_WITH_TOKEN state.

ReceivedUnexpectedFrame
: If SilenceTimer is less than $\mathbf{T}_{\mathbf{reply_timeout}}$ and ReceivedValidFrame is TRUE and either

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

(a) DestinationAddress is not equal to TS (the expected reply should not be broadcast) or
(b) FrameType has a value other than Test_Response, BACnet Data Not Expecting Reply, BACnet Extended Data Not Expecting Reply, Reply Postponed or a FrameType known to this node that indicates a reply,

then an unexpected frame was received. This may indicate the presence of multiple tokens. Set ReceivedValidFrame to FALSE, and enter the IDLE state to synchronize with the network. This action drops the token.

9.5.6.5 DONE_WITH_TOKEN

The DONE_WITH_TOKEN state either sends another data frame, passes the token, or initiates a Poll For Master cycle.

SendAnotherFrame
If FrameCount is less than $\mathbf{N}_{max_info_frames}$,

then this node may send another information frame before passing the token. Enter the USE_TOKEN state.

NextStationUnknown
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$, SoleMaster is FALSE and NS is equal to TS,

then the next station to which the token should be sent is unknown. Set PS to (TS+1) modulo ($\mathbf{N}_{max_master}$+1); call SendFrame to transmit a Poll For Master frame to PS; set RetryCount to zero, and enter the POLL_FOR_MASTER state.

SoleMaster
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$ and TokenCount is less than $\mathbf{N}_{poll}$ and SoleMaster is TRUE,

then there are no other known master nodes to which the token may be sent (true master-slave operation). Set FrameCount to zero, increment TokenCount, and enter the USE_TOKEN state.

SendToken
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$ and TokenCount is less than $\mathbf{N}_{poll}$ and SoleMaster is FALSE, or if NS is equal to (TS+1) modulo ($\mathbf{N}_{max_master}$+1),

then increment TokenCount; call SendFrame to transmit a Token frame to NS; set RetryCount and EventCount to zero; and enter the PASS_TOKEN state. (The comparison of NS and TS+1 eliminates the Poll For Master if there are no addresses between TS and NS, since there is no address at which a new master node may be found in that case).

SendMaintenancePFM
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$ and TokenCount is greater than or equal to $\mathbf{N}_{poll}$ and (PS+1) modulo ($\mathbf{N}_{max_master}$+1) is not equal to NS,

then set PS to (PS+1) modulo ($\mathbf{N}_{max_master}$+1); call SendFrame to transmit a Poll For Master frame to PS; set RetryCount to zero; and enter the POLL_FOR_MASTER state.

ResetMaintenancePFM
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$ and TokenCount is greater than or equal to $\mathbf{N}_{poll}$ and (PS+1) modulo ($\mathbf{N}_{max_master}$+1) is equal to NS, and SoleMaster is FALSE,

then set PS to TS; call SendFrame to transmit a Token frame to NS; set RetryCount and EventCount to zero; set TokenCount to one; and enter the PASS_TOKEN state.

SoleMasterRestartMaintenancePFM
If FrameCount is greater than or equal to $\mathbf{N}_{max_info_frames}$, and TokenCount is greater than or equal to $\mathbf{N}_{poll}$, and (PS+1) modulo ($\mathbf{N}_{max_master}$+1) is equal to NS, and SoleMaster is TRUE,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then set PS to (NS +1) modulo (N_{max_master}+1) ; call SendFrame to transmit a Poll For Master to PS; set NS to TS (no known successor node); set RetryCount to zero; set TokenCount to one; and enter the POLL_FOR_MASTER state to find a new successor to TS.

9.5.6.6 PASS_TOKEN

The PASS_TOKEN state listens for a successor to begin using the token that this node has just attempted to pass.

SawTokenUser
If SilenceTimer is less than $T_{usage_timeout}$ and EventCount is greater than N_{min_octets},

then assume that a frame has been sent by the new token user. Enter the IDLE state to process the frame.

RetrySendToken
If SilenceTimer is greater than or equal to $T_{usage_timeout}$ and RetryCount is less than N_{retry_token},

then increment RetryCount; call SendFrame to transmit a Token frame to NS; set EventCount to zero; and re-enter the current state to listen for NS to begin using the token.

FindNewSuccessorUnknown
If SilenceTimer is greater than or equal to $T_{usage_timeout}$ and RetryCount is greater than or equal to N_{retry_token}, and (NS+1) modulo (N_{max_master}+1) is equal to TS,

then assume that NS has failed. Set PS to (TS+1) modulo (N_{max_master}+1); call SendFrame to transmit a Poll For Master frame to PS; set NS to TS (no known successor node); set RetryCount and TokenCount to zero; and enter the POLL_FOR_MASTER state to find a new successor to TS.

FindNewSuccessor
If SilenceTimer is greater than or equal to $T_{usage_timeout}$ and RetryCount is greater than or equal to N_{retry_token},

then assume that NS has failed. Set PS to (NS+1) modulo (N_{max_master}+1); call SendFrame to transmit a Poll For Master frame to PS; set NS to TS (no known successor node); set RetryCount and TokenCount to zero; and enter the POLL_FOR_MASTER state to find a new successor to TS.

9.5.6.7 NO_TOKEN

The NO_TOKEN state is entered if SilenceTimer becomes greater than Tno_token, indicating that there has been no network activity for that period of time. The timeout is continued to determine whether or not this node may create a token.

SawFrame
If SilenceTimer is less than T_{no_token}+(T_{slot}*TS) and EventCount is greater than N_{min_octets},

then some other node exists at a lower address. Enter the IDLE state to receive and process the incoming frame.

GenerateToken
If SilenceTimer is greater than or equal to T_{no_token}+(T_{slot}*TS) and SilenceTimer is less than T_{no_token}+(T_{slot}*(TS+1)),

then assume that this node is the lowest numerical address on the network and is empowered to create a token. Set PS to (TS+1) modulo (N_{max_master}+1); call SendFrame to transmit a Poll For Master frame to PS; set NS to TS (indicating that the next station is unknown); set RetryCount and TokenCount to zero; and enter the POLL_FOR_MASTER state to find a new successor to TS.

9.5.6.8 POLL_FOR_MASTER

In the POLL_FOR_MASTER state, the node listens for a reply to a previously sent Poll For Master frame in order to find a successor node.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ReceivedReplyToPFM

If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to Reply To Poll For Master,

then set SoleMaster to FALSE; set NS equal to SourceAddress; set EventCount to zero; call SendFrame to transmit a Token frame to NS; set PS to the value of TS; set TokenCount and RetryCount to zero; set ReceivedValidFrame to FALSE; and enter the PASS_TOKEN state.

ReceivedUnexpectedFrame

If ReceivedValidFrame is TRUE and either

(a) DestinationAddress is not equal to TS or
(b) FrameType is not equal to Reply To Poll For Master,

then an unexpected frame was received. This may indicate the presence of multiple tokens. Set ReceivedValidFrame to FALSE and enter the IDLE state to synchronize with the network. This action drops the token.

SoleMaster

If SoleMaster is TRUE and either

(a) SilenceTimer is greater than or equal to $\mathbf{T}_{\text{usage_timeout}}$ or
(b) ReceivedInvalidFrame is TRUE,

then there was no valid reply to the periodic poll by the sole known master for other masters. Set FrameCount to zero, set ReceivedInvalidFrame to FALSE, and enter the USE_TOKEN state.

DoneWithPFM

If SoleMaster is FALSE and NS is not equal to TS and either:

(a) SilenceTimer is greater than or equal to $\mathbf{T}_{\text{usage_timeout}}$ or
(b) ReceivedInvalidFrame is TRUE,

then there was no valid reply to the maintenance poll for a master at address PS. Set EventCount to zero; call SendFrame to transmit a Token frame to NS; set RetryCount to zero; set ReceivedInvalidFrame to FALSE; and enter the PASS_TOKEN state.

SendNextPFM

If SoleMaster is FALSE and NS is equal to TS (no known successor node) and (PS+1) modulo ($\mathbf{N}_{\text{max_master}}$+1) is not equal to TS and either:

(a) SilenceTimer greater than or equal to $\mathbf{T}_{\text{usage_timeout}}$ or
(b) ReceivedInvalidFrame is TRUE,

then set PS to (PS+1) modulo ($\mathbf{N}_{\text{max_master}}$+1); call SendFrame to transmit a Poll For Master frame to PS; set RetryCount to zero; set ReceivedInvalidFrame to FALSE; and re-enter the current state.

DeclareSoleMaster

If SoleMaster is FALSE and NS is equal to TS (no known successor node) and (PS+1) modulo ($\mathbf{N}_{\text{max_master}}$+1) is equal to TS and either

(a) SilenceTimer is greater than or equal to $\mathbf{T}_{\text{usage_timeout}}$ or
(b) ReceivedInvalidFrame is TRUE,

then set SoleMaster TRUE to indicate that this station is the only master; set FrameCount to zero; set ReceivedInvalidFrame to FALSE; and enter the USE_TOKEN state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

9.5.6.9 ANSWER_DATA_REQUEST

The ANSWER_DATA_REQUEST state is entered when a BACnet Data Expecting Reply, BACnet Extended Data Expecting Reply, a Test_Request, or a FrameType known to this node that expects a reply is received.

Reply

If a reply is available from the higher layers within $\mathbf{T}_{\text{reply_delay}}$ after the reception of the final octet of the requesting frame (the mechanism used to determine this is a local matter),

then call SendFrame to transmit the reply frame and enter the IDLE state to wait for the next frame.

DeferredReply

If no reply will be available from the higher layers within $\mathbf{T}_{\text{reply_delay}}$ after the reception of the final octet of the requesting frame (the mechanism used to determine this is a local matter)), or after a DL-RELEASE.request is received,

then an immediate reply is not possible. Any reply shall wait until this node receives the token. Call SendFrame to transmit a Reply Postponed frame, and enter the IDLE state.

9.5.7 Slave Node Finite State Machine

The state machine for a slave node is similar to, but considerably simpler than, that for a master node. A slave node shall neither transmit nor receive segmented messages. If a slave node receives a segmented BACnet-Confirmed-Request-PDU, the node shall respond with a BACnet-Abort-PDU specifying abort-reason "segmentation not supported." Figure 9-5 shows the Slave Node state machine, which is described fully in the following text.

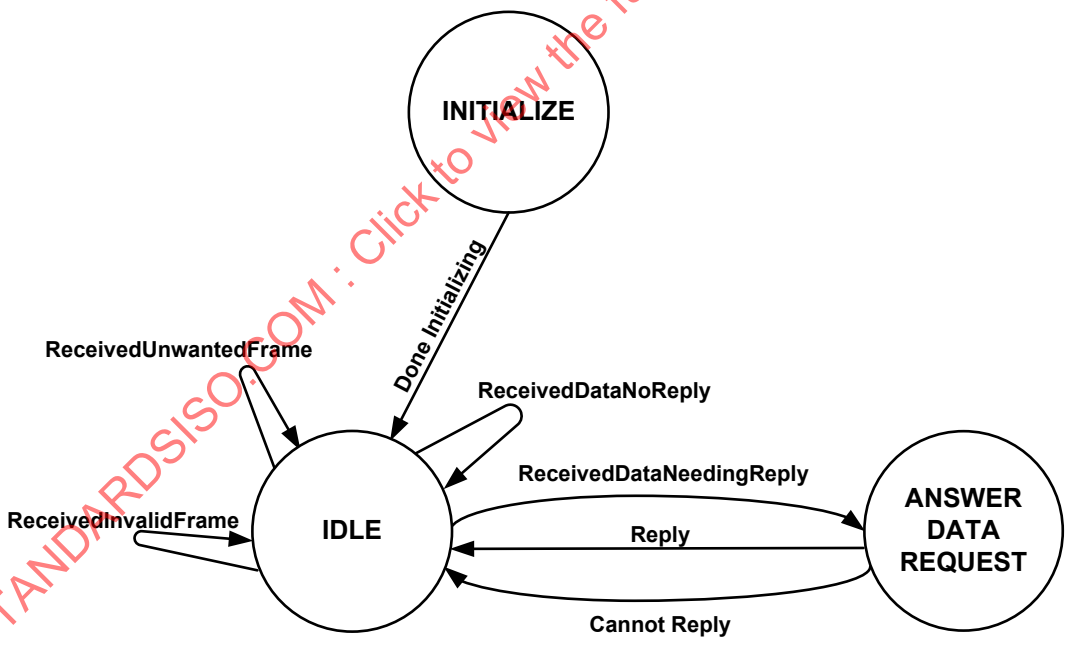


Figure 9-5. Slave Node State Machine

9.5.7.1 INITIALIZE

When a slave node is powered up or reset, it shall unconditionally enter the INITIALIZE state.

DoneInitializing

Unconditionally,

set TS to the node's station address; set ReceivedValidFrame and ReceivedInvalidFrame to FALSE; and enter the IDLE state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

9.5.7.2 IDLE

In the IDLE state, the node waits for a frame.

ReceivedInvalidFrame
If ReceivedInvalidFrame is TRUE,

then an invalid frame was received. Set ReceivedInvalidFrame to FALSE, and enter the IDLE state to wait for the next frame.

ReceivedUnwantedFrame
If ReceivedValidFrame is TRUE and either

(a) DestinationAddress is not equal to either TS (this station) or 255 (broadcast) or
(b) DestinationAddress is equal to 255 (broadcast) and FrameType has a value of BACnet Data Expecting Reply, Test_Request, or a FrameType known to this node that expects a reply (such frames may not be broadcast) or
(c) FrameType has a value of Token, Poll For Master, Reply To Poll For Master, Reply Postponed, or a FrameType not known to this node,

then an unexpected or unwanted frame was received. Set ReceivedValidFrame to FALSE, and enter the IDLE state to wait for the next frame.

ReceivedDataNoReply
If ReceivedValidFrame is TRUE and DestinationAddress is equal to either TS (this station) or 255 (broadcast) and FrameType is equal to BACnet Data Not Expecting Reply, Test_Response, or a FrameType known to this node that does not expect a reply,

then indicate successful reception to the higher layers, set ReceivedValidFrame to FALSE, and enter the IDLE state.

ReceivedDataNeedingReply
If ReceivedValidFrame is TRUE and DestinationAddress is equal to TS (this station) and FrameType is equal to BACnet Data Expecting Reply, Test_Request, or a FrameType known to this node that expects a reply,

then indicate successful reception to the higher layers (management entity in the case of Test_Request), set ReceivedValidFrame to FALSE, and enter the ANSWER_DATA_REQUEST state.

9.5.7.3 ANSWER_DATA_REQUEST

The ANSWER_DATA_REQUEST state is entered when a BACnet Data Expecting Reply, a Test_Request, or a FrameType known to this node that expects a reply is received.

Reply
If a reply is available from the higher layers within $\mathbf{T}_{\mathbf{reply_delay}}$ after the reception of the final octet of the requesting frame (the mechanism used to determine this is a local matter),

then call SendFrame to transmit the reply frame, and enter the IDLE state to wait for the next frame.

CannotReply
If no reply will be available from the higher layers within $\mathbf{T}_{\mathbf{reply_delay}}$ after the reception of the final octet of the requesting frame (the mechanism used to determine this is a local matter),

then no reply is possible. Enter the IDLE state.

9.5.8 The CheckHeader Procedure

This procedure checks for invalid header fields and sets the value of GoodHeader accordingly.

If the value of HeaderCRC is not X '55',
or the value of SourceAddress is 255 (broadcast),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

or the value of FrameType is less than $\mathbf{N}_{\mathbf{min_COBS_type}}$ and DataLength is greater than 501,
or the value of FrameType is greater than $\mathbf{N}_{\mathbf{max_COBS_type}}$ and DataLength is greater than 501,
or the value of FrameType is greater than or equal to $\mathbf{N}_{\mathbf{min_COBS_type}}$ and
the value of FrameType is less than or equal to $\mathbf{N}_{\mathbf{max_COBS_type}}$ and
DataLength is less than $\mathbf{N}_{\mathbf{min_COBS_length}}$,
or the value of FrameType is greater than or equal to $\mathbf{N}_{\mathbf{min_COBS_type}}$ and
the value of FrameType is less than or equal to $\mathbf{N}_{\mathbf{max_COBS_type}}$ and
DataLength is greater than $\mathbf{N}_{\mathbf{max_COBS_length}}$,

then set GoodHeader to FALSE, else set GoodHeader to TRUE.

9.6 Cyclic Redundancy Check (CRC)

MS/TP uses Cyclic Redundancy Checks (CRC) to provide error-detection. CRCs have a number of advantages over simpler error detection methods, such as parity or checksums, which are commonly used with UART-based networks. The major drawbacks of parity are that it adds one bit of overhead to each transmitted octet and that it will not detect an even number of errors within one octet. Exclusive OR checksums, sometimes called longitudinal parity, offer reduced overhead over octet parity but suffer from the same inability to detect an even number of errors in a given bit position. Additive checksums are similar but the exact error detection characteristics are dependent on bit position.

Ethernet, ARCNET, and many other standard and proprietary communications systems use more robust CRCs. Mathematically, a CRC is the remainder that results when a data stream (such as a frame) taken as a binary number is divided modulo two by a generator polynomial. The proof of the error-detecting properties of the CRC and the selection of appropriate polynomials are beyond the scope of this document. Annex G describes the implementation of the CRC algorithms in software.

9.6.1 Frame Header CRC

The MS/TP frame header CRC uses the polynomial

$$G(X) = X^8 + X^7 + 1$$

In operation, at the transmitter, the initial content of the CRC register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Frame Type, Destination Address, Source Address, and Length fields. The ones-complement of the resulting remainder is transmitted as the 8-bit Header CRC.

At the receiver, the initial content of the CRC register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Frame Type, Destination Address, Source Address, Length, and Header CRC fields of the incoming message. In the absence of transmission errors, the resultant remainder will be:

0101 0101 (x^0 through x^7, respectively).

9.6.2 Data CRC

The MS/TP Data CRC uses the CRC-16-CCITT polynomial

$$G(X) = X^{16} + X^{12} + X^5 + 1$$

In operation, at the transmitter, the initial content of the CRC register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Data field. The ones-complement of the resulting remainder is transmitted, least significant octet first, as the 16 bit Data CRC.

At the receiver, the initial content of the CRC register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Data and Data CRC fields of the incoming message. In the absence of transmission errors, the resultant remainder will be

1111 0000 1011 1000 (x^0 through x^{15}, respectively).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NOTE: The initialization of the CRC register to all ones and the complementing of the register before transmission prevent the CRC from having a value of zero if the covered field is all zeros.

9.6.3 CRC-32K

COBS-encoded MS/TP frames use the CRC-32K (Koopman) polynomial

$$G(X) = X^{32} + X^{30} + X^{29} + X^{28} + X^{26} + X^{20} + X^{19} + X^{17} + X^{16} + X^{15} + X^{11} + X^{10} + X^{7} + X^{6} + X^{4} + X^{2} + X + 1$$
$$= (X + 1)(X^{3} + X^{2} + 1)(X^{28} + X^{22} + X^{20} + X^{19} + X^{16} + X^{14} + X^{12} + X^{9} + X^{8} + X^{6} + 1)$$

In operation, at the transmitter, the initial content of the CRC-32K register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Encoded Data field (see Clause 9.10.2). The ones-complement of the resulting remainder is ordered least-significant octet first and then COBS-encoded to generate the five-octet Encoded CRC-32K field.

At the receiver, the initial content of the CRC-32K register of the device computing the remainder of the division is preset to all ones. The register is then modified by division by the generator polynomial G(x) of the Encoded Data field octets of the incoming message before they are decoded. The Encoded CRC-32K field is then decoded (see Clause 9.10.3) and the register is modified by division by the generator polynomial G(x) of the four decoded octets. In the absence of transmission errors, the resultant remainder will be

0000 1000 0100 0011 0011 0010 0011 1011 (x^0 through x^{31}, respectively).

NOTE: The initialization of the CRC-32K register to all ones and the complementing of the register before transmission prevent the CRC-32K from having a value of zero if the covered field is all zeros.

9.7 Interfacing MS/TP LANs with Other BACnet LANs

9.7.1 Routing of BACnet Messages from MS/TP

When a BACnet network entity with routing capability receives from a directly connected MS/TP data link an NPDU whose 'data_expecting_reply' parameter is TRUE and the NPDU is to be routed to another BACnet device according to the procedures of Clause 6, the network entity shall direct the MS/TP data link to transmit a Reply Postponed frame by issuing a DL-RELEASE.request before attempting to route the NPDU. This allows the routing node to leave the ANSWER_DATA_REQUEST state and the sending node to leave the WAIT_FOR_REPLY state before the potentially lengthy process of routing the NPDU is begun.

BACnet routers directly connected to MS/TP links must be updated to support COBS-encoded BACnet MS/TP frame types (i.e. BACnet Extended Data Expecting Reply and BACnet Extended Data Not Expecting Reply) before any non-routing devices installed on the link can utilize COBS-encoded frames. This does not, however, ensure that all links on the path from a given source to a given destination network have been updated. Before sending large MS/TP frames, a sending device should determine whether the given destination network is reachable using large MS/TP frames. A procedure by which this can be achieved is described in Clause 19.4.

9.7.2 Routing of BACnet Messages to MS/TP

When a BACnet network entity issues a DL-UNITDATA.request to a directly connected MS/TP data link, it shall set the 'data_expecting_reply' parameter of the DL-UNITDATA.request equal to the value of the 'data_expecting_reply' parameter of the network protocol control information of the NPDU, which is transferred in the 'data' parameter of the request.

9.8 Responding BACnet User Processing of Messages from MS/TP

In Clause 5.4.5.3, AWAIT_RESPONSE, the following transition shall be added:

PostponeReply

If a CONF_SERV.response will not be received from the local application layer early enough that a reply MS/TP frame would be received by the remote node within $\mathbf{T}_{\text{reply_timeout}}$ (defined in Clause 9.5.3) after the transmission of the original BACnet-Confirmed-Request-PDU (the means of this determination are a local matter),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then direct the MS/TP data link to transmit a Reply Postponed frame and enter the AWAIT_RESPONSE state.

In Clause 5.4.5.3, AWAIT_RESPONSE, in the transition SendSegmentedComplexACK, the text "transmit a BACnet-ComplexACK-PDU..." shall be replaced by "direct the MS/TP data link to transmit a Reply Postponed frame; transmit a BACnet-ComplexACK-PDU...." (It is necessary to postpone the reply because transmission of the segmented ComplexACK cannot begin until the node holds the token.)

9.9 Repeaters

If any of the limits in Clauses 9.2.1 and 9.2.2 are exceeded, one or more repeaters is required. An MS/TP EIA-485 Repeater is defined as an active device that provides selective interconnection between two or more segments of MS/TP cable. The repeater contains logic that detects and passes signals received from one segment onto all other segments. The segment from which signals are received is determined according to a priority algorithm.

The method used by a repeater to detect signals and to distinguish them from noise is a local matter, subject to the following constraints:

(a) The repeater may not lengthen or shorten the duration of any bit of the output data stream by more than 2% relative to the input data stream.
(b) The repeater may not delay the output data stream by more than two bit times relative to the input data stream.

No more than 10 bit times of delay shall exist in the path between any two nodes of an MS/TP network. This corresponds to five repeaters with worst-case delays (if delay by the medium is negligible, as it will be at all except the highest baud rates) or a greater number of repeaters with smaller delays.

The minimum value of repeater turnoff delay $\mathbf{T_{roff}}$ is dictated by the maximum amount of idle line allowed during a single frame, $\mathbf{T_{frame_gap}}$. If the value of the octet immediately preceding the idle is X'FF', then there may be up to $9+\mathbf{T_{frame_gap}} = 29$ bit times of one state between zero (start) bits. Thus, $\mathbf{T_{roff}}$ must be larger than 29 bit times if it is not to turn off during the transmission of a frame.

In order to avoid contention between repeater turnoff and the beginning of the next frame, the repeater turnoff delay $\mathbf{T_{roff}}$ may be no larger than $\mathbf{T_{turnaround}}$ bit times, and a node may not enable its driver until a minimum of $\mathbf{T_{turnaround}}$ after the end of the previous frame. Thus

29 bit times < $\mathbf{T_{roff}}$ < 40 bit times.

An N-way repeater may be represented by an N+1 state finite state machine. One state is the IDLE state, and the others each receive from one segment and re-transmit on all other segments. The repeater state machine uses the timer PortIdle to control the return to the IDLE state. PortIdle shall have a resolution of one bit time or finer.

9.9.1 IDLE

In the IDLE state, all receivers are enabled and all transmitters are disabled. The repeater remains in the IDLE state until a logical zero (i.e. the beginning of a start bit) is detected on some port.

When a zero is detected, the repeater disables all receivers, except the one on which the zero was detected, and enables all transmitters, except the one associated with the port on which the zero was detected. The repeater then enters the state associated with the active receiver port. If a zero is detected simultaneously on more than one port, the method used to arbitrate between them is a local matter.

Port1Active
If a zero is detected on Port 1,

then disable all receivers except Port 1; enable all transmitters except Port 1; pass the zero to all enabled transmitters; set PortIdle to zero; and enter the PORT_1_ACTIVE state.

Port2Active
If a zero is detected on Port 2,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

then disable all receivers except Port 2; enable all transmitters except Port 2; pass the zero to all enabled transmitters; set PortIdle to zero; and enter the PORT_2_ACTIVE state.

This may be extended to as many ports as desired simply by adding transitions and PORT_N_ACTIVE states.

9.9.2 PORT_i_ACTIVE

In the PORT_i_ACTIVE state, the Repeater passes signals from Port i to all other ports. The Repeater will remain in this state until Port i becomes idle. Idleness is defined as the absence of the logical zero state at Port i for more than $\mathbf{T_{roff}}$ bit times.

When idleness is detected, all transmitters are disabled, all receivers are enabled, and the Repeater enters the IDLE state to await renewed activity.

PortActive0
If a zero is detected on Port i,

then pass the zero to all other ports, set PortIdle to zero, and re-enter the current state.

PortActive1
If PortIdle is less than $\mathbf{T_{roff}}$ and a one is detected at Port i,

then pass the one to all other ports and re-enter the current state.

PortInactive
If PortIdle is greater than or equal to $\mathbf{T_{roff}}$,

then disable all transmitters, enable all receivers, and enter the IDLE state.

9.10 COBS (Consistent Overhead Byte Stuffing) Encoding

The original MS/TP specification did not prevent preamble sequences from appearing in the Data or Data CRC fields. This resulted in lost synchronization or dropped frames under certain circumstances. The Receive Frame state machine was later revised to mitigate this problem, but many deployed MS/TP devices do not have the update.

In order to decrease the likelihood of compatibility problems in environments with a mix of new and legacy devices, all MS/TP devices that support COBS-encoded frames shall COBS-encode the Encoded Data and Encoded CRC-32K fields before transmission and decode these fields upon reception. The result of this encoding is to remove X'55' octets from these portions of the frame, therefore preamble sequences cannot appear "on the wire" except where intended to indicate the beginning of a frame.

9.10.1 COBS Description

This clause provides a basic description of the COBS encoding excerpted from "Consistent Overhead Byte Stuffing" by S. Cheshire and M. Baker. A reference to this paper appears in Clause 25.

COBS performs a reversible transformation on a data field to eliminate a single octet value (e.g., X'55') from it. Once eliminated from the data, that octet value may then be safely used in the framing marker (preamble). The most efficient way to accomplish this is to first eliminate all X'00' (zero) octets as described below, then exclusive OR (XOR) the output with the octet selected for removal.

COBS first takes its input data and logically appends a single zero octet. It then locates all the zero octets in the frame (including the appended one) and then divides the frame at these boundaries into one or more zero-terminated chunks. Every zero-terminated chunk contains exactly one zero octet, and that zero is always the last octet of the chunk. A chunk may be as short as one octet (i.e. a chunk containing a single zero octet) or as long as the entire input sequence (i.e. no zeros in the data except the appended one).

COBS encodes each zero-terminated chunk using one or more variable length COBS code blocks. Chunks of 254 octets or less (counting the terminating zero) are encoded as single COBS code blocks. Chunks longer than 254 octets are encoded using multiple code blocks, as described later in this clause. After a packet's constituent chunks have all been

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

encoded using COBS code blocks and concatenated, the resulting aggregate block of data is completely free of zero octets.

A COBS code block consists of a single code octet, followed by zero or more data octets. The number of data octets is represented by the code octet. For codes X'01' to X'FE', the meaning of each code block is that it represents the sequence of data octets contained within the code block, followed by an implicit zero octet. The zero octet is implicit, meaning it is not actually contained within the sequence of data octets in the code block, but it is counted.

These code blocks encode data without adding any overhead. Each code block begins with one code octet followed by n data octets, and represents n data octets followed by one trailing zero octet. Thus the code block adds no overhead to the data: a chunk (n+1) octets long is encoded using a code block (1+n) octets long. These basic codes are suitable for encoding zero-terminated chunks up to 254 octets in length, but some of the zero-terminated chunks that make up a packet may be longer than that. To encode these chunks, code X'FF' is defined slightly differently. Code X'FF' represents the sequence of 254 data octets contained within the code block, without any implicit zero. This encoding is illustrated in Figure 9-6 below.

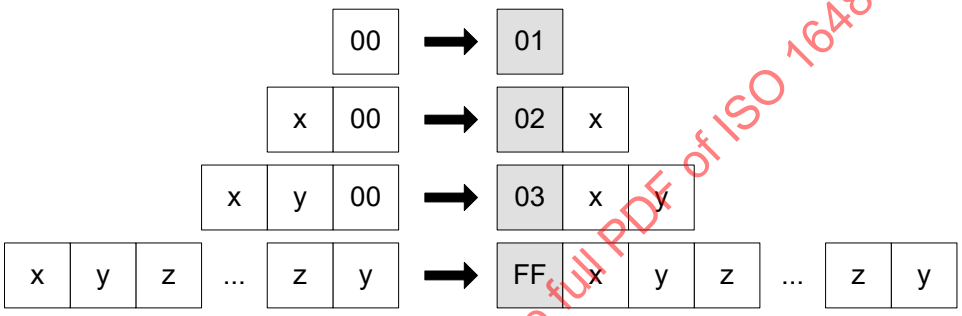


Figure 9-6. COBS Encoding of Chunks into Code Blocks

The two examples in Figure 9-7 show the effect of logically appending a zero (X'00') octet to the input data before encoding. For codes less than 255, a "phantom zero" after the data is used to differentiate between final chunks that end in X'00' and those that do not.

In the upper example, the final chunk of data is the string "Hello" without a terminating zero. A phantom zero is appended to the data and this results in consistent encoding of the final chunk. The phantom zero is discarded by the COBS decoder upon reception and the original data is restored in the receive buffer.

In the lower example, the final chunk of data is again the string "Hello" but this is terminated by a null in the data. Again, a phantom zero is appended to the data. Note the difference after encoding; the result is actually two code blocks transmitted. At the receiver, the phantom zero is discarded and the original data is restored, including the terminating null.

In the case of a final chunk that consists of exactly 254 non-zero octets, a phantom zero SHALL NOT be appended to the encoding since the meaning of code 255 is unambiguous: no implicit zero follows this data.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

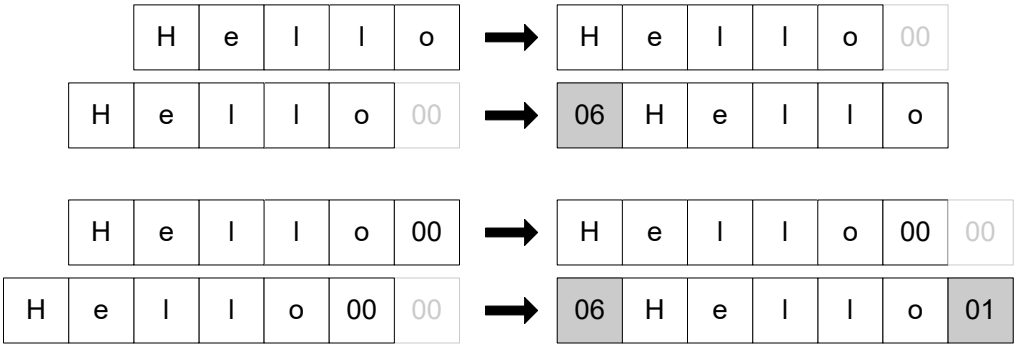


Figure 9-7. Effect of Logically Appending X'00' to the Final Code Block Before Encoding

It can be seen from this example that COBS encoding always adds at least one octet of overhead. In the worst case, COBS encoding adds one octet of overhead for every 254 data octets (a maximum of six octets for a full NPDU, or less than 0.4%). Annex T describes the implementation of the cobs_encode() and cobs_decode() algorithms in software.

9.10.2 Preparing COBS-Encoded MS/TP Frames for Transmission

The COBS encoding described above eliminates all X'00' octets from the input stream and writes the encoded chunks to output. However, the actual goal is to eliminate MS/TP preamble sequences from the Encoded Data and Encoded CRC-32K fields, which can only be accomplished by eliminating either X'55' or X'FF' from these fields. X'FF' cannot be eliminated as this octet may be optionally appended to the end of any frame, leaving X'55' as the only option. The goal is accomplished by first running the encoding function over the input stream, then XOR'ing every octet in the output stream with X'55'. This modifies every octet in the output stream and transforms all occurrences of X'55' into X'00' (which is an acceptable value in the Encoded Data or Encoded CRC-32K fields).

The logical procedure for preparing COBS-encoded MS/TP frames for transmission is as follows:

(1) Pass the client data through the COBS encoder and save the returned length in DataLength. XOR each of the DataLength octets in the output buffer with X'55' to produce the Encoded Data field.
(2) Add three to DataLength (the five octet length of Encoded CRC-32K field, minus two octets for backward compatibility with legacy MS/TP devices that implement the SKIP_DATA state in their Receive Frame State Machine). Use this modified value of DataLength as the Length field of the outgoing frame.
(3) Initialize the value of CRC32K to X'FFFFFFFF'.
(4) Pass the input stream (Encoded Data field) through CalcCRC32K() and accumulate the result in CRC32K. (It is recommended that this be done as each octet is transmitted to reduce latency.)
(5) Take the ones' complement of CRC32K and if necessary reorder it for transmission least significant octet first.
(6) Pass the modified CRC32K through the COBS encoder and XOR each of the five output octets with X'55' to produce the Encoded CRC-32K field.

9.10.3 Decoding COBS-Encoded MS/TP Frames Upon Reception

The logical procedure for decoding COBS-encoded MS/TP frames upon reception is the inverse of encoding and restores the sender's original NPDU at the receiving node. This procedure assumes the Encoded Data and Encoded CRC-32K octets reside contiguously in InputBuffer in the order received and that the Index and CRC32K variables are initialized on the EncodedFields transition of the Receive Frame Finite State Machine (see Clause 9.5.4.4). The required local variables are initialized as described below. Note that this procedure shows CRC32K being calculated over the Extended Data octets before decoding. However, it is recommended this be performed as the octets are received to reduce latency (see Clause 9.5.4.8).

9.10.3.1 Local Variables

DecodeIndex — Used as an index into InputBuffer[] during decoding. Initialize to zero.

DecodeCount — Used to count down the number of octets remaining to be decoded in the InputBuffer. Initialize to DataLength minus three.

CodeOctet — Used to count down the number of octets remaining to be decoded in the current code block. Initialized in the procedure to the code octet of the current code block.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

LastCode Used to preserve the value (1 - 255) of the code octet of the current code block.

DataOctet Used to buffer the current octet being decoded in the current code block.

9.10.3.2 Procedure

(1) Process a code octet in the Encoded Data field

Set the contents of CodeOctet to the value of InputBuffer[DecodeIndex]; accumulate the contents of CodeOctet into CRC32K; and set CodeOctet equal to the value of CodeOctet XOR'd with X'55'.

If CodeOctet is equal to zero or greater than DecodeCount,

then set the value of CRC32K to X'FFFFFFFF' to indicate that an error has occurred during the reception of a frame and return;

else increment DecodeIndex by 1; decrement DecodeCount by 1; set LastCode to the value of CodeOctet; and decrement CodeOctet by 1.

If CodeOctet is equal to zero then goto step (3) else goto step (2).

(2) Process a data octet in the Encoded Data field

Set the contents of DataOctet to the value of InputBuffer[DecodeIndex]; accumulate the contents of DataOctet into CRC32K; set DataOctet equal to the contents of DataOctet XOR'd with X'55'; save the contents of DataOctet at InputBuffer[Index]; increment Index by 1; increment DecodeIndex by 1; decrement DecodeCount by 1; and decrement CodeOctet by 1.

If CodeOctet is equal to zero then goto step (3) else goto step (2).

(3) If DecodeCount is greater than zero and LastCode is not equal to 255,

then save the value zero at InputBuffer[Index]; and increment Index by 1.

If DecodeCount is greater than zero then goto step (1);

else set DataLength to Index; set the value of DecodeCount to five; and goto step (4).

(4) Process a code octet in the Encoded CRC-32K field

Set the contents of CodeOctet to the value of InputBuffer[DecodeIndex]; and set CodeOctet equal to the contents of CodeOctet XOR'd with X'55'.

If CodeOctet is equal to zero or greater than DecodeCount,

then set the value of CRC32K to X'FFFFFFFF' to indicate that an error has occurred during the reception of a frame and return;

else increment DecodeIndex by 1; decrement DecodeCount by 1; and decrement CodeOctet by 1.

If CodeOctet is equal to zero then goto step (6) else goto step (5).

(5) Process a data octet in the Encoded CRC-32K field, i.e. CRC1, CRC2, CRC3, or CRC4

Set the contents of DataOctet to the value of InputBuffer[DecodeIndex] XOR'd with X'55'; accumulate the contents of DataOctet into CRC32K; increment DecodeIndex by 1; decrement DecodeCount by 1; and decrement CodeOctet by 1.

If CodeOctet is equal to zero then goto step (6) else goto step (5).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(6) If DecodeCount is greater than zero,

then accumulate the value zero into CRC32K; and goto step (4);

else return.

9.11 Documenting MS/TP Device Design Choices

Every MS/TP device includes a collection of design choices that affect the behavior of the device. Among these, the following choices are important in terms of MS/TP interoperability:

(a) master or slave implementation
(b) isolated or non-isolated power source for EIA/TIA-485 transceiver
(c) whether local biasing is built-in to the device
(d) transceiver unit loading
(e) data rates supported by the device

Each MS/TP device shall indicate all of this information in its Protocol Implementation Conformance Statement (PICS). See
Annex A.

9.11.1 Master or Slave Implementation

The PICS shall indicate whether the device implements a Master Node State Machine or Slave Node State Machine or either.

9.11.2 EIA/TIA-485 Power Isolation

The PICS shall indicate whether the EIA/TIA-485 transceiver implements an optically and magnetically isolated power source, or not. Designs that implement such a power source shall be called ISOLATED, and designs that do not shall be called NON-ISOLATED.

9.11.3 Local Biasing

The PICS shall indicate whether the device implements local biasing, by way of 47K ohms pull-up/pull-down (see 9.2.2), as
having “Local Bias 47K.” Devices without 47K ohms biasing shall indicate “Local Bias none.”

9.11.4 Transceiver Unit Loading

Based on the characteristics of the EIA/TIA-485 transceiver of the device, the PICS shall indicate “Unit Load X” where X is
1, ½, ¼, or ⅛.

9.11.5 Data Rates

The PICS shall include a list of each data rate supported by the device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10 DATA LINK/PHYSICAL LAYERS: POINT-TO-POINT (PTP)

This clause defines a data link layer protocol by which two BACnet devices may communicate using a variety of point-to-point (PTP) communication mechanisms. These mechanisms may be accessed through an EIA-232 or bus-level interface to modems, line drivers, or other data communication equipment. The specific physical connection composing the PTP connection is a local matter.

This clause does not attempt to specify the means by which the virtual or physical connection is established. Rather, it specifies a protocol that allows two BACnet network layer entities to establish a BACnet PTP data link connection, reliably exchange BACnet PDUs, and perform an orderly termination of a BACnet PTP connection using an already established physical connection.

This data link layer protocol addresses the particular characteristics associated with a PTP connection. A PTP connection differs from other BACnet data link/physical layer options in several ways: it is capable of full duplex operation, it may be temporary in nature, and it may be significantly slower. The PTP protocol is only used between devices that are half-routers. See Clause 6.7.

This protocol does not assume that the answering device is in a state where it can accept BACnet PDUs containing binary information. For instance, the same serial device port that is dialed into by a BACnet device may also be dialed into and logged onto by a human operator using a simple ANSI X3.4 terminal. Therefore, the connection establishment procedure is initiated using an ANSI X3.4 printable character sequence.

The connection process also provides for optional password protection. The configuration and checking of the password parameter is considered to be a local matter.

10.1 Overview

Once a physical connection has been established between the calling device and the answering device, a sequence of frames are exchanged to establish a BACnet connection. If the connection is established, the two devices may freely exchange BACnet PDUs. Either the calling device or the answering device may initiate a termination of the connection. The connection remains until a request for termination has been issued by either device, either device determines that the physical layer connection has been lost, or until a local timer expires, indicating that the peer device is no longer active. Unlike other BACnet data link protocols, the PTP protocol is acknowledged using an alternating bit approach. It should be noted that the protocol allows PDUs to be exchanged between the two devices simultaneously to take advantage of full duplex operation.

Note that it is also possible that two devices are permanently connected at the physical layer in which case the BACnet connect sequence is performed only once, at initialization time. In this case both devices would be running the PTP data link layer and would always be capable of sending and receiving BACnet PTP data link frames.

10.2 Service Specification

PTP includes a data link layer sufficient to provide to the BACnet network layer the same services as are offered by ISO 8802-2 Type 1. Because PTP is a connection-oriented data link layer, additional primitives are needed to manage the connection establishment and termination phases. PTP does not provide all of the functionality of ISO 8802-2 Type 2.

This clause describes the primitives and parameters associated with the provided services. The parameters are described in an abstract sense, which does not constrain the implementation method. These primitives provide an interface to the BACnet network layer consistent with the other BACnet data link options except for the addition of connection management primitives.

10.2.1 DL-UNITDATA.request

10.2.1.1 Function

This primitive is the service request primitive for the unacknowledged connectionless-mode data transfer service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.2.1.2 Semantics of the Service Primitive

The primitive shall provide parameters as follows:

DL-UNITDATA.request (
source_address,
destination_address,
data,
priority
)

Each source and destination address consists of the logical concatenation of a medium access control (MAC) address and a link service access point (LSAP). However, since PTP does not define or use MAC addresses and since it supports only the BACnet network layer, the 'source_address' and 'destination_address' parameters are ignored.

The 'data' parameter specifies the link service data unit (LSDU) to be transferred by the PTP entity.

The 'priority' parameter specifies the priority desired for the data unit transfer. The priority parameter is ignored by PTP.

10.2.1.3 When Generated

This primitive is passed from the network layer to the PTP entity to request that a network protocol data unit (NPDU) be sent to one or more remote LSAPs using unacknowledged connectionless-mode procedures.

10.2.1.4 Effect on Receipt

Receipt of this primitive causes the PTP entity to attempt to send the NPDU using unacknowledged connectionless-mode procedures.

10.2.2 DL-UNITDATA.indication

10.2.2.1 Function

This primitive is the service indication primitive for the unacknowledged connectionless-mode data transfer service.

10.2.2.2 Semantics of the Service Primitive

The primitive shall provide the following parameters:

DL-UNITDATA.indication (
source_address,
destination_address,
data,
priority
)

Each source and destination address consists of the logical concatenation of a medium access control (MAC) address and a link service access point (LSAP). However, since PTP does not define or use MAC addresses, and since it supports only the BACnet network layer, the 'source_address' and 'destination_address' parameters are ignored.

The 'data' parameter specifies the link service data unit that has been received by the PTP entity.

The 'priority' parameter specifies the priority desired for the data unit transfer. The priority parameter is ignored by PTP.

10.2.2.3 When Generated

This primitive is passed from the PTP entity to the network layer to indicate the arrival of an NPDU from the specified remote entity.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.2.2.4 Effect on Receipt

The effect of receipt of this primitive by the network layer is unspecified.

10.2.3 Test_Request and Test_Response

ISO 8802-2 Type 1 defines XID and TEST PDUs and procedures but does not define an interface to invoke them from the network layer. Test_Request and Test_Response PDUs and procedures have been defined for PTP to accomplish the same functions. Because PTP supports only the equivalent of a single LSAP, these PDUs are sufficient to implement the relevant aspects of XID as well.

The response with Test_Response to a Test_Request PDU is mandatory for all PTP nodes. The origination of a Test_Request PDU is optional.

10.2.3.1 Use of Test_Request and Test_Response for ISO 8802-2 TEST Functions

The TEST function provides a facility to conduct loopback tests of the PTP to PTP transmission path. Successful completion of the test consists of sending a Test_Request PDU with a particular information field to the designated destination, and receiving, in return, the identical information field in a Test_Response PDU.

If a receiving node can successfully receive and return the information field, it shall do so. If it cannot receive and return the entire information field but can detect the reception of a valid Test_Request frame (for example, by computing the CRC on octets as they are received), then the receiving node shall discard the information field and return a Test_Response containing no information field. If the receiving node cannot detect the valid reception of frames with overlength information fields, then no response shall be returned.

10.2.3.2 Use of Test_Request and Test_Response for ISO 8802-2 XID functions

ISO 8802-2 describes seven possible uses of XID:

(a) XID can be used with a null DSAP and null SSAP as an "Are You There" test. Since PTP supports only the equivalent of a single LSAP, the Test_Request PDU with no data can perform this function.
(b) XID can be used with a group or global DSAP to identify group members or all active stations. Since PTP supports only the equivalent of a single LSAP, the Test_Request PDU with no data can perform this function.
(c) XID can be used for a duplicate address check.
(d) Class II LLCs may use XID to determine window size. PTP does not support Class II operation.
(e) XID may be used to identify the class of each LLC. Since PTP supports only Class I operation, this is a trivial operation.
(f) XID may be used to identify the service types supported by each LSAP. Since PTP supports only Class I operation, this is a trivial operation.
(g) An LLC can announce its presence by transmitting an XID with global DSAP. Since PTP supports only one LSAP, the equivalent can be accomplished by transmitting a Test_Response PDU.

10.2.4 DL-CONNECT.request

10.2.4.1 Function

This primitive is the service request primitive for the connection establishment service.

10.2.4.2 Semantics of the Service Primitive

The primitive shall provide the following parameters:

```
DL-CONNECT.request (
        destination_address,
        password
        )
```

The 'destination_address' parameter specifies the information required by the PTP entity to initiate the establishment of a physical connection between the local and remote BACnet devices. Although, as stated at the beginning of this clause, the establishment of the physical connection is a local matter, it is likely that this parameter would convey information

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

such as contained in the Port Info field of the Initialize-Routing-Table network layer message and subsequently stored in a node's routing table. See Clause 6.4.7.

The 'password' parameter specifies the password to be used in the PTP connection process described in Clause 10.4.8.

10.2.4.3 When Generated

This primitive is passed from the network layer to the PTP entity to request that a logical link connection be established.

10.2.4.4 Effect on Receipt

The receipt of this primitive causes the PTP entity to initiate establishment of a connection with the remote PTP entity.

10.2.5 DL-CONNECT.indication

10.2.5.1 Function

This primitive is the service indication primitive for the connection establishment service.

10.2.5.2 Semantics of the Service Primitive

The primitive provides no parameters.

10.2.5.3 When Generated

This primitive is passed from the PTP entity to the network layer to indicate that a logical link connection has been established.

10.2.5.4 Effect on Receipt

The network layer entity may use this connection for data unit transfer.

10.2.6 DL-CONNECT.confirm

10.2.6.1 Function

This primitive is the service confirmation primitive for the connection establishment service.

10.2.6.2 Semantics of the Service Primitive

The primitive shall provide the following parameter:

```
DL-CONNECT.confirm (
        status
        )
```

The 'status' parameter specifies whether or the not the connection has been successfully established.

10.2.6.3 When Generated

This primitive is passed from the PTP entity to the network layer to indicate that a logical link connection has been established.

10.2.6.4 Effect on Receipt

The network layer entity may use this connection for data unit transfer if the 'status' parameter indicates the successful establishment of a PTP connection.

10.2.7 DL-DISCONNECT.request

10.2.7.1 Function

This primitive is the service request primitive for the connection termination service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.2.7.2 Semantics of the Service Primitive

The primitive shall provide the following parameters:

DL-DISCONNECT.request (
destination_address
)

The 'destination_address' parameter specifies the information required by the PTP entity to initiate the establishment of a physical connection between the local and remote BACnet devices. The PTP entity uses this same information to identify the particular PTP connection instance that is to be terminated.

10.2.7.3 When Generated

This primitive is passed from the network layer to the PTP entity to request that a logical link connection be terminated.

10.2.7.4 Effect on Receipt

The receipt of this primitive causes the PTP entity to initiate termination of a connection with the remote PTP entity.

10.2.8 DL-DISCONNECT.indication

10.2.8.1 Function

This primitive is the service indication primitive for the connection termination service.

10.2.8.2 Semantics of the Service Primitive

The primitive shall provide the following parameters:

DL-DISCONNECT.indication (
reason
)

The 'reason' parameter specifies the reason for the disconnection. The reasons for disconnection may include a request by the remote entity, loss of physical connection, or an error internal to the PTP sublayer.

10.2.8.3 When Generated

This primitive is passed from the PTP entity to the network layer to indicate that a logical link connection has been terminated.

10.2.8.4 Effect on Receipt

The network layer entity may no longer use this connection for data unit transfer.

10.2.9 DL-DISCONNECT.confirm

10.2.9.1 Function

This primitive is the service confirmation primitive for the connection termination service.

10.2.9.2 Semantics of the Service Primitive

The primitive shall provide the following parameters:

DL-DISCONNECT.confirm (
destination_address
)

The 'destination_address' parameter specifies the information required by the PTP entity to initiate the establishment of a physical connection between the local and remote BACnet devices. The network layer entity uses this same information to identify the particular PTP connection instance that has been terminated.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.2.9.3 When Generated

This primitive is passed from the PTP entity to the network layer to indicate that a logical link connection has been terminated.

10.2.9.4 Effect on Receipt

The network layer entity may no longer use this connection for data unit transfer.

10.3 Point-to-Point Frame Format

All PTP data link frames, with the exception of the ANSI X3.4 sequence used to initiate a PTP connection, have the following format:

Preamble	two octet preamble X'55FF'
Frame Type	one octet
Length	length of data field not including CRC, two octets, most significant octet first
Header CRC	one octet
Data	varies with frame type; variable length
Data CRC	(if data is present) two octets, least significant octet first

The Preamble, Frame Type, Length, and Header CRC are collectively known as the header segment of the frame. The Data and Data CRC are collectively known as the data segment of the frame. The Frame Type is used to distinguish between different types of MAC frames. Defined types are:

X'00'	Heartbeat XOFF
X'01'	Heartbeat XON
X'02'	Data 0
X'03'	Data 1
X'04'	Data Ack 0 XOFF
X'05'	Data Ack 1 XOFF
X'06'	Data Ack 0 XON
X'07'	Data Ack 1 XON
X'08'	Data Nak 0 XOFF
X'09'	Data Nak 1 XOFF
X'0A'	Data Nak 0 XON
X'0B'	Data Nak 1 XON
X'0C'	Connect Request
X'0D'	Connect Response
X'0E'	Disconnect Request
X'0F'	Disconnect Response
X'14'	Test_Request
X'15'	Test_Response

Frame Types X'00' through X'7F' are reserved by ASHRAE. Frame types X'10', X'11', and X '13' shall never be used for a valid frame type because of the character transparency method described in Clause 10.3.1. Frame Types X'80' through X'FF' are available to vendors for proprietary (non-BACnet) frames. Proprietary PTP frames shall follow the same state machine transitions defined for Data frames.

The Data field is conditional on the Frame Type, as specified in the description of each Frame Type. If there is no Data field, then the length field shall be zero and the Data and Data CRC (data segment) shall be omitted.

The header CRC octet is the ones complement of the remainder that results when the Frame Type and Length fields are divided by the CRC polynomial

$$G(X) = X^8 + X^7 + 1.$$

The data CRC octets are the ones complement of the remainder that results when the Data field is divided by the CRC-CCITT polynomial

$$G(X) = X^{16} + X^{12} + X^5 + 1.$$

Annex G describes in detail the generation and checking of the CRCs.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.3.1 Character Transparency and Flow Control

In order to support modems that respond to flow control or other control characters, character stuffing is used to prevent transmission of these codes as part of the data. Where a value corresponding to a control character would appear in a frame, it shall be prefixed with a data link escape code (X'10') and the high order bit shall be set in the value as transmitted. The control characters listed below shall be encoded in this manner. Implementations shall be able to receive and decode all encoded control characters.

X'10'	(DLE)	=>	X'10' X'90'
X'11'	(XON)	=>	X'10' X'91'
X'13'	(XOFF)	=>	X'10' X'93'

The characters X'11' (XON) and X'13' (XOFF) are never transmitted by the SendFrame procedure described in Clause 10.4.4 and are ignored by the Receive Frame state machine described in Clause 10.4.7. The use of these characters or of Request To Send (RTS), Clear To Send (CTS), or other EIA-232 control lines for flow control purposes is a local matter. The use of such methods of flow control is allowed only between a PTP device and local equipment such as a modem. Flow control between PTP devices shall be implemented using the flow control frames defined in Clause 10.3.

10.3.2 Frame Types X'00' and X'01': Heartbeat Frames

A frame of one of these types is transmitted by each device periodically when no other data are ready to transmit, to indicate to the peer device that the data link is still active. Heartbeat frames contain no data segment. A type X'00' frame is transmitted to indicate to the peer device that the local device is not ready to accept Data frames. A type X'01' frame is transmitted to indicate readiness to receive Data frames.

10.3.3 Frame Types X'02' and X'03': Data Frames

A frame of one of these types is transmitted to convey data (NPDUs) to the peer device. The length of the data field of a Data frame may range from 0 to 501 octets. Successive transmissions alternate frame types; type X'02' corresponds to transmit sequence number 0, and type X'03' corresponds to transmit sequence number 1.

10.3.4 Frame Types X'04' through X'07': Data Ack Frames

A frame of one of these types is transmitted to acknowledge a correctly received Data frame. Data Ack frames contain no data segment. Frame types X'04' and X'06' acknowledge receipt of Data frames with sequence number 0 (type X'02'). Frame types X'05' and X'07' acknowledge receipt of Data frames with sequence number 1 (type X'03'). Frame types X'04' and X'05' indicate that the device is not ready to receive additional Data frames (XOFF). Frame types X'06' and X'07' indicate that the device is ready to receive additional Data frames (XON).

10.3.5 Frame Types X'08' through X'0B': Data Nak Frames

A frame of one of these types is used to reject an incorrectly received Data frame. Data Nak frames are transmitted when the header segment of a Data frame has been correctly received but the data segment of the frame contains an error or when a Data frame cannot be accepted due to receiver buffer limitations. Data Nak frames contain no data segment. Frame types X'08' and X'0A' reject Data frames with sequence number 0 (type X'02'). Frame types X'09' and X'0B' reject Data frames with sequence number 1 (type X'03'). Frame types X'08' and X'09' indicate that the device is not ready to receive additional Data frames (XOFF). Frame types X'0A' and X'0B' indicate that the device is ready to receive additional Data frames (XON).

10.3.6 Frame Type X'0C': Connect Request Frame

The Connect Request frame is issued by the answering device in an attempt to establish a BACnet connection. Connect Request frames contain no data segment.

10.3.7 Frame Type X'0D': Connect Response Frame

The Connect Response frame is issued by a device in response to a received Connect Request frame. The data field of the Connect Response frame, if present, contains a password. The length and content of the optional password field are a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.3.8 Frame Type X'0E': Disconnect Request Frame

The Disconnect Request frame may be issued by either device when it wishes to discontinue the BACnet PTP dialogue. The data field of the frame conveys the reason for requesting a disconnect and shall be one octet in length. The permissible values for the data are:

X'00' no more data needs to be transmitted,
X'01' the peer process is being preempted,
X'02' the received password is invalid,
X'03' other.

10.3.9 Frame Type X'0F': Disconnect Response Frame

The Disconnect Response frame is used to acknowledge a previously received Disconnect Request frame. The Disconnect Response frame indicates that the responding device accepts the request to disconnect. Disconnect Response frames contain no data segment.

10.3.10 Frame Type X'14': Test_Request

This frame is used to initiate a loopback test of the PTP transmission path. The use of this frame is described in detail in Clause 10.2.3. The length of the data field of a Test_Request frame may range from 0 to 501 octets.

10.3.11 Frame Type X'15': Test_Response

This frame is used to reply to a Test_Request frame. The use of this frame is described in detail in Clause 10.2.3. The length of the data field of a Test_Response frame may range from 0 to 501 octets. The data, if present, shall be those that were present in the initiating Test_Request.

10.4 PTP Medium Access Control Protocol

This clause defines the PTP protocol. The protocol definition has been broken into several discrete parts that collectively describe and define the entire PTP protocol. The first part presents a universal asynchronous receiver transmitter (UART) model of the hardware platform. This is followed by definitions of the variables and constants used to define the protocol. A finite state machine is used to define in detail the process of receiving PTP frames (see Clause 10.4.7). An informal procedure describes in detail the process of transmitting a PTP frame (see Clause 10.4.4). These details are referenced in subsequent clauses, which define the protocol at a higher level of abstraction.

The process of establishing a PTP connection and terminating a PTP connection is defined by a finite state machine called the Connection State Machine (see Clause 10.4.9). The PTP protocol is full duplex. Thus, when a PTP connection is active, each device may simultaneously play the role of transmitting and receiving PTP messages. Two separate finite-state machines define the aspects of the protocol that pertain to these two roles. These finite state machines are call the Transmission State Machine (see Clause 10.4.10) and the Reception State Machine (see Clause 10.4.11). The Transmission State Machine and the Reception State Machine are assumed to operate concurrently whenever the Connection State Machine is in the CONNECTED state. In this model, the Reception State Machine and the Transmission State Machine exchange information through the use of shared variables. The details of transmitting and receiving a PTP frame are described by the SendFrame, SendHeaderOctet, and SendOctet procedures and the ReceiveFrame state machine respectively.

10.4.1 UART Receiver Model

The receiver interface to a UART is modeled as a data register and two Boolean flags. These are intended to closely resemble the functions of commercial UART chips but in a generic and nonprescriptive fashion. The model is used by both the procedural and state machine descriptions.

10.4.1.1 DataRegister

The DataRegister holds the octet most recently received. The contents of this register after the occurrence of a framing or overrun error are not specified.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.4.1.2 DataAvailable

The flag DataAvailable is TRUE if an octet is available in DataRegister. A means of setting this flag to FALSE when the associated data have been read from DataRegister shall be provided. Many common UARTs set DataAvailable FALSE automatically when DataRegister is read.

10.4.1.3 ReceiveError

The flag ReceiveError is TRUE if an error is detected during the reception of an octet. Many common UARTs detect several types of receive errors, in particular framing errors and overrun errors. ReceiveError shall be TRUE if any of these errors is detected.

A framing error occurs if a logical zero is received when a stop bit (logical one) is expected.

An overrun error occurs if an octet is received before an earlier octet is read from DataRegister. In general, the occurrence of overrun errors is evidence of improper design. However, it is recognized that critical system events may cause overrun errors to occur from time to time. The inclusion of this error in the state machine processing ensures that such errors are handled in a deterministic fashion.

A means of setting ReceiveError to FALSE when the associated error has been recognized shall be provided.

10.4.2 Variables

The variables and timers used by the PTP protocol are described in this clause.

Ack0Received A Boolean flag indicating whether (TRUE) or not (FALSE) the Reception State Machine has received an acknowledgment that a previous Data frame with a sequence number of 0 was correctly received.

Ack1Received A Boolean flag indicating whether (TRUE) or not (FALSE) the Reception State Machine has received an acknowledgment that a previous Data frame with a sequence number of 1 was correctly received.

DataCRC Used to accumulate the CRC on the data field of a frame.

DataLength An unsigned integer in the range from 0 to 501 that indicates the expected number of octets in the Data field of a PTP frame. This value is derived from the Length field of the received PTP frame. The value of DataLength excludes any data link escape octets (X'10') and the octets in the Data CRC.

DLE_Mask A bit mask used to process received octets in order to account for the fact that they may be encoded.

FrameType Used by the Receive State Machine to store the frame type of a received frame.

HeaderCRC Used to accumulate the CRC of the header of a frame.

HeartbeatTimer A timer used to initiate Heartbeat frames to keep the link active.

InactivityTimer A timer used to monitor the time since this station received a frame.

Index Indicates the location of the end of the data in the InputBuffer array.

InputBuffer[ ] An array of octets used to store data octets as they are received. InputBuffer is indexed from 0 to InputBufferSize-1. The maximum size of the data field of a frame is 501 octets.

InputBufferSize The number of elements in the array InputBuffer[].

Nak0Received A Boolean flag indicating whether (TRUE) or not (FALSE) the Reception State Machine has received a reject in response to a previous Data frame with a sequence number of 0.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Nak1Received A Boolean flag indicating whether (TRUE) or not (FALSE) the Reception State Machine has received a reject in response to a previous Data frame with a sequence number of 1.

ReceivedInvalidFrame A Boolean flag set to TRUE by the Receive Frame Machine if an error of any type is detected. Errors include octet framing, overrun, CRC, and receive buffer overflow.

ReceivedValidFrame A Boolean flag set to TRUE by the Receive Frame Machine if a valid frame has been received.

ReceptionBlocked An enumerated variable indicating whether or not reception of Data frames is blocked. The values of the enumeration are BLOCKED, ALMOST_BLOCKED, and NOT_BLOCKED. The value of this variable is determined by a buffer manager. The buffer manager process is a local matter.

ResponseTimer A timer used to monitor the time spent waiting for a response from the remote device.

RetryCount A counter of transmission retries.

RxSequenceNumber An integer containing the sequence number (0 or 1) expected for the next Data frame to be received.

SendingFrameNow A Boolean flag indicating whether (TRUE) or not (FALSE) an invocation of the SendFrame procedure is currently in the process of transmitting octets.

SilenceTimer A timer used to monitor the time since this station received an octet.

TransmissionBlocked A Boolean flag indicating whether (TRUE) or not (FALSE) frame transmission has been blocked. The value of this flag is determined by the receipt of XON and XOFF frames from the peer device.

TransmitDataCRC Used to accumulate the CRC on the data field of a frame being transmitted.

TransmitHeaderCRC Used to accumulate the CRC on the header of a frame being transmitted.

TxSequenceNumber An integer containing the sequence number (0 or 1) for the next Data frame to be transmitted.

10.4.3 Parameters

The following parameters are used in the PTP data link protocol.

$N_{retries}$ The maximum number of times a frame shall be sent before an error is reported. The value of $N_{retries}$ shall be 3.

T_{conn_rqst} The maximum time allowed by a calling device for an answering device to issue a PTP Connect Request frame once the physical connection has been established and the ANSI X3.4 trigger sequence has been transmitted. The value of T_{conn_rqst} shall be 15 seconds. This represents the processing time required for the answering device to recognize the ANSI X3.4 trigger sequence, prepare the communication port for PTP protocol, and transmit the Connect Request frame.

T_{conn_rsp} The maximum time allowed to respond to a PTP Connect Request frame with a PTP Connect Response frame. The value of T_{conn_rsp} shall be 15 seconds. This represents the time required for the calling device to process the Connect Request frame and the time required to transmit the Connect Response frame.

T_{frame_abort} The maximum time allowed between receipt of octets in a frame after which time the receiving device shall assume a transmission error. The value of T_{frame_abort} shall be 2 seconds.

$T_{heartbeat}$ The maximum delay between frame transmissions before a heartbeat frame must be sent. The value of $T_{heartbeat}$ shall be 15 seconds.

$T_{inactivity}$ The maximum amount of time the InactivityTimer may attain, after which a device may assume that the PTP connection has been disrupted. The value of $T_{inactivity}$ shall be 60 seconds.

$T_{response}$ The maximum time allowed waiting for a Data Ack frame in response to a Data frame. The value of $T_{response}$ shall be 5 seconds.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.4.4 SendFrame Procedure

This clause describes the transmission of PTP data link frames. A mutual exclusion semaphore, SendingFrameNow, synchronizes access to the functionality by multiple, asynchronous invocations. Frames are transmitted in two parts, the header segment and the data segment.

NOTE: At the implementor's option, character-level flow control may be implemented by conditioning the transmission of octets by this procedure based on the reception of the characters XOFF (X'13') and XON (X'11') or by the presence or absence of active levels on certain EIA-232 control lines. Such character-level flow control is a local matter.

(1) If SendingFrameNow is TRUE, then wait for the other invocation of the SendFrame procedure to set SendingFrameNow to FALSE.
(2) Set SendingFrameNow to TRUE.
(3) Transmit the preamble octets X'55' and X'FF'.
(4) Initialize TransmitHeaderCRC to X'FF'.
(5) Call SendHeaderOctet to transmit the frame type and the high and low data length octets.
(6) Call SendOctet to transmit the one's-complement of TransmitHeaderCRC.
(7) If the data length is zero, proceed to step (m). Otherwise, proceed to step (h).
(8) Initialize TransmitDataCRC to X'FFFF'.
(9) Accumulate each data octet into TransmitDataCRC.
(10) Call SendOctet to transmit each data octet.
(11) Call SendOctet to transmit the one's-complement of the least significant octet of TransmitDataCRC.
(12) Call SendOctet to transmit the one's-complement of the most significant octet of TransmitDataCRC.
(13) Set HeartbeatTimer to zero; set SendingFrameNow to FALSE. Transmission is complete.

10.4.5 SendHeaderOctet Procedure

This clause describes the transmission of the header octets of PTP frames.

(1) Accumulate the header octet into TransmitHeaderCRC.
(2) Call SendOctet to transmit the header octet.

10.4.6 SendOctet Procedure

This clause describes the transmission of octets of PTP frames.

If the value of the octet is not X'10', X'11', or X'13', then transmit the octet. Otherwise, transmit the value X'10', set the high order bit of the octet, and transmit the modified octet.

10.4.7 Receive Frame State Machine

This clause describes the reception of a PTP frame by a BACnet device. The description of operation is as a finite state machine. Figure 10-1 shows the Receive Frame state machine, which is described fully in this clause. Each state is given a name, specified in all capital letters. Transitions are also named, in mixed upper- and lowercase letters. Transitions are described as a series of conditions followed by a series of actions to be taken if the conditions are met. The final action in each transition is entry into a new state, which may be the same as the current state.

The Receive Frame state machine operates independently from the other PTP state machines, communicating with them by means of flags and other variables. The description assumes that the other state machines can process received frames and other indications from the Receive Frame state machine before the next frame begins. The means by which this behavior is implemented are a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

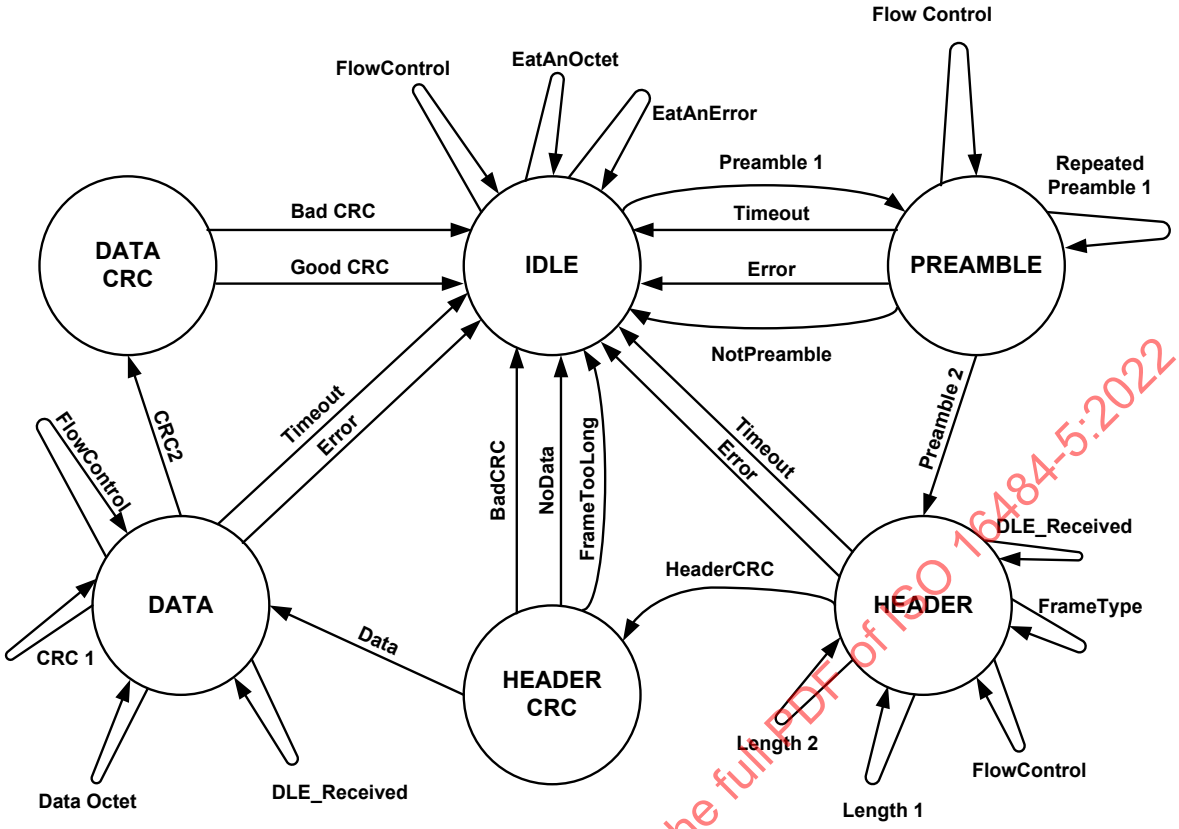


Figure 10-1. Receive Frame State Machine.

10.4.7.1 IDLE

In the IDLE state, the node waits for the beginning of a frame.

EatAnError

If ReceiveError is TRUE,

then set SilenceTimer to zero; set ReceiveError to FALSE; and enter the IDLE state to wait for the start of a frame.

FlowControl

If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is equal to either X'11' or X'13',

then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the IDLE state.

EatAnOctet

If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is not equal to X'55', X'11', or X'13',

then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the IDLE state to wait for the start of a frame.

Preamble1

If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is equal to X'55',

then set SilenceTimer to zero; set DataAvailable to FALSE; set ReceivedValidFrame to FALSE; set ReceivedInvalidFrame to FALSE; and enter the PREAMBLE state to receive the remainder of the frame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

10.4.7.2 PREAMBLE

In the PREAMBLE state, the node waits for the second octet of the preamble.

Timeout
: If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

  then a correct preamble has not been received. Enter the IDLE state to wait for the start of a frame.

Error
: If ReceiveError is TRUE,

  then set SilenceTimer to zero; set ReceiveError to FALSE; and enter the IDLE state to wait for the start of a frame.

FlowControl
: If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is equal to either X'11' or X'13',

  then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the PREAMBLE state.

RepeatedPreamble1
: If ReceiveError is FALSE and DataAvailable is TRUE and the contents of DataRegister is equal to X'55',

  then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the PREAMBLE state to wait for the second preamble octet.

NotPreamble
: If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is not equal to X'FF', X'55', X'11', or X'13',

  then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the IDLE state to wait for the start of a frame.

Preamble2
: If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is equal to X'FF',

  then set SilenceTimer to zero; set DLE_Mask to X'00'; set HeaderCRC to X'FF'; set DataAvailable to FALSE; set Index to zero; and enter the HEADER state to receive the remainder of the frame.

10.4.7.3 HEADER

In the HEADER state, the node waits for the fixed frame header.

Timeout
: If SilenceTimer is greater than $\mathbf{T}_{\text{frame_abort}}$,

  then enter the IDLE state to wait for the start of a frame.

Error
: If ReceiveError is TRUE,

  then set SilenceTimer to zero; set ReceiveError to FALSE; and enter the IDLE state to wait for the start of a frame.

FlowControl
: If ReceiveError is FALSE and DataAvailable is TRUE and the contents of DataRegister is equal to either X'11' or X'13',

  then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the HEADER state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DLE_Received
If ReceiveError is FALSE and DataAvailable is TRUE and the content of the DataRegister is equal to X'10',

then set DLE_Mask to X'80' and enter the HEADER state.

FrameType
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 0 and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; save the result as FrameType; accumulate the result into HeaderCRC; set DataAvailable to FALSE; set DLE_Mask to X'00'; set Index to 1; and enter the HEADER state.

Length1
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 1 and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into HeaderCRC; multiply the result by 256 and save this result as DataLength; set DataAvailable to FALSE; set DLE_Mask to X'00'; set Index to 2; and enter the HEADER state.

Length2
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 2 and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into HeaderCRC; add the result to DataLength and save this result as DataLength; set DataAvailable to FALSE; set DLE_Mask to X'00'; set Index to 3; and enter the HEADER state.

HeaderCRC
If ReceiveError is FALSE and DataAvailable is TRUE and Index is 3 and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into HeaderCRC; set DataAvailable to FALSE; set DLE_Mask to X'00'; and enter the HEADER_CRC state.

10.4.7.4 HEADER_CRC

In the HEADER_CRC state, the node validates the CRC on the fixed frame header.

BadCRC
If the value of HeaderCRC is not X'55',

then enter the IDLE state to wait for the start of the next frame.

FrameTooLong
If the value of HeaderCRC is X'55' and DataLength is greater than InputBufferSize,

then set ReceivedInvalidFrame to TRUE to indicate that a frame cannot be received, and enter the IDLE state to wait for the start of the next frame.

NoData
If the value of HeaderCRC is X'55' and DataLength is zero,

then set ReceivedValidFrame to TRUE to indicate that a frame with no data has been received, and enter the IDLE state to wait for the start of the next frame.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Data

If the value of HeaderCRC is X'55' and DataLength is greater than zero but less than or equal to InputBufferSize,

then set Index to zero; set DataCRC to X'FFFF'; and enter the DATA state to receive the data field of the frame.

10.4.7.5 DATA

In the DATA state, the node waits for the data field of a frame.

Timeout

If SilenceTimer is greater than $\mathbf{T}_{\mathbf{frame_abort}}$,

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

Error

If ReceiveError is TRUE,

then set SilenceTimer to zero; set ReceiveError to FALSE; set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame; and enter the IDLE state to wait for the start of the next frame.

FlowControl

If ReceiveError is FALSE and DataAvailable is TRUE and the content of DataRegister is equal to either X'11' or X'13',

then set SilenceTimer to zero; set DataAvailable to FALSE; and enter the DATA state.

DLE_Received

If ReceiveError is FALSE and DataAvailable is TRUE and the content of the DataRegister is equal to X'10',

then set DLE_Mask to X'80' and enter the DATA state.

DataOctet

If ReceiveError is FALSE and DataAvailable is TRUE and Index is less than DataLength and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into DataCRC; save the results at InputBuffer[Index]; increment Index by 1; set DataAvailable to FALSE; set DLE_Mask to X'00'; and enter the DATA state.

CRC1

If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into DataCRC; increment Index by 1; set DataAvailable to FALSE; set DLE_Mask to X'00'; and enter the DATA state.

CRC2

If ReceiveError is FALSE and DataAvailable is TRUE and Index is equal to DataLength plus 1 and the content of DataRegister is not equal to X'10', X'11', or X'13',

then perform a bitwise AND of the ones-complement of the DLE_Mask and the contents of DataRegister; accumulate the result into DataCRC; set DataAvailable to FALSE; set DLE_Mask to X'00'; and enter the DATA_CRC state.

10.4.7.6 DATA_CRC

In the DATA_CRC state, the node validates the CRC on the frame data.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BadCRC
If the value of DataCRC is not X'F0B8',

then set ReceivedInvalidFrame to TRUE to indicate that an error has occurred during the reception of a frame, and enter the IDLE state to wait for the start of the next frame.

GoodCRC
If the value of DataCRC is X'F0B8',

then set ReceivedValidFrame to TRUE to indicate the complete reception of a valid frame, and enter the IDLE state to wait for the start of the next frame.

10.4.8 Data Link Connection Establishment and Termination Procedures

This clause provides an overview of the protocol for establishing and terminating PTP connections. The details for this protocol are defined by the Connection State Machine in Clause 10.4.9.

Upon establishment of a physical connection between BACnet devices, the calling device shall transmit the seven character ANSI X3.4 trigger sequence "BACnet<CR>", where "<CR>" denotes the ANSI X3.4 character X'0D', to inform the answering device that it wishes to establish a BACnet PTP connection. The answering device shall then transmit a Connect Request frame. The calling device shall respond by transmitting a Connect Response frame including a password, if password protection is implemented. After successful completion of this process, including verification of the password, both devices enter the data exchange phase.

Upon completion of the data link establishment procedure, each device shall assume that the other is not yet ready to receive Data frames. When a Heartbeat XON frame is received from a device, data transmission to that device may begin. Upon completion of the data link establishment procedure and when each device is ready to receive Data frames, it shall immediately transmit a Heartbeat XON frame.

When either device wishes to terminate an active PTP connection, it shall transmit a Disconnect Request frame indicating the reason for the disconnection. The peer device shall respond by transmitting a Disconnect Response frame to acknowledge the request. Both devices shall then notify their respective network layers of data link termination.

If no Disconnect Response is received in reply to the Disconnect Request, an assumption is made that the request was not received by the peer device. The Disconnect Request is retransmitted up to three times. If a Disconnect Response is not received after the third retry, the connection is unilaterally terminated.

If, following transmission of a Disconnect Request, a device receives a Disconnect Request frame from the peer device, the device shall respond by transmitting a Disconnect Response frame and terminating the connection.

If, following transmission of a Disconnect Request, a device receives a Data frame from the peer device, the device shall not acknowledge the Data frame. Thus, a Disconnect Request is an attempt to terminate the connection in an orderly manner, but it is not negotiable. Once a Disconnect Request has been made, the connection shall be terminated. If the peer device needs to continue the communication, a new connection must be established.

10.4.9 Connection State Machine

The operation of the connection establishment state machine is described in this clause and is depicted in Figure 10-2. The state machine models the actions taken to establish a BACnet PTP data link between two devices and includes the actions required for both the calling and answering devices.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

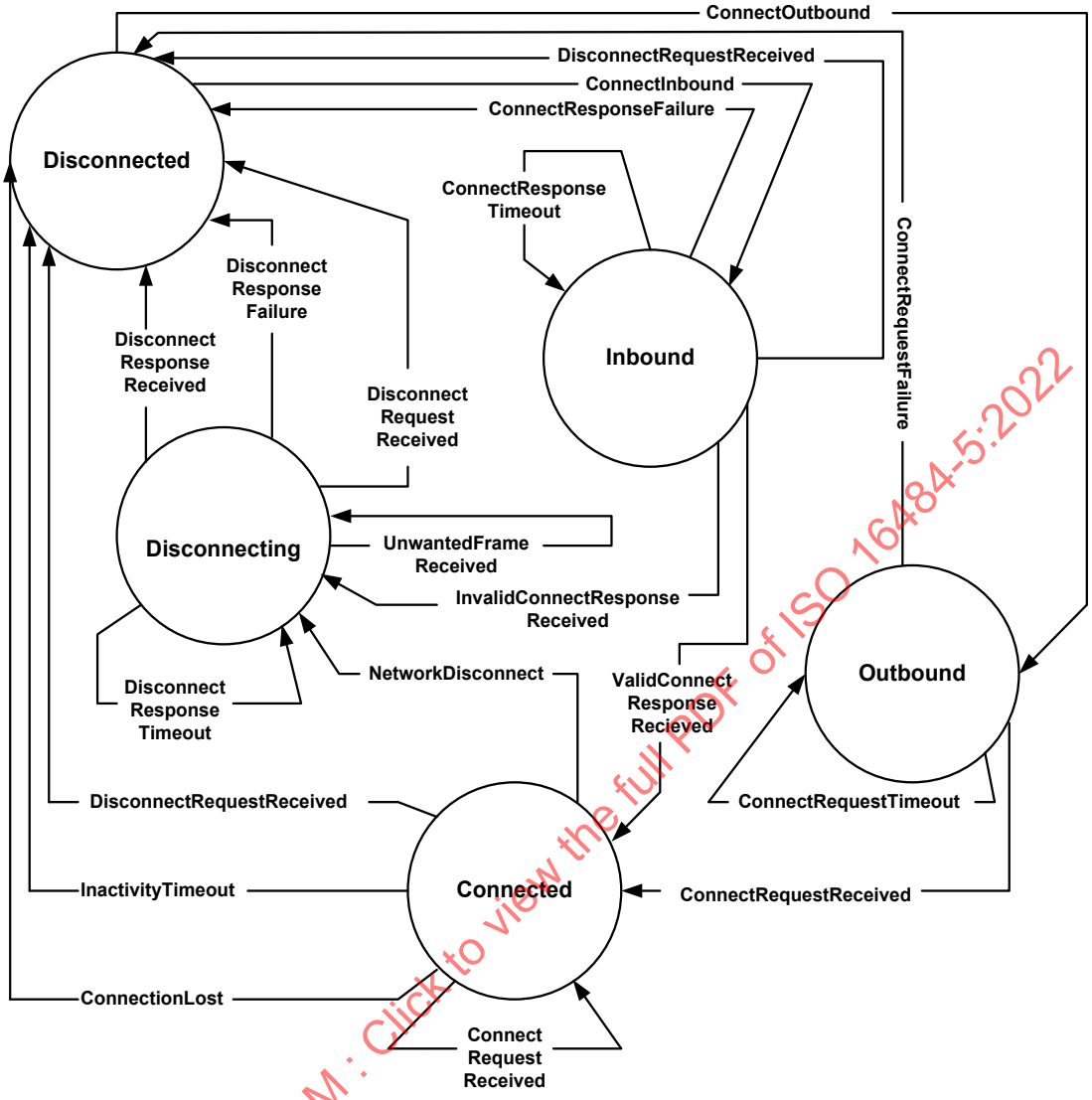


Figure 10-2. Point-To-Point Connection State Machine

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

10.4.9.1 DISCONNECTED

In this state, the device waits for the network layer to initiate a PTP data link connection or for the physical layer to indicate the occurrence of a physical layer connection.

ConnectOutbound
If a DL-CONNECT.request is received,

then establish a physical connection; transmit the "BACnet<CR>" trigger sequence; set RetryCount to zero; set ResponseTimer to zero; and enter the OUTBOUND state.

ConnectInbound
If a physical layer connection has been made and the "BACnet<CR>" trigger sequence is received,

then call SendFrame to transmit a Connect Request frame; set RetryCount to zero; set ResponseTimer to zero; and enter the INBOUND state.

10.4.9.2 OUTBOUND

In this state, the network layer has issued a request to start the data link as a caller, and the device is waiting for a Connect Request frame from the answering device.

© ISO 2022 – All rights reserved

ConnectRequestReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Connect Request,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Connect Response frame containing the password contained in the "data" parameter of the DL-CONNECT.request that initiated the connection; issue a DL-CONNECT.confirm to notify the network layer that a connection has been established; and enter the CONNECTED state.

ConnectRequestTimeout
If ResponseTimer is greater than or equal to $\mathbf{T}_{\mathbf{conn_rqst}}$ and RetryCount is less than $\mathbf{N}_{\mathbf{retries}}$,

then set RetryCount to RetryCount + 1; retransmit the "BACnet<CR>" trigger sequence; set the ResponseTimer to zero; and enter the OUTBOUND state.

ConnectRequestFailure
If ResponseTimer is greater than or equal to $\mathbf{T}_{\mathbf{conn_rqst}}$ and RetryCount is greater than or equal to $\mathbf{N}_{\mathbf{retries}}$,

then issue a DL-CONNECT.confirm to notify the network layer of the failure and enter the DISCONNECTED state.

10.4.9.3 INBOUND

In this state, the Connection State Machine has recognized that the calling device wishes to establish a BACnet connection, and the local device is waiting for a Connect Response frame from the calling device.

ValidConnectResponseReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Connect Response and a password is not needed or a valid password is present in the data field of the frame,

then set ReceivedValidFrame to FALSE; issue a DL-CONNECT.indication to notify the network layer of the connection; and enter the CONNECTED state.

InvalidConnectResponseReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Connect Response and a password is needed but not present or an invalid password is present in the data field of the frame,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Disconnect Request frame indicating the receipt of an invalid password; set ResponseTimer to zero; set RetryCount to zero; and enter the DISCONNECTING state.

ConnectResponseTimeout
If ResponseTimer is greater than or equal to $\mathbf{T}_{\mathbf{conn_rsp}}$ and RetryCount is less than $\mathbf{N}_{\mathbf{retries}}$,

then set RetryCount to RetryCount + 1; call SendFrame to transmit a Connect Request frame; set ResponseTimer to zero; and enter the INBOUND state.

ConnectResponseFailure
If ResponseTimer is greater than or equal to $\mathbf{T}_{\mathbf{conn_rsp}}$ and RetryCount is greater than or equal to $\mathbf{N}_{\mathbf{retries}}$,

then enter the DISCONNECTED state.

DisconnectRequestReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Disconnect Request,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Disconnect Response frame; and enter the DISCONNECTED state.

10.4.9.4 CONNECTED

In this state, the connection procedure has been completed, and the two devices may exchange BACnet PDUs. The data link remains in this state until termination.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NetworkDisconnect
If a DL-DISCONNECT.request is received,

then call SendFrame to transmit a Disconnect Request frame; set ResponseTimer to zero; issue a DL-DISCONNECT.confirm to notify the network layer of the disconnection; set RetryCount to zero; and enter the DISCONNECTING state.

DisconnectRequestReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Disconnect Request,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Disconnect Response frame; issue a DL-DISCONNECT.indication to notify the network layer of the disconnection; and enter the DISCONNECTED state.

ConnectRequestReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Connect Request,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Connect Response frame; issue a DL-CONNECT.indication to notify the network layer of the connection; and enter the CONNECTED state.

InactivityTimeout
If InactivityTimer is greater than $T_{inactivity}$ and ReceivedValidFrame is FALSE,

then issue a DL-DISCONNECT.indication to notify the network layer of the disconnection and enter the DISCONNECTED state.

ConnectionLost
If the physical connection has been terminated, e.g., due to loss of carrier,

then issue a DL-DISCONNECT.indication to notify the network layer of the disconnection and enter the DISCONNECTED state.

10.4.9.5 DISCONNECTING

In this state, the network layer has requested termination of the data link. The device is waiting for a Disconnect Response frame from the peer device.

DisconnectResponseReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Disconnect Response,

then set ReceivedValidFrame to FALSE and enter the DISCONNECTED state.

DisconnectRequestReceived
If ReceivedValidFrame is TRUE and FrameType is equal to Disconnect Request,

then set ReceivedValidFrame to FALSE; call SendFrame to transmit a Disconnect Response frame; and enter the DISCONNECTED state.

UnwantedFrameReceived
If ReceivedValidFrame is TRUE and FrameType is not equal to either Disconnect Response or Disconnect Request,

then set ReceivedValidFrame to FALSE and enter the DISCONNECTING state. (Note that ResponseTimer is not reset in this case.)

DisconnectResponseTimeout
If ResponseTimer is greater than $\mathbf{T_{response}}$ and ReceivedValidFrame is FALSE and RetryCount is less than $\mathbf{N_{retries}}$,

then increment RetryCount; call SendFrame to transmit a Disconnect Request frame; set ResponseTimer to zero; and enter the DISCONNECTING state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DisconnectResponseFailure

If ResponseTimer is greater than or equal to $\mathbf{T_{response}}$ and ReceivedValidFrame is FALSE and RetryCount is greater than or equal to $\mathbf{N_{retries}}$,

then enter the DISCONNECTED state.

10.4.10 Transmission State Machine

The operation of the Transmission State Machine is described in this clause and is depicted in Figure 10-3. The state machine models the actions taken to transmit Data frames and receive corresponding acknowledgments.

10.4.10.1 TRANSMIT IDLE

In this state, the transmitter is waiting for the data link to be established between the local device and the peer device. The transmitter waits to be notified that a peer device is ready to communicate.

ConnectionEstablishedXON

If the Connection State Machine is in the CONNECTED state and ReceptionBlocked is equal to NOT_BLOCKED,

then call SendFrame to transmit a HeartbeatXON frame; set TxSequenceNumber to zero; set HeartbeatTimer to zero; and enter the TRANSMIT BLOCKED state.

ConnectionEstablishedXOFF

If the Connection State Machine is in the CONNECTED state and ReceptionBlocked is equal to ALMOST_BLOCKED or BLOCKED,

then call SendFrame to transmit a HeartbeatXOFF frame; set TxSequenceNumber to zero; and enter the TRANSMIT BLOCKED state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

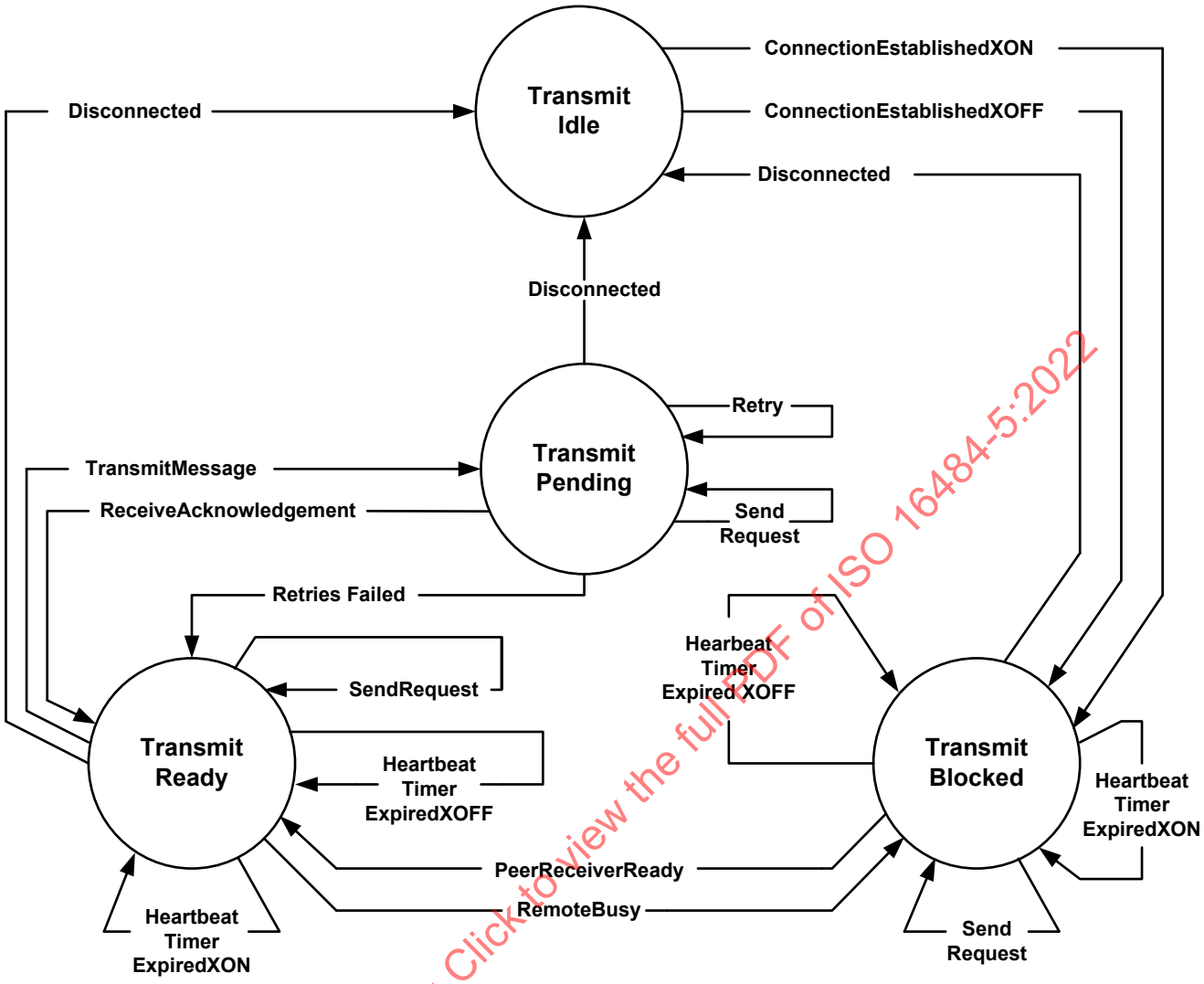


Figure 10-3. Point-To-Point Transmission State Machine.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

10.4.10.2 TRANSMIT BLOCKED

In this state, the peer device has indicated that it is not ready to receive Data frames. The local device may have data ready to transmit. The local device periodically transmits a Heartbeat frame to maintain the data link and waits for the peer device to become ready to receive data or for the termination of the data link.

SendRequest

If a DL-UNITDATA.request primitive is received,

then queue the request for later transmission and enter the TRANSMIT BLOCKED state.

PeerReceiverReady

If TransmissionBlocked is equal to FALSE,

then enter the TRANSMIT READY state.

Disconnected

If the Connection State Machine is in the DISCONNECTED state,

then enter the TRANSMIT IDLE state.

© ISO 2022 – All rights reserved

HeartbeatTimerExpiredXON
If HeartbeatTimer is greater than $\mathbf{T_{heartbeat}}$ and ReceptionBlocked is equal to NOT_BLOCKED,

then call SendFrame to transmit a HeartbeatXON frame; set HeartbeatTimer to zero; and enter the TRANSMIT BLOCKED state.

HeartbeatTimerExpiredXOFF
If HeartbeatTimer is greater than $\mathbf{T_{heartbeat}}$ and ReceptionBlocked is equal to BLOCKED or ALMOST_BLOCKED,

then call SendFrame to transmit a HeartbeatXOFF frame; set HeartbeatTimer to zero; and enter the TRANSMIT BLOCKED state.

10.4.10.3 TRANSMIT READY

In this state, the peer device has indicated its readiness to receive Data frames, but the local device has no data ready to transmit. The local device periodically transmits a Heartbeat frame to maintain the data link and waits for a local request to transmit data or for the termination of the data link.

Disconnected
If the Connection State Machine is in the DISCONNECTED state,

then enter the TRANSMIT IDLE state.

SendRequest
If a DL-UNITDATA.request primitive is received,

then queue the request for later transmission and enter the TRANSMIT READY state.

TransmitMessage
If the transmit queue is not empty and TransmissionBlocked is equal to FALSE,

then call SendFrame to transmit the frame at the head of the queue using a Data frame type (Data 0 or Data 1) that indicates TxSequenceNumber; set RetryCount to zero; set ResponseTimer to zero; and enter the TRANSMIT PENDING state.

RemoteBusy
If TransmissionBlocked is equal to TRUE,

then enter the TRANSMIT BLOCKED state.

HeartbeatTimerExpiredXON
If HeartbeatTimer is greater than $\mathbf{T_{heartbeat}}$ and ReceptionBlocked is equal to NOT_BLOCKED,

then call SendFrame to transmit a HeartbeatXON frame; set HeartbeatTimer to zero; and enter the TRANSMIT READY state.

HeartbeatTimerExpiredXOFF
If HeartbeatTimer is greater than $\mathbf{T_{heartbeat}}$ and ReceptionBlocked is equal to BLOCKED or ALMOST_BLOCKED,

then call SendFrame to transmit a HeartbeatXOFF frame; set HeartbeatTimer to zero; and enter the TRANSMIT READY state.

10.4.10.4 TRANSMIT PENDING

In this state, the local device has transmitted a Data frame to the peer device and is waiting for an acknowledgment from the peer device.

Disconnected
If the Connection State Machine is in the DISCONNECTED state,

then enter the TRANSMIT IDLE state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

SendRequest
If a DL-UNITDATA.request primitive is received,

then queue the request for later transmission and enter the TRANSMIT PENDING state.

ReceiveAcknowledgment
If TxSequenceNumber is equal to 0 and Ack0Received is equal to TRUE or if TxSequenceNumber is equal to 1 and Ack1Received is equal to TRUE,

then set TxSequenceNumber = 1 - TxSequenceNumber; set Ack0Received to FALSE; set Ack1Received to FALSE; and enter the TRANSMIT READY state.

Retry
If RetryCount is less than $\mathbf{N}_{\mathbf{retries}}$ and either

(a) TxSequenceNumber is equal to 0 and Nak0Received is equal to TRUE or
(b) TxSequenceNumber is equal to 1 and Nak1Received is equal to TRUE or
(c) ResponseTimer is greater than $\mathbf{T}_{\mathbf{response}}$,

then set RetryCount to RetryCount + 1; set Nak0Received to FALSE; set Nak1Received to FALSE; set ResponseTimer to zero; call SendFrame to retransmit the Data frame; and enter the TRANSMIT PENDING state.

RetriesFailed
If RetryCount is equal to $\mathbf{N}_{\mathbf{retries}}$, and either

(a) TxSequenceNumber is equal to 0 and Nak0Received is equal to TRUE or
(b) TxSequenceNumber is equal to 1 and Nak1Received is equal to TRUE or
(c) ResponseTimer is greater than $\mathbf{T}_{\mathbf{response}}$,

then set RetryCount to 0; set Nak0Received to FALSE; set Nak1Received to FALSE; set ResponseTimer to zero; and enter the TRANSMIT READY state.

10.4.11 Reception State Machine

The operation of the Reception State Machine is described in this clause and is depicted in Figure 10-4.

10.4.11.1 RECEIVE IDLE

In this state, the receiver is waiting for the data link to be established between the local device and the peer device. The receiver waits to be notified that a peer device is ready to communicate.

ConnectionEstablished
If the Connection State Machine is in the CONNECTED state,

then set RxSequenceNumber to zero and enter the RECEIVE READY state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

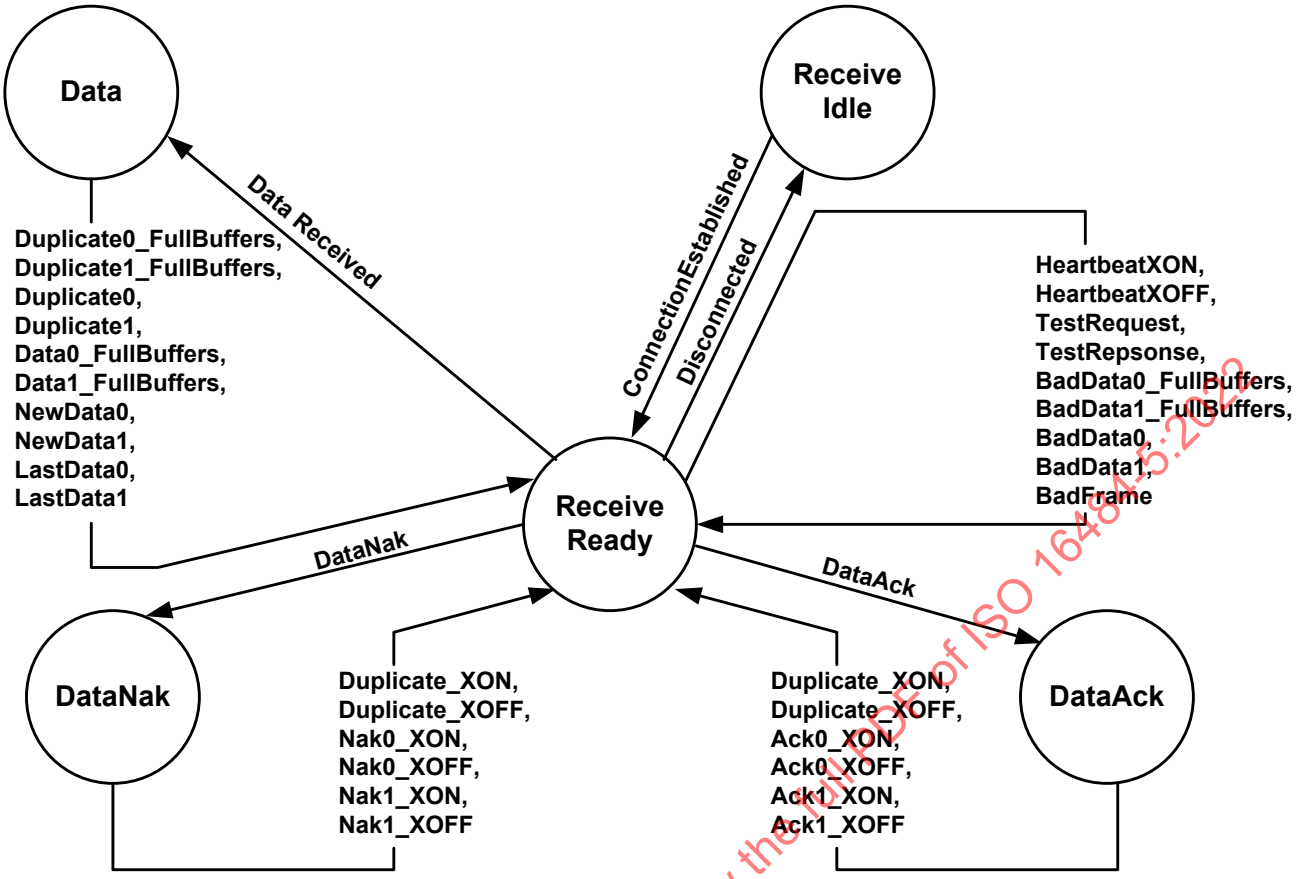


Figure 10-4. Point-To-Point Reception State Machine.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

10.4.11.2 RECEIVE READY

In this state, the device is ready to receive frames from the peer device.

DataReceived

If ReceivedValidFrame is TRUE and FrameType is equal to Data 0 or Data 1,

then set ReceivedValidFrame to FALSE; set InactivityTimer to zero; and enter the DATA state.

DataAck

If ReceivedValidFrame is TRUE and FrameType is equal to Data Ack 0 XOFF, Data Ack 0 XON, Data ACK 1 XOFF, or Data Ack 1 XON,

then set ReceivedValidFrame to FALSE; set InactivityTimer to zero; and enter the DATA ACK state.

DataNak

If ReceivedValidFrame is TRUE and FrameType is equal to Data Nak 0 XOFF, Data Nak 0 XON, Data Nak 1 XOFF, or Data Nak 1 XON,

then set ReceivedValidFrame to FALSE; set InactivityTimer to zero; and enter the DATA NAK state.

HeartbeatXON

If ReceivedValidFrame is TRUE and FrameType is equal to Heartbeat XON,

then set TransmissionBlocked to FALSE; set InactivityTimer to zero; set ReceivedValidFrame to FALSE; and enter the RECEIVE READY state.

© ISO 2022 – All rights reserved

HeartbeatXOFF
If ReceivedValidFrame is TRUE and FrameType is equal to Heartbeat XOFF,

then set TransmissionBlocked to TRUE; set InactivityTimer to zero; set ReceivedValidFrame to FALSE; and enter the RECEIVE READY state.

TestRequest
If ReceivedValidFrame is TRUE and FrameType is equal to Test_Request,

then call SendFrame to transmit the Test_Response; set InactivityTimer to zero; set ReceivedValidFrame to FALSE; and enter the RECEIVE READY state.

TestResponse
If ReceivedValidFrame is TRUE and FrameType is equal to Test_Response,

then issue a DL-UNITDATA.indication conveying the Test_Response data; set InactivityTimer to zero; set ReceivedValidFrame to FALSE; and enter the RECEIVE READY state.

BadData0_FullBuffers
If ReceivedInvalidFrame is TRUE and FrameType is equal to Data 0 and ReceptionBlocked is equal to BLOCKED,

then discard the frame; set InactivityTimer to zero; call SendFrame to transmit a Data Nak 0 XOFF frame; set ReceivedInvalidFrame to FALSE; and enter the RECEIVE READY state.

BadData1_FullBuffers
If ReceivedInvalidFrame is TRUE and FrameType is equal to Data 1 and ReceptionBlocked is equal to BLOCKED,

then discard the frame; set InactivityTimer to zero; call SendFrame to transmit a Data Nak 1 XOFF frame; set ReceivedInvalidFrame to FALSE; and enter the RECEIVE READY state.

BadData0
If ReceivedInvalidFrame is TRUE and FrameType is equal to Data 0 and ReceptionBlocked is equal to NOT_BLOCKED or ALMOST_BLOCKED,

then discard the frame; set InactivityTimer to zero; call SendFrame to transmit a Data Nak 0 XON frame; set ReceivedInvalidFrame to FALSE; and enter the RECEIVE READY state.

BadData1
If ReceivedInvalidFrame is TRUE and FrameType is equal to Data 1 and ReceptionBlocked is equal to NOT_BLOCKED or ALMOST_BLOCKED,

then discard the frame; set InactivityTimer to zero; call SendFrame to transmit a Data Nak 1 XON frame; set ReceivedInvalidFrame to FALSE; and enter the RECEIVE READY state.

BadFrame
If ReceivedInvalidFrame is TRUE and FrameType is not equal to either Data 0 or Data 1,

then discard the frame; set InactivityTimer to zero; set ReceivedInvalidFrame to FALSE; and enter the RECEIVE READY state.

Disconnected
If the Connection State Machine is in the DISCONNECTED state,

then enter the RECEIVE IDLE state.

10.4.11.3 DATA

In this state the device has received a Data frame for processing.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Duplicate0_FullBuffers
If FrameType is equal to Data 0 and RxSequenceNumber is equal to 1 and ReceptionBlocked is equal to BLOCKED,

then discard the frame as a duplicate; call SendFrame to transmit a Data Ack 0 XOFF frame; and enter the RECEIVE READY state.

Duplicate1_FullBuffers
If FrameType is equal to Data 1 and RxSequenceNumber is equal to 0 and ReceptionBlocked is equal to BLOCKED,

then discard the frame as a duplicate; call SendFrame to transmit a Data Ack 1 XOFF frame; and enter the RECEIVE READY state.

Duplicate0
If FrameType is equal to Data 0 and RxSequenceNumber is equal to 1 and ReceptionBlocked is equal to NOT_BLOCKED or ALMOST_BLOCKED,

then discard the frame as a duplicate; call SendFrame to transmit a Data Ack 0 XON frame; and enter the RECEIVE READY state.

Duplicate1
If FrameType is equal to Data 1 and RxSequenceNumber is equal to 0 and ReceptionBlocked is equal to NOT_BLOCKED or ALMOST_BLOCKED,

then discard the frame as a duplicate; call SendFrame to transmit a Data Ack 1 XON frame; and enter the RECEIVE READY state.

Data0_FullBuffers
If FrameType is equal to Data 0 and RxSequenceNumber is equal to 0 and ReceptionBlocked is equal to BLOCKED,

then discard the frame for lack of space; call SendFrame to transmit a Data Nak 0 XOFF frame; and enter the RECEIVE READY state.

Data1_FullBuffers
If FrameType is equal to Data 1 and RxSequenceNumber is equal to 1 and ReceptionBlocked is equal to BLOCKED,

then discard the frame for lack of space; call SendFrame to transmit a Data Nak 1 XOFF frame; and enter the RECEIVE READY state.

NewData0
If FrameType is equal to Data 0 and RxSequenceNumber is equal to 0 and ReceptionBlocked is equal to NOT_BLOCKED,

then issue a DL-UNITDATA.indication conveying the data; call SendFrame to transmit a Data Ack 0 XON frame; set RxSequenceNumber to 1; and enter the RECEIVE READY state.

NewData1
If FrameType is equal to Data 1 and RxSequenceNumber is equal to 1 and ReceptionBlocked is equal to NOT_BLOCKED,

then issue a DL-UNITDATA.indication conveying the data; call SendFrame to transmit a Data Ack 1 XON frame; set RxSequenceNumber to 0; and enter the RECEIVE READY state.

LastData0
If FrameType is equal to Data 0 and RxSequenceNumber is equal to 0 and ReceptionBlocked is equal to ALMOST_BLOCKED,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then issue a DL-UNITDATA.indication conveying the data; call SendFrame to transmit a Data Ack 0 XOFF frame; set RxSequenceNumber to 1; and enter the RECEIVE READY state.

LastData1
If FrameType is equal to Data 1 and RxSequenceNumber is equal to 1 and ReceptionBlocked is equal to ALMOST_BLOCKED,

then issue a DL-UNITDATA.indication conveying the data; call SendFrame to transmit a Data Ack 1 XOFF frame; set RxSequenceNumber to 0; and enter the RECEIVE READY state.

10.4.11.4 DATA ACK

In this state the device has received a Data Ack frame for processing.

Duplicate_XON
If FrameType is equal to Data Ack 0 XON and TxSequenceNumber is equal to 1, or if FrameType is equal to Data Ack 1 XON and TxSequenceNumber is equal to 0,

then set TransmissionBlocked to FALSE and enter the RECEIVE READY state.

Duplicate_XOFF
If FrameType is equal to Data Ack 0 XOFF and TxSequenceNumber is equal to 1, or if FrameType is equal to Data Ack 1 XOFF and TxSequenceNumber is equal to 0,

then set TransmissionBlocked to TRUE and enter the RECEIVE READY state.

Ack0_XON
If FrameType is equal to Data Ack 0 XON and TxSequenceNumber is equal to 0,

then set Ack0Received to TRUE; set TransmissionBlocked to FALSE; and enter the RECEIVE READY state.

Ack0_XOFF
If FrameType is equal to Data Ack 0 XOFF and TxSequenceNumber is equal to 0,

then set Ack0Received to TRUE; set TransmissionBlocked to TRUE; and enter the RECEIVE READY state.

Ack1_XON
If FrameType is equal to Data Ack 1 XON and TxSequenceNumber is equal to 1,

then set Ack1Received to TRUE; set TransmissionBlocked to FALSE; and enter the RECEIVE READY state.

Ack1_XOFF
If FrameType is equal to Data Ack 1 XOFF and TxSequenceNumber is equal to 1,

then set Ack1Received to TRUE; set TransmissionBlocked to TRUE; and enter the RECEIVE READY state.

10.4.11.5 DATA NAK

In this state the device has received a Data Nak frame for processing.

Duplicate_XON
If FrameType is equal to Data Nak 0 XON and TxSequenceNumber is equal to 1, or if FrameType is equal to Data Nak 1 XON and TxSequenceNumber is equal to 0,

then set TransmissionBlocked to FALSE and enter the RECEIVE READY state.

Duplicate_XOFF
If FrameType is equal to Data Nak 0 XOFF and TxSequenceNumber is equal to 1, or if FrameType is equal to Data Nak 1 XOFF and TxSequenceNumber is equal to 0,

then set TransmissionBlocked to TRUE and enter the RECEIVE READY state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Nak0_XON
If FrameType is equal to Data Nak 0 XON and TxSequenceNumber is equal to 0,

then set Nak0Received to TRUE; set TransmissionBlocked to FALSE; and enter the RECEIVE READY state.

Nak0_XOFF
If FrameType is equal to Data Nak 0 XOFF and TxSequenceNumber is equal to 0,

then set Nak0Received to TRUE; set TransmissionBlocked to TRUE; and enter the RECEIVE READY state.

Nak1_XON
If FrameType is equal to Data Nak 1 XON and TxSequenceNumber is equal to 1,

then set Nak1Received to TRUE; set TransmissionBlocked to FALSE; and enter the RECEIVE READY state.

Nak1_XOFF
If FrameType is equal to Data Nak 1 XOFF and TxSequenceNumber is equal to 1,

then set Nak1Received to TRUE; set TransmissionBlocked to TRUE; and enter the RECEIVE READY state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

11 DATA LINK/PHYSICAL LAYERS: LonTalk (ISO/IEC 14908.1) LAN

This clause describes the transport of BACnet LSDUs using the services of the LonTalk protocol described in EIA/CEA-709.1-B-2002: "Control Network Protocol Specification". EIA/CEA-709.1-B-2002, as amended and extended by the Electronic Industries Alliance, is deemed to be included in this standard by reference. Persons desiring to implement BACnet in products containing the LonTalk Protocol may obtain an OEM license to do so, without cost, by contacting Echelon Corporation, San Jose, California.

11.1 The Use of ISO 8802-2 Logical Link Control (LLC)

Standard BACnet networks may pass BACnet link service data units (LSDUs) using the data link services of ISO 8802-2 LLC. A BACnet LSDU consists of an NPDU constructed as described in Clause 6. BACnet devices using LonTalk LAN technology shall conform to the requirements of LLC Class I, subject to the constraints specified in this clause. Class I LLC service consists of Type 1 LLC - Unacknowledged Connectionless-Mode service. LLC parameters shall be conveyed using the DL-UNITDATA primitives as described in the referenced standards.

In a LonTalk implementation, BACnet DL-UNITDATA primitives are mapped into the LonTalk Application Layer Interface. The mapping of these primitives onto the LonTalk Application layer primitives is described in Clause 11.3.

11.2 Parameters Required by the LLC Primitives

The DL-UNITDATA primitive requires source address, destination address, data, and priority parameters. Each source and destination address consists of a LonTalk address, link service access point (LSAP), and a message code (MC). The LonTalk address is a variable-length value determined by the configuration of the BACnet device, and the MC used to indicate a BACnet frame is the single-octet value X'4E'. Since the LonTalk message code identifies the BACnet network layer, the LSAP is not used. The data parameter is the NPDU from the network layer.

11.3 Mapping the LLC Services to the LonTalk Application Layer

The Type 1 Unacknowledged Connectionless LLC service, DL_UNITDATA.request shall map onto the LonTalk msg_send request primitive, while the DL_UNITDATA.indication shall map to the LonTalk msg_receive request primitive.

An LPDU longer than 228 octets cannot be conveyed via LonTalk.

11.4 Parameters Required by the Application Layer Primitives

The LonTalk Application layer primitives are msg_send and msg_receive. These convey the encoded LLC data using the destination LonTalk address described above in conjunction with the BACnet message code. The DL_UNITDATA.request primitive contains the following parameters:

```
DL_UNITDATA.request (
        destination_address,
        data,
        priority,
        message_code
        )
```

The 'destination_address' consists of any form of a LonTalk address except address format 2B (which is used exclusively for multicast acknowledgments and multicast responses). See Figure 6-4. The 'data' parameter specifies the LSDU to be transferred. The 'priority' parameter conveys the priority specified for the data unit. Any BACnet priority other than "Normal message" shall be sent using the LonTalk priority mechanism. The 'message_code' parameter shall be X'4E' for BACnet LPDUs.

LonTalk Authentication is not supported in BACnet.

LonTalk "UNACKD" and "UNACKD_RPT" are the only LonTalk services that shall be allowed within BACnet. The choice between these two LonTalk services and the repeat count for the "UNACKD_RPT" service shall be considered a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The DL_UNITDATA.indication primitive contains the following parameters:

DL_UNITDATA.indication (
source_address,
destination_address,
length,
data,
message_code,
priority
)

Except as noted below, the parameters in DL_UNITDATA.indication convey the same information as the parameters in DL_UNITDATA.request.

The 'source_address' always consists of address format 2A.

The 'length' indicates the number of octets contained in the 'data' parameter.

Figure 11-1 illustrates the format of a MPDU on a LonTalk BACnet network destined for a device on the same LonTalk BACnet network.

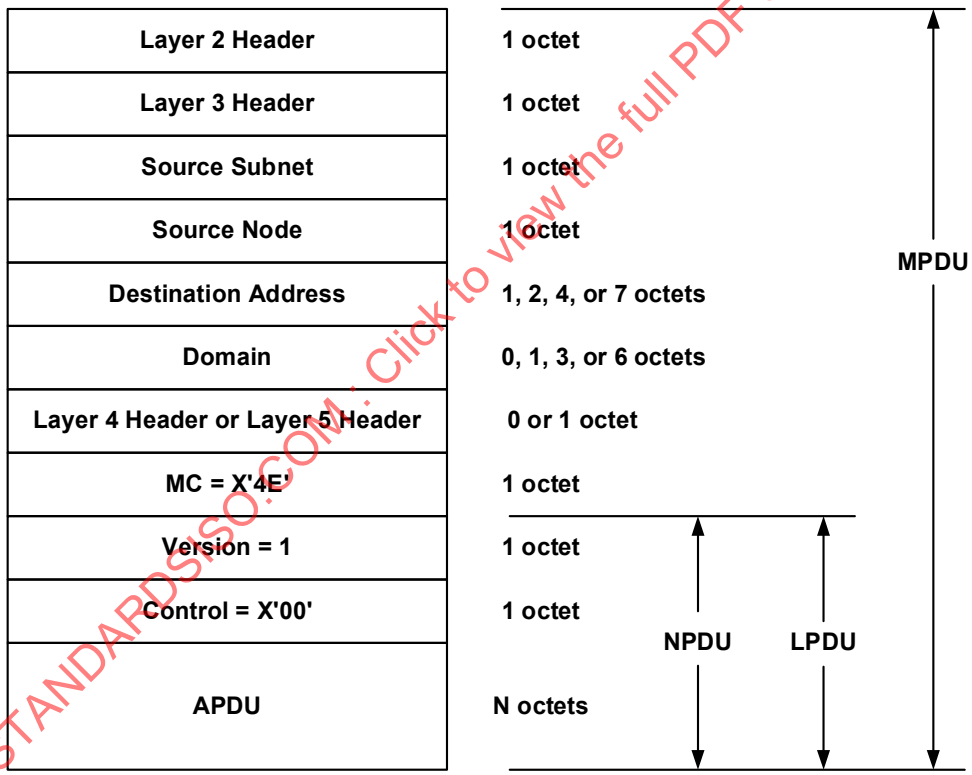


Figure 11-1. Format of an MPDU on a LonTalk network destined for a device on the same LonTalk BACnet network.

11.5 Physical Media

Any of the "Standard Channel Types" defined in the "LonMark™ Layer 1-6 Interoperability Guidelines" is acceptable. Transceivers built for this network technology shall follow the guidelines specified in the "LonMark™ Layer 1-6 Interoperability Guidelines." The most recent version of the "LonMark™ Layer 1-6 Interoperability Guidelines" as released by the LonMark™ Interoperability Association (currently version 3.3) shall apply.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12 MODELING CONTROL DEVICES AS A COLLECTION OF OBJECTS

The data structures used in a device to store information are a local matter. In order to exchange that information with another device using this protocol, there must be a "network-visible" representation of the information that is accessible from a standardized mechanism. An object-oriented approach has been adopted to provide this network-visible representation. This clause defines a set of standard object types. These object types define an abstract data structure that provides a framework for building the application layer services. The application layer services are designed, in part, to access and manipulate the properties of these standard object types as well as the properties of nonstandard object types. Mapping the effect of these services to the real data structures used in the device is a local matter. The number of instances of a particular object type that a device will support is also a local matter. Objects provide and accept information through a collection of properties that may use various datatypes. It is intended that the collection of object types and their properties defined in this standard be comprehensive, but implementors are free to define additional nonstandard object types or additional nonstandard properties of standard object types. This is the principal means for extending the standard as control technology develops. Innovative changes can be accommodated without waiting for changes in the standard. This extensibility could also be used to adapt this standard to other types of building services. See Clauses 23.3 and 23.4.

12.1 Object Characteristics and Requirements

All objects are subject to a common set of rules and requirements.

12.1.1 Identification of Objects

All objects are referenced by their Object_Identifier property and every object shall have an Object_Identifier property. The Object_Identifier is composed of two parts: an object type and an object instance (see Clause 20.2.14). Each object within a single BACnet device shall have a unique value for the Object_Identifier property. No object shall have an Object_Identifier with an instance number of 4194303. Object properties that contain values whose datatype is BACnetObjectIdentifier may use 4194303 as the instance number to indicate that the property is uninitialized, disabled, or unused, except where noted in individual clauses.

12.1.1.1 Object Type

Every object shall have an Object_Type property whose value shall be an enumeration equal to the object type portion of the Object_Identifier.

12.1.1.2 Object Name

Every object shall have an Object_Name property which is a character string. Object_Name properties shall be unique within the BACnet device that maintains them. The Object_Name string shall be at least one character in length and shall consist only of printable characters.

12.1.1.3 Device Instance

Each BACnet device shall have one and only one Device object instance. The object type of this object shall be DEVICE and the instance number shall be a internetwork-wide unique instance number. This is the Device Instance of the BACnet device. When an Object_Identifier is combined with the internetwork-wide unique Object_Identifier of the Device object of the BACnet device (having an object type DEVICE and the Device Instance), this provides a mechanism for referencing every object in the BACnet internetwork. Because of this requirement, every BACnet device shall provide a means of altering the Device Instance to assure uniqueness. The altered Device Instance shall be persistent across power failure and restart of the device.

12.1.1.4 Device Name

The Object_Name property of the Device object (the Device Name) shall be an internetwork-wide unique character string. Because of this requirement, every BACnet device shall provide a means of altering the Device Name to assure uniqueness. The altered Device Name shall be persistent across power failure and restart of the device.

12.1.1.4.1 Property_List

Every object shall have a Property_List property which is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.1.2 Object Type and Property Conformance

Not all object types defined in this standard need to be supported in order to conform to the standard. Other than the objects required by Clause 22.1.5, every BACnet device may choose any combination of standard and nonstandard object types and instances of those types that are appropriate to the device's application. In addition, object types may depend on a combination of particular properties. Some properties may be optionally implemented for particular objects as an implementor decision.

Each standard object type includes a property table summarizing the standard properties that are relevant to that object type. The property table lists each property that is defined for that object. Each property includes the property identifier, the datatype of the property, and one of the following: **O**, **R**, **W**. A conformance code of **O** indicates that the property is optional; **R** indicates that the property is required to be present and readable using BACnet services; **W** indicates that the property is required to be present, readable, and writable using BACnet services.

When a property is designated as required or **R**, this shall mean that the property is required to be present in all BACnet standard objects of that type. When a property is designated as optional or **O**, this shall mean that the property is not required to be present in all standard BACnet objects of that type. The value of **R** or **O** properties may be examined through the use of one or more of the ReadProperty services defined in this standard. Such **R** or **O** properties may also be writable at the implementor's option unless specifically prohibited in the text describing that particular standard object's property. When a property is designated as writable or **W**, this shall mean that the property is required to be present in all BACnet standard objects of that type and that the value of the property can be changed through the use of one or more of the WriteProperty services defined in this standard. The value of **W** properties may be examined through the use of one or more of the ReadProperty services defined in this standard. An **O** property, if present in a particular object, is not required to be writable unless specifically identified as such in the text describing that particular standard object's property.

12.1.3 Required and Optional Properties

A BACnet standard object shall support all required properties specified in the clause for its object type. It may support, in addition to these properties, any optional properties specified in the clause for its object type or properties not defined in the standard. A required property shall function as specified in the standard for each object of that type. If properties that are defined as optional in the standard are supported, then they shall function as specified in the standard. A required property for a given object type shall be present in all objects of that type. An optional property, if present in one object of a given type, need not be present in all objects of that type. A supported property, whether required or optional, shall return the datatype specified in the standard. A supported property, whether required or optional, is not required to be able to return the entire range of values for a datatype unless otherwise specified in the property description. Supported properties, whether required or optional, which do not return the entire range of values for a datatype when read or which restrict the range of values that may be written to the property, shall specify those restrictions for each such property in the protocol implementation conformance statement (PICS).

12.1.3.1 Required Properties Introduced in New Protocol Revisions

Required properties may change from revision to revision. Clients that depend on property requirements should take into account the Protocol_Revision, recognizing that a required property in some revision may not have existed in prior revisions.

12.1.4 Asymmetry in Property Values

In some devices, property values may be stored internally in a different form than indicated by the property datatype. For example, real numbers may be stored internally as integers, or with a more limited precision. This may result in the situation where a property value is changed by one of the WriteProperty services but a subsequent read returns a slightly different value. This behavior is acceptable as long as a "best effort" is made to store the written value specified.

Character strings may also be stored or processed internally using a different character set than indicated by the service request or may be modified when stored or processed in order to eliminate invalid characters. Any changes made to a character string as part of a write operation shall also be made to any character string received in another service request that is to be used for comparison or matching to a changed written character string.

For example, if a value written to an Object_Name property can be stored in a changed form, then a Who-Has service with an Object_Name parameter shall undergo the same changes such that a Who-Has containing the character string that was written to the Object_Name property will match the Object_Name value as it was stored in the object. Other

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

examples of related character string values that are subject to the above requirement are the Present_Value, Alarm_Values, and Fault_Values properties of a CharacterString Value object.

12.1.5 Array and List Properties

Some of the properties of certain BACnet objects need to represent a collection of data elements of the same type, rather than a single primitive data value or a complex datatype constructed from other datatypes. In some instances, the size of this collection of data elements is fixed, while in other instances the number of elements may be variable. In some cases the elements may need to be accessed individually or their order may be important. BACnet provides three forms of datatypes for properties that represent a collection of data elements of the same type: "BACnetARRAY", "BACnetLIST", and "BACnetARRAY of BACnetLIST".

12.1.5.1 Array Properties

A "BACnetARRAY" datatype is a structured datatype consisting of an ordered sequence of data elements, each having the same datatype. The components of an array property may be individually accessed (read or written) using an "array index," which is an unsigned integer value. An index of 0 (zero) shall specify the count of the number of data elements as datatype Unsigned. If the array index is omitted, it means that all of the elements of the array are to be accessed. An array index N, greater than zero, shall specify the Nth element in the sequence. When array properties are used in BACnet objects, the notation "BACnetARRAY[N] of datatype" shall mean an ordered sequence of N data elements, beginning with index 1, each of which has that datatype.

If the size of an array may be changed by writing to the array, then array element 0 shall be writable. If the value of array element 0 is decreased, for those arrays whose size is writable, the array shall be truncated and the elements of the array with an index greater than the new value of array element 0 are deleted. If the value of array element 0 is increased, for those arrays whose size is writable, the new elements of the array, with an index greater than the old value of array element 0, shall be created; the values that are assigned to those elements shall be a local matter except where otherwise specified. Where the size of an array is allowed to be changed, writing the entire array as a single property with a different number of elements than the existing array shall cause the array size to be changed. An attempt to write to an array element with an index greater than the size of the array shall result in an error and shall not cause the array to grow to accommodate the element. Arrays whose sizes are fixed by the standard shall not be resizable.

12.1.5.2 List Properties

A "BACnetLIST" datatype is a structured datatype consisting of a sequence of zero or more data elements, each having the same datatype. The length of each "BACnetLIST" may be variable. Unless specified for a particular use, no maximum size should be assumed for any "BACnetLIST" implementation. The notation "BACnetLIST of datatype" shall mean a sequence of zero or more data elements, each of which has the indicated type.

The difference between a "BACnetARRAY" property and a "BACnetLIST" property is that the elements of the array can be uniquely accessed by an array index while the elements of the "BACnetLIST" property can only be positionally accessed using the ReadRange service. Moreover, the number of elements in the BACnetARRAY may be ascertained by reading the array index 0, while the number of elements present in a "BACnetLIST" property can only be determined by reading the entire property value and performing a count.

In the context of ReadRange 'By Position', the ordering of "BACnetLIST" elements shall follow the conventions that the first element of the "BACnetLIST" shall be position 1, and positions 2, 3, 4 and greater shall correspond to list elements in strict sequence. The sequence of list elements shall follow the same ordering that those elements would appear in if the entire list was read using ReadProperty to read the entire list. The order of list elements when a list is written or modified is not required to be preserved by the device. However, assuming that the list has not been written or modified, repeated reading of list elements shall return those elements in the same order each time.

When adding items to a “BACnetLIST” via AddListElement, only unique items shall be added. For example a list {“A”, “B”} to which “A” is added will remain the same. However, particular properties may specify special criteria whereby only a portion of each list element is compared for uniqueness when checking for equality.

12.1.5.3 Array of List Properties

A "BACnetARRAY of BACnetLIST" datatype is a structured datatype consisting of a BACnetARRAY, where all array elements are a sequence of data elements having the same datatype except for array index 0 as note in Clause 12.1.5.1. When accessing a property of this datatype, then accessing an array element shall access the entire sequence of the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

array element. Only the AddListElement, RemoveListElement, and ReadRange services allow access to individual list elements within an array element of a property of this datatype. Array elements of this datatype shall behave as a BACnetLIST. Properties of this datatype require special encoding. See Clause 20.2.20.

12.1.6 Special Property Identifiers

Several object types defined in this clause - for example, the Command, Event Enrollment, Group, Loop, and Schedule object types - have one or more properties that are capable of referencing object properties. The property identifier component of these references shall not be any of the special property identifiers ALL, REQUIRED, or OPTIONAL. These are reserved for use in the ReadPropertyMultiple service or in services not defined in this standard.

12.1.7 Unspecified Dates and Times

Several object types defined in this clause have properties that are of type BACnetDateTime, and specify a specific point in time. These properties shall have an unspecified datetime value if the point in time is undefined or a specific datetime value if the point in time is specified.

There are a number of objects defined in this clause that have properties of the type BACnetDateRange, for example the Date_List property in the Calendar object, whose construct includes a startDate and an endDate. Both startDate and endDate may be an unspecified date or a specific date only. For purposes of comparing date ranges, the following logic shall be applied.

The use of an unspecified date in the startDate means "any date up to and including the endDate." The use of an unspecified date in the endDate means "any date after and including the startDate." The use of an unspecified date in both the startDate and the endDate means "any date" or "always."

Several object types defined in this clause have properties that contain timestamp values. If no event or operation has yet occurred, then timestamp values of type BACnetDateTime shall have an unspecified datetime value, and timestamp values of type Unsigned shall have a value of zero. If the event or operation has occurred, then the timestamp value shall have a specific datetime value, or a value greater than zero, respectively. If a device supports the Local_Date and Local_Time properties, then all timestamps created by the device shall use the BACnetDateTime form. For interoperability with products claiming a protocol revision prior to 12, it is recommended that implementations accept time values with trailing unspecified octets, such as unspecified hundredths of a second.

12.1.8 Reliability

Several object types defined in this clause have a property called "Reliability" that indicates the existence of fault conditions for the object. Reliability-evaluation is the process of determining the value for this property. Reliability-evaluation is separate from, but used by, event reporting and may be supported by objects that do not support event reporting.

The first stage of reliability-evaluation is internal to the object and is completely defined by the device's vendor. The second stage, which is only found in certain object types, is the application of a fault algorithm. Faults detected in the first stage shall take precedence over faults detected in the second stage. See Clause 13.4 for fault algorithm definitions and see the object type definitions to determine the fault algorithm supported by any particular object type. The different values that the Reliability property can take on are described below. Note that not all values are applicable to all object types.

NO_FAULT_DETECTED	The present value is reliable; that is, no other fault (enumerated below) has been detected.
NO_SENSOR	No sensor is connected to the Input object.
OVER_RANGE	The sensor connected to the Input is reading a value higher than the normal operating range. If the object is a Binary Input, this is possible when the Binary state is derived from an analog sensor or a binary input equipped with electrical loop supervision circuits.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

UNDER_RANGE	The sensor connected to the Input is reading a value lower than the normal operating range. If the object is a Binary Input, this is possible when the Binary Input is actually a binary state calculated from an analog sensor.
OPEN_LOOP	The connection between the defined object and the physical device is providing a value indicating an open circuit condition.
SHORTED_LOOP	The connection between the defined object and the physical device is providing a value indicating a short circuit condition.
NO_OUTPUT	No physical device is connected to the Output object.
PROCESS_ERROR	A processing error was encountered.
MULTI_STATE_FAULT	The FAULT_STATE, FAULT_LIFE_SAFETY or FAULT_CHARACTERSTRING fault algorithm has evaluated a fault condition. For details of this evaluation see the respective fault algorithms in Clause 13.4.
CONFIGURATION_ERROR	The object's properties are not in a consistent state.
COMMUNICATION_FAILURE	Proper operation of the object is dependent on communication with a remote sensor or device and communication with the remote sensor or device has been lost.
MONITORED_OBJECT_FAULT	Indicates that the monitored object is in fault.
REFERENCED_OBJECT_FAULT	A referenced object is in fault, but the referencing object is otherwise not in fault.
MEMBER_FAULT	Indicates that the set of referenced member objects includes one or more Status_Flags properties whose FAULT flag value is equal to TRUE.
TRIPPED	The end device, such as an actuator, is not responding to commands, prevented by a tripped condition or by being mechanically held open.
LAMP_FAILURE	Indicates that the lamp has failed in a physical lighting device.
ACTIVATION_FAILURE	Activation of changes has failed.
RENEW_FD_REGISTRATION_FAILURE	Renewing a foreign device registration with a BBMD has failed.
RENEW_DHCP_FAILURE	The attempt to obtain an IP address from a DHCP server has failed.
RESTART_AUTONEGOTIATION_FAILURE	The auto-negotiation algorithm has failed.
RESTART_FAILURE	The attempt to restart the port has failed.
PROPRIETARY_COMMAND_FAILURE	A proprietary command has failed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

FAULTS_LISTED	At least one fault indication is present in a BACnetLIST property. For details of the respective FAULT_LISTED fault algorithm see Clause 13.4.8.
MULTI_STATE_OUT_OF_RANGE	The value of the Present_Value of the Multi-state object is greater than the value of Number_of_States, for example, when the number of states becomes less than the current value of the Present_Value.
UNRELIABLE_OTHER	The controller has detected that the present value is unreliable, but none of the other conditions describe the nature of the problem. A generic fault other than those listed above has been detected, e.g., a Binary Input is not cycling as expected.

12.1.9 Handling of Invalid Data Encoding or Excessive Data Length

Some BACnet data values are encoded using standards such as ISO 10646 (UTF-8) or ANSI/IEEE-754 (floating point) that do not allow for all possible combinations of bit values. In addition, limitations may exist for the display or use of some BACnet data values in a particular implementation. If a service or result contains data that is properly tagged, but has either invalid value encoding or is too long, the receiver of the data shall handle the data in a way that is consistent with the service procedure described for the service.

Some services such as AcknowledgeAlarm, ConfirmedEventNotification or UnconfirmedEventNotification shall be processed using recovery mechanisms that are a local matter. Other confirmed services such as WriteProperty may respond with a Result(-) if recovery mechanisms are not used. If a confirmed service Error Type clause lists the error INVALID_DATA_ENCODING, then it is permitted to respond with a Result(-) if invalid data encoding is detected.

Detection of invalid data encoding and data value recovery mechanisms are not required, but are permitted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.2 Analog Input Object Type

The Analog Input object type defines a standardized object whose properties represent the externally visible characteristics of an analog input.

Analog Input objects that support intrinsic reporting shall apply the OUT_OF_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

The object and its properties are summarized in Table 12-2 and described in detail in this clause.

Table 12-2. Properties of the Analog Input Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Update_Interval	Unsigned	O
Units	BACnetEngineeringUnits	R
Min_Pres_Value	REAL	O
Max_Pres_Value	REAL	O
Resolution	REAL	O
COV_Increment	REAL	O[2]
Time_Delay	Unsigned	O[3,5]
Notification_Class	Unsigned	O[3,5]
High_Limit	REAL	O[3,5]
Low_Limit	REAL	O[3,5]
Deadband	REAL	O[3,5]
Limit_Enable	BACnetLimitEnable	O[3,5]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalREAL	O
Fault_High_Limit	REAL	O[8]
Fault_Low_Limit	REAL	O[8]
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.

[2] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.
[8] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[9] This property shall be present only if the device supports audit reporting.

12.2.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.2.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.2.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ANALOG_INPUT.

12.2.4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the input being measured. The Present_Value property shall be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.2.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.2.6 Device_Type

This property, of type CharacterString, is a text description of the physical device connected to the analog input. It will typically be used to describe the type of sensor attached to the analog input.

12.2.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an analog input. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.2.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

12.2.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical input that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the physical input and will not track changes to the physical input;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical input;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred in the physical input.

12.2.11 Update_Interval

This property, of type Unsigned, indicates the maximum period of time between updates to the Present_Value in hundredths of a second when the input is not overridden and not out-of-service.

12.2.12 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.2.13 Min_Pres_Value

This property, of type REAL, indicates the lowest number in engineering units that can be reliably obtained for the Present_Value property of this object.

12.2.14 Max_Pres_Value

This property, of type REAL, indicates the highest number in engineering units that can be reliably obtained for the Present_Value property of this object.

12.2.15 Resolution

This read-only property, of type REAL, indicates the smallest recognizable change in Present_Value in engineering units.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.2.16 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.2.17 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.18 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.2.19 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.20 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.21 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.22 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.23 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.2.24 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.2.25 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.2.26 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.2.27 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.2.28 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.2.29 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.2.30 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.2.31 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.2.32 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.2.33 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.2.34 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.2.35 Interface_Value

This read-only property, of type BACnetOptionalREAL, indicates the value, in engineering units, of the physical input. If the BACnet device is not capable of knowing the value of the physical input, then the value of this property shall be NULL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.2.36 Fault_High_Limit

This property, of type REAL, shall specify a limit that the Present_Value must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.2.37 Fault_Low_Limit

This property, of type REAL, shall specify a limit that the Present_Value must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.2.38 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.2.39 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.2.40 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.2.41 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.2.42 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.3 Analog Output Object Type

The Analog Output object type defines a standardized object whose properties represent the externally visible characteristics of an analog output.

Analog Output objects that support intrinsic reporting shall apply the OUT_OF_RANGE event algorithm.

The object and its properties are summarized in Table 12-3 and described in detail in this clause.

Table 12-3. Properties of the Analog Output Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	W
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Units	BACnetEngineeringUnits	R
Min_Pres_Value	REAL	O
Max_Pres_Value	REAL	O
Resolution	REAL	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	REAL	R
COV_Increment	REAL	O[1]
Time_Delay	Unsigned	O[2,4]
Notification_Class	Unsigned	O[2,4]
High_Limit	REAL	O[2,4]
Low_Limit	REAL	O[2,4]
Deadband	REAL	O[2,4]
Limit_Enable	BACnetLimitEnable	O[2,4]
Event_Enable	BACnetEventTransitionBits	O[2,4]
Acked_Transitions	BACnetEventTransitionBits	O[2,4]
Notify_Type	BACnetNotifyType	O[2,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Event_Detection_Enable	BOOLEAN	O[2,4]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[4]
Event_Algorithm_Inhibit	BOOLEAN	O[4,5]
Time_Delay_Normal	Unsigned	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[6]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalREAL	O
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[7,9,11]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[8,10]
Last_Command_Time	BACnetTimeStamp	O[8,10]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[10]
Audit_Level	BACnetAuditLevel	O[12]
Auditable_Operations	BACnetAuditOperationFlags	O[12]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[12]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[2] These properties are required if the object supports intrinsic reporting.
[3] Footnote removed.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[6] If this property is present, then the Reliability property shall be present.
[7] This property is required if the object supports the value source mechanism.
[8] These properties are required if the object supports the value source mechanism and is commandable.
[9] This property shall be present only if the object supports the value source mechanism.
[10] These properties shall be present only if the object supports the value source mechanism and is commandable.
[11] This property shall be writable as described in Clause 19.5.
[12] This property shall be present only if the device supports audit reporting.

12.3.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.3.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.3.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ANALOG_OUTPUT.

12.3.4 Present_Value (Commandable)

This property, of type REAL, indicates the current value, in engineering units, of the output.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.3.6 Device_Type

This property, of type CharacterString, is a text description of the physical device connected to the analog output. It will typically be used to describe the type of device attached to the analog output.

12.3.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an analog output. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.3.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

12.3.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical point that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the physical output;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical output;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred to the physical output;
(e) the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

12.3.11 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.3.12 Min_Pres_Value

This property, of type REAL, indicates the lowest number in engineering units that can be reliably used for the Present_Value property of this object.

12.3.13 Max_Pres_Value

This property, of type REAL, indicates the highest number in engineering units that can be reliably used for the Present_Value property of this object.

12.3.14 Resolution

This read-only property, of type REAL, indicates the smallest recognizable change in Present_Value in engineering units.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.3.15 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.3.16 Relinquish_Default

This property, of type REAL, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.3.17 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.3.18 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.19 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.3.20 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.21 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.22 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.23 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.24 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.3.25 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.3.26 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.3.27 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.3.28 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.3.29 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.3.30 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.3.31 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.3.32 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.3.33 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.3.34 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.3.35 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.3.36 Interface_Value

This read-only property, of type BACnetOptionalREAL, indicates the value, in engineering units, of the physical output. If the BACnet device is not capable of knowing the value of the physical output, then the value of this property shall be NULL.

12.3.37 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.3.38 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.3.39 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.3.40 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.3.41 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.3.42 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.3.43 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

For details on auditing and the use of this property, see Clause 19.6.3.

12.3.44 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.3.45 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.3.46 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.3.47 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4 Analog Value Object Type

The Analog Value object type defines a standardized object whose properties represent the externally visible characteristics of an analog value. An "analog value" is a control system parameter residing in the memory of the BACnet device.

Analog Value objects that support intrinsic reporting shall apply the OUT_OF_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

The object and its properties are summarized in Table 12-4 and described in detail in this clause.

Table 12-4. Properties of the Analog Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	R[4]
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Units	BACnetEngineeringUnits	R
Priority_Array	BACnetPriorityArray	O[1]
Relinquish_Default	REAL	O[1]
COV_Increment	REAL	O[2]
Time_Delay	Unsigned	O[3,6]
Notification_Class	Unsigned	O[3,6]
High_Limit	REAL	O[3,6]
Low_Limit	REAL	O[3,6]
Deadband	REAL	O[3,6]
Limit_Enable	BACnetLimitEnable	O[3,6]
Event_Enable	BACnetEventTransitionBits	O[3,6]
Acked_Transitions	BACnetEventTransitionBits	O[3,6]
Notify_Type	BACnetNotifyType	O[3,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[3,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Min_Pres_Value	REAL	O
Max_Pres_Value	REAL	O
Resolution	REAL	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Fault_High_Limit	REAL	O[9]
Fault_Low_Limit	REAL	O[9]
Current_Command_Priority	BACnetOptionalUnsigned	O[1]
Value_Source	BACnetValueSource	O[10,12,14]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[11,13]
Last_Command_Time	BACnetTimeStamp	O[11,13]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[13]
Audit_Level	BACnetAuditLevel	O[15]
Auditable_Operations	BACnetAuditOperationFlags	O[15]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[15]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if, and shall be present only if, Present_Value is commandable.
[2] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[3] These properties are required if the object supports intrinsic reporting.
[4] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[10] This property is required if the object supports the value source mechanism.
[11] These properties are required if the object supports the value source mechanism and is commandable.
[12] This property shall be present only if the object supports the value source mechanism.
[13] These properties shall be present only if the object supports the value source mechanism and is commandable.
[14] This property shall be writable as described in Clause 19.5.
[15] This property shall be present only if the device supports audit reporting.

12.4.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.4.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.4.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ANALOG_VALUE.

12.4.4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the analog value. Present_Value shall be optionally commandable. If Present_Value is commandable for a given object instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.4.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an analog value. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.4.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet device can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

12.4.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.4.10 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.11 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.4.12 Relinquish_Default

This property, of type REAL, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.4.13 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.4.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.4.16 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.17 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.18 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.19 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.4.21 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.4.22 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.23 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.4.24 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.4.25 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.4.26 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.4.27 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.4.28 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.4.29 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.4.30 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.31 Min_Pres_Value

This property, of type REAL, indicates the lowest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.4.32 Max_Pres_Value

This property, of type REAL, indicates the highest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.4.33 Resolution

This read-only property, of type REAL, indicates the smallest recognizable change in Present_Value in engineering units.

12.4.34 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.4.35 Fault_High_Limit

This property, of type REAL, shall specify a limit that the Present_Value must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.4.36 Fault_Low_Limit

This property, of type REAL, shall specify a limit that the Present_Value must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.4.37 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.4.38 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.4.39 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.4.40 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.41 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.4.42 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.4.43 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.4.44 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.4.45 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.4.46 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.4.47 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.5 Averaging Object Type

The Averaging object type defines a standardized object whose properties represent the externally visible characteristics of a value that is sampled periodically over a specified time interval. The Averaging object records the minimum, maximum and average value over the interval, and makes these values visible as properties of the Averaging object. The sampled value may be the value of any BOOLEAN, INTEGER, Unsigned, Enumerated or REAL property value of any object within the BACnet device in which the object resides. Optionally, the object property to be sampled may exist in a different BACnet device. The Averaging object shall use a "sliding window" technique that maintains a buffer of N samples distributed over the specified interval. Every (time interval/N) seconds a new sample is recorded displacing the oldest sample from the buffer. At this time, the minimum, maximum and average are recalculated. The buffer shall maintain an indication for each sample that permits the average calculation and minimum/maximum algorithm to determine the number of valid samples in the buffer.

The Averaging object type and its properties are summarized in Table 12-5 and described in detail in this clause.

Table 12-5. Properties of the Averaging Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Minimum_Value	REAL	R
Minimum_Value_Timestamp	BACnetDateTime	O
Average_Value	REAL	R
Variance_Value	REAL	O
Maximum_Value	REAL	R
Maximum_Value_Timestamp	BACnetDateTime	O
Description	CharacterString	O
Attempted_Samples	Unsigned	W[1]
Valid_Samples	Unsigned	R
Object_Property_Reference	BACnetDeviceObjectPropertyReference	R[1]
Window_Interval	Unsigned	W[1]
Window_Samples	Unsigned	W[1]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[2]
Auditable_Operations	BACnetAuditOperationFlags	O[2]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If any of these properties are written to using BACnet services, then all of the buffer samples shall become invalid, 'Attempted_Samples' shall become zero, 'Valid_Samples' shall become zero, 'Minimum_Value' shall become INF, 'Average_Value' shall become NaN and 'Maximum_Value' shall become -INF.

[2] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.5.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.5.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

© ISO 2022 – All rights reserved

12.5.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be AVERAGING.

12.5.4 Minimum_Value

This property, of type REAL, shall reflect the lowest value contained within the buffer window for the most recent 'Window_Samples' samples, or the actual number of samples ('Valid_Samples') if less than 'Window_Samples' samples have been taken. After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services, until a sample is taken, 'Minimum_Value' shall have the value **INF**.

12.5.5 Minimum_Value_Timestamp

This property, of type BACnetDateTime, indicates the date and time at which the value stored in Minimum_Value was sampled.

12.5.6 Average_Value

This property, of type REAL, shall reflect the average value contained within the buffer window for the most recent 'Window_Samples' samples, or the actual number of samples ('Valid_Samples') if less than 'Window_Samples' samples have been taken. The average shall be calculated by taking the arithmetic sum of all non-missed buffer samples and dividing by the number of non-missed buffer samples. After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services, until a sample is taken, 'Average_Value' shall have the value **NaN**.

12.5.7 Variance_Value

This property, of type REAL, shall reflect the variance value contained within the buffer window for the most recent 'Window_Samples' samples, or the actual number of samples ('Valid_Samples') if less than 'Window_Samples' samples have been taken. After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services, until a sample is taken, 'Variance_Value' shall have the value **NaN**.

12.5.8 Maximum_Value

This property, of type REAL, shall reflect the highest value contained within the buffer window for the most recent 'Window_Samples' samples, or the actual number of samples ('Valid_Samples') if less than 'Window_Samples' samples have been taken. After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services, until a sample is taken, 'Maximum_Value' shall have the value **-INF**.

12.5.9 Maximum_Value_Timestamp

This property, of type BACnetDateTime, indicates the date and time at which the value stored in Maximum_Value was sampled.

12.5.10 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.5.11 Attempted_Samples

This property, of type Unsigned, indicates the number of samples that have been attempted to be collected for the current window. The only acceptable value that may be written to this property shall be zero. If 'Attempted_Samples' is less than 'Window_Samples' then a period of time less than 'Window_Interval' has elapsed since a device restart, or 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' have been written to using BACnet services. The number of missed samples in the current window can be calculated by subtracting 'Valid_Samples' from 'Attempted_Samples.' After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services, until a sample is taken, 'Attempted_Samples' shall have the value zero.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.5.12 Valid_Samples

This read-only property, of type Unsigned, indicates the number of samples that have been successfully collected for the current window. This value can be used to determine whether any of the samples in the current 'Window_Interval' are missing. The number of missed samples in the current window can be calculated by subtracting 'Valid_Samples' from 'Attempted_Samples.' A result greater than zero indicates the number of samples that encountered an error when the sample was being recorded. After a device restart, or after 'Attempted_Samples', 'Object_Property_Reference', 'Window_Samples' or 'Window_Interval' are written to using BACnet services until a sample is taken, 'Valid_Samples' shall have the value zero.

12.5.13 Object_Property_Reference

This property, of type BACnetDeviceObjectPropertyReference, shall identify the object and property whose value is to be sampled during the 'Window_Interval'. The object referenced may be located within the device containing the Averaging object, or optionally the Averaging object may support the referencing of object properties in other devices. External references may be restricted to a particular set of BACnet devices. The value of the property referenced in the referenced object property must represent a numeric quantity. All sampled data shall be converted to REAL for calculation purposes. BOOLEAN FALSE shall be considered to be zero and TRUE shall be considered to be one. Enumerated datatypes shall be treated as Unsigned values. If an implementation supports writing to 'Object_Property_Reference', then if 'Object_Property_Reference' is written to using BACnet services, then all of the buffer samples shall become invalid, 'Attempted_Samples' shall become zero, 'Valid_Samples' shall become zero, 'Minimum_Value' shall become INF, 'Average_Value' shall become NaN and 'Maximum_Value' shall become -INF.

12.5.14 Window_Interval

This property, of type Unsigned, shall indicate the period of time in seconds over which the minimum, maximum and average values are calculated. The minimum acceptable value for 'Window_Interval' shall be a local matter. Every 'Window_Interval' divided by 'Window_Samples' seconds a new sample shall be taken by reading the value of the property referenced by the 'Object_Property_Reference'. Whether the sample represents an instantaneous "snapshot" or a continuously calculated sample shall be a local matter. If 'Window_Interval' is written to using BACnet services, then all of the buffer samples shall become invalid, 'Attempted_Samples' shall become zero, 'Valid_Samples' shall become zero, 'Minimum_Value' shall become INF, 'Average_Value' shall become NaN and 'Maximum_Value' shall become -INF.

12.5.15 Window_Samples

This property, of type Unsigned, shall indicate the number of samples to be taken during the period of time specified by the 'Window_Interval' property. 'Window_Samples' must be greater than zero and all implementations shall support at least 15 samples. Every 'Window_Interval' divided by 'Window_Samples' seconds a new sample shall be taken by reading the value of the property referenced by the 'Object_Property_Reference'. Whether the sample represents an instantaneous "snapshot" or a continuously calculated sample shall be a local matter. If 'Window_Samples' is written to using BACnet services, then all of the buffer samples shall become invalid, 'Attempted_Samples' shall become zero, 'Valid_Samples' shall become zero, 'Minimum_Value' shall become INF, 'Average_Value' shall become NaN and 'Maximum_Value' shall become -INF.

12.5.16 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.5.17 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.5.18 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.5.19 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.5.20 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.5.21 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.6 Binary Input Object Type

The Binary Input object type defines a standardized object whose properties represent the externally visible characteristics of a binary input. A "binary input" is a physical device or hardware input that can be in only one of two distinct states. In this description, those states are referred to as ACTIVE and INACTIVE. A typical use of a binary input is to indicate whether a particular piece of mechanical equipment, such as a fan or pump, is running or idle. The state ACTIVE corresponds to the situation when the equipment is on or running, and INACTIVE corresponds to the situation when the equipment is off or idle.

In some applications, electronic circuits may reverse the relationship between the application-level logical states ACTIVE and INACTIVE and the physical state of the underlying hardware. For example, a normally open relay contact may result in an ACTIVE state when the relay is energized, while a normally closed relay contact may result in an INACTIVE state when the relay is energized. The Binary Input object provides for this possibility by including a Polarity property. See Clauses 12.6.4 and 12.6.11.

Binary Input objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

The object and its properties are summarized in Table 12-6 and described in detail in this clause.

Table 12-6. Properties of the Binary Input Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetBinaryPV	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Polarity	BACnetPolarity	R
Inactive_Text	CharacterString	O[2]
Active_Text	CharacterString	O[2]
Change_Of_State_Time	BACnetDateTime	O[3]
Change_Of_State_Count	Unsigned	O[3]
Time_Of_State_Count_Reset	BACnetDateTime	O[3]
Elapsed_Active_Time	Unsigned32	O[4]
Time_Of_Active_Time_Reset	BACnetDateTime	O[4]
Time_Delay	Unsigned	O[5,7]
Notification_Class	Unsigned	O[5,7]
Alarm_Value	BACnetBinaryPV	O[5,7]
Event_Enable	BACnetEventTransitionBits	O[5,7]
Acked_Transitions	BACnetEventTransitionBits	O[5,7]
Notify_Type	BACnetNotifyType	O[5,7]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[5,7]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[7]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[7]
Event_Detection_Enable	BOOLEAN	O[5,7]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[7]
Event_Algorithm_Inhibit	BOOLEAN	O[7,8]
Time_Delay_Normal	Unsigned	O[7]
Reliability_Evaluation_Inhibit	BOOLEAN	O[9]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalBinaryPV	O
Audit_Level	BACnetAuditLevel	O[10]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Auditable_Operations	BACnetAuditOperationFlags	O[10]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] If one of the optional properties Inactive_Text or Active_Text is present, then both of these properties shall be present.
[3] If one of the optional properties Change_Of_State_Time, Change_Of_State_Count, or Time_Of_State_Count_Reset is present, then all of these properties shall be present.
[4] If one of the optional properties Elapsed_Active_Time or Time_Of_Active_Time_Reset is present, then both of these properties shall be present.
[5] These properties are required if the object supports intrinsic reporting.
[6] Footnote removed.
[7] These properties shall be present only if the object supports intrinsic reporting.
[8] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[9] If this property is present, then the Reliability property shall be present.
[10] This property shall be present only if the device supports audit reporting.

12.6.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.6.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.6.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be BINARY_INPUT.

12.6.4 Present_Value

This property, of type BACnetBinaryPV, reflects the logical state of the Binary Input. The logical state of the Input shall be either INACTIVE or ACTIVE. The relationship between the Present_Value and the physical state of the Input is determined by the Polarity property. The possible states are summarized in Table 12-7.

Table 12-7. BACnet Polarity Relationships

Present_Value	Polarity	Physical State of Input	Physical State of Device
INACTIVE	NORMAL	OFF or INACTIVE	not running
ACTIVE	NORMAL	ON or ACTIVE	Running
INACTIVE	REVERSE	ON or ACTIVE	not running
ACTIVE	REVERSE	OFF or INACTIVE	running

The Present_Value property shall be writable when Out_Of_Service is TRUE. When Out_Of_Service is TRUE, changes in the Polarity property shall not cause changes in Present_Value. When Out_Of_Service is FALSE, a change of the Polarity property shall alter Present_Value accordingly.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.6.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.6.6 Device_Type

This property, of type CharacterString, is a text description of the physical device connected to the binary input. It will typically be used to describe the type of device attached to the binary input.

12.6.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a binary input. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.6.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.6.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

12.6.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical input that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the physical input and will not track changes to the physical input;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical input;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred in the physical input.

12.6.11 Polarity

This property, of type BACnetPolarity, indicates the relationship between the physical state of the Input and the logical state represented by the Present_Value property. If the Polarity property is NORMAL, then the ACTIVE state of the Present_Value property is also the ACTIVE or ON state of the physical Input as long as Out_Of_Service is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the Present_Value property is the INACTIVE or OFF state of the physical Input as long as Out_Of_Service is FALSE. See Table 12-7. Therefore, when Out_Of_Service is FALSE for a constant physical input state, a change in the Polarity property shall produce a change in the Present_Value property. If Out_Of_Service is TRUE, then the Polarity property shall have no effect on the Present_Value property.

12.6.12 Inactive_Text

This property, of type CharacterString, characterizes the intended effect of the INACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the INACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

12.6.13 Active_Text

This property, of type CharacterString, characterizes the intended effect of the ACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the ACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

12.6.14 Change_Of_State_Time

This property, of type BACnetDateTime, represents the date and time at which the most recent change of state occurred. A "change of state" shall be defined as any event that alters the Present_Value property.

12.6.15 Change_Of_State_Count

This property, of type Unsigned, represents the number of times that the Present_Value property has changed state since the date and time indicated by the Time_Of_State_Count_Reset property.

When this property is set to zero, the Time_Of_State_Count_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_State_Count_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

The Change_Of_State_Count property shall have a range of 0-65535 or greater.

12.6.16 Time_Of_State_Count_Reset

This property, of type BACnetDateTime, indicates the date and time at which the counting of state changes, indicated by the Change_Of_State_Count property, started.

If the Change_Of_State_Count property accepts writes of non-zero values, then the Time_Of_State_Count_Reset property shall be writable, otherwise it shall be read-only.

12.6.17 Elapsed_Active_Time

This property, of type Unsigned32, represents the accumulated number of seconds that the Present_Value property has had the value ACTIVE since the date and time indicated by the Time_Of_Active_Time_Reset property.

When this property is set to zero, the Time_Of_Active_Time_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_Active_Time_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

12.6.18 Time_Of_Active_Time_Reset

This property, of type BACnetDateTime, indicates the date and time at which the accumulation of active time, indicated by the Elapsed_Active_Time property, has been started.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the Elapsed_Active_Time property accepts writes of non-zero values, then the Time_Of_Active_Time_Reset property shall be writable, otherwise it shall be read-only.

12.6.19 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.6.20 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.6.21 Alarm_Value

This property is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.6.22 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.6.23 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.6.24 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.6.25 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.6.26 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.6.27 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.6.28 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.6.29 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.6.30 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.6.31 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.6.32 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.6.33 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.6.34 Interface_Value

This read-only property, of type BACnetOptionalBinaryPV, reflects the logical state of the physical input. If the BACnet device is not capable of knowing the logical state of the physical input, then the value of this property shall be NULL. Otherwise, the logical state of this property shall be either INACTIVE or ACTIVE. The relationship between this property and the physical state of the input is determined by the Polarity property. The possible states are summarized in Table 12-7.

12.6.35 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.6.36 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.6.37 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.6.38 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.6.39 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.7 Binary Output Object Type

The Binary Output object type defines a standardized object whose properties represent the externally visible characteristics of a binary output. A "binary output" is a physical device or hardware output that can be in only one of two distinct states. In this description, those states are referred to as ACTIVE and INACTIVE. A typical use of a binary output is to switch a particular piece of mechanical equipment, such as a fan or pump, on or off. The state ACTIVE corresponds to the situation when the equipment is on or running, and INACTIVE corresponds to the situation when the equipment is off or idle.

In some applications, electronic circuits may reverse the relationship between the application-level logical states, ACTIVE and INACTIVE, and the physical state of the underlying hardware. For example, a normally open relay contact may result in an ACTIVE state (device energized) when the relay is energized, while a normally closed relay contact may result in an ACTIVE state (device energized) when the relay is not energized. The Binary Output object provides for this possibility by including a Polarity property. See Clauses 12.7.4 and 12.7.11.

Binary Output objects that support intrinsic reporting shall apply the COMMAND_FAILURE event algorithm.

The object and its properties are summarized in Table 12-8 and described in detail in this clause.

Table 12-8. Properties of the Binary Output Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetBinaryPV	W
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Polarity	BACnetPolarity	R
Inactive_Text	CharacterString	O[1]
Active_Text	CharacterString	O[1]
Change_Of_State_Time	BACnetDateTime	O[2]
Change_Of_State_Count	Unsigned	O[2]
Time_Of_State_Count_Reset	BACnetDateTime	O[2]
Elapsed_Active_Time	Unsigned32	O[3]
Time_Of_Active_Time_Reset	BACnetDateTime	O[3]
Minimum_Off_Time	Unsigned32	O
Minimum_On_Time	Unsigned32	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	BACnetBinaryPV	R
Time_Delay	Unsigned	O[4,6]
Notification_Class	Unsigned	O[4,6]
Feedback_Value	BACnetBinaryPV	O[4]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalBinaryPV	O
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[9,11,13]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[10,12]
Last_Command_Time	BACnetTimeStamp	O[10,12]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[12]
Audit_Level	BACnetAuditLevel	O[14]
Auditable_Operations	BACnetAuditOperationFlags	O[14]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[14]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If one of the optional properties Inactive_Text or Active_Text is present, then both of these properties shall be present.
[2] If one of the optional properties Change_Of_State_Time, Change_Of_State_Count, or Time_Of_State_Count_Reset is present, then all of these properties shall be present.
[3] If one of the optional properties Elapsed_Active_Time or Time_Of_Active_Time_Reset is present, then both of these properties shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] This property is required if the object supports the value source mechanism.
[10] These properties are required if the object supports the value source mechanism and is commandable.
[11] This property shall be present only if the object supports the value source mechanism.
[12] These properties shall be present only if the object supports the value source mechanism and is commandable.
[13] This property shall be writable as described in Clause 19.5.
[14] This property shall be present only if the device supports audit reporting.

12.7.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.7.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.7.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be BINARY_OUTPUT.

12.7.4 Present_Value (Commandable)

This property, of type BACnetBinaryPV, reflects the logical state of the Binary Output. The logical state of the output shall be either INACTIVE or ACTIVE. The relationship between the Present_Value and the physical state of the output is determined by the Polarity property. The possible states are summarized in Table 12-9.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 12-9. BACnet Polarity Relationships

Present_Value	Polarity	Physical State of Output	Physical State of Device
INACTIVE	NORMAL	OFF or INACTIVE	not running
ACTIVE	NORMAL	ON or ACTIVE	running
INACTIVE	REVERSE	ON or ACTIVE	not running
ACTIVE	REVERSE	OFF or INACTIVE	running

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.7.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.7.6 Device_Type

This property, of type CharacterString, is a text description of the physical device connected to the binary output. It will typically be used to describe the type of device attached to the binary output.

12.7.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a binary output. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT — Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN — Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output. Otherwise, the value is logical FALSE (0).

OUT_OF_SERVICE — Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.7.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.7.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.7.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical point that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the physical output;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical output;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred to the physical output;
(e) the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

12.7.11 Polarity

This property, of type BACnetPolarity, indicates the relationship between the physical state of the output and the logical state represented by the Present_Value property. If the Polarity property is NORMAL, then the ACTIVE state of the Present_Value property is also the ACTIVE or ON state of the physical output as long as Out_Of_Service is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the Present_Value property is the INACTIVE or OFF state of the physical output as long as Out_Of_Service is FALSE. See Table 12-9. If Out_Of_Service is TRUE, then the Polarity property shall have no effect on the physical output state.

12.7.12 Inactive_Text

This property, of type CharacterString, characterizes the intended effect of the INACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the INACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

12.7.13 Active_Text

This property, of type CharacterString, characterizes the intended effect of the ACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the ACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

12.7.14 Change_Of_State_Time

This property, of type BACnetDateTime, represents the date and time at which the most recent change of state occurred. A "change of state" shall be defined as any event that alters the Present_Value property.

12.7.15 Change_Of_State_Count

This property, of type Unsigned, represents the number of times that the Present_Value property has changed state since the date and time indicated by the Time_Of_State_Count_Reset property.

When this property is set to zero, the Time_Of_State_Count_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_State_Count_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

The Change_Of_State_Count property shall have a range of 0-65535 or greater.

12.7.16 Time_Of_State_Count_Reset

This property, of type BACnetDateTime, indicates the date and time at which the counting of state changes, indicated by the Change_Of_State_Count property, started.

If the Change_Of_State_Count property accepts writes of non-zero values, then the Time_Of_State_Count_Reset property shall be writable, otherwise it shall be read-only.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.7.17 Elapsed_Active_Time

This property, of type Unsigned32, represents the accumulated number of seconds that the Present_Value property or the Feedback_Value property has had the value ACTIVE since the date and time indicated by the Time_Of_Active_Time_Reset property. Whether Present_Value or Feedback_Value is used as the indicator for the calculation of the Elapsed_Active_Time is a local matter.

When this property is set to zero, the Time_Of_Active_Time_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_Active_Time_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

12.7.18 Time_Of_Active_Time_Reset

This property, of type BACnetDateTime, indicates the date and time at which the accumulation of active time, indicated by the Elapsed_Active_Time property, has been started.

If the Elapsed_Active_Time property accepts writes of non-zero values, then the Time_Of_Active_Time_Reset property shall be writable, otherwise it shall be read-only.

12.7.19 Minimum_Off_Time

This property, of type Unsigned32, represents the minimum number of seconds that the Present_Value shall remain in the INACTIVE state after a write to the Present_Value property causes that property to assume the INACTIVE state.

The mechanism by which this is accomplished is described in Clause 19.2.3.

12.7.20 Minimum_On_Time

This property, of type Unsigned32, represents the minimum number of seconds that the Present_Value shall remain in the ACTIVE state after a write to the Present_Value property causes that property to assume the ACTIVE state.

The mechanism by which this is accomplished is described in Clause 19.2.3.

12.7.21 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.7.22 Relinquish_Default

This property, of type BACnetBinaryPV, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.7.23 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.7.24 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.7.25 Feedback_Value

This property is an indication of the actual value of the entity controlled by Present_Value. The manner by which the Feedback_Value is determined shall be a local matter.

If the object supports event reporting, then this property is the pFeedback parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.7.26 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.7.27 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.7.28 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.7.29 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.7.30 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.7.31 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.7.32 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.7.33 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.7.34 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.7.35 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.7.36 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.7.37 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.7.38 Interface_Value

This read-only property, of type BACnetOptionalBinaryPV, reflects the logical state of the physical output. If the BACnet device is not capable of knowing the logical state of the physical output, then the value of this property shall be NULL. Otherwise, the logical state of this property shall be either INACTIVE or ACTIVE. The relationship between this property and the physical state of the output is determined by the Polarity property. The possible states are summarized in Table 12-9.

12.7.39 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.7.40 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.7.41 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.7.42 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.7.43 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.7.44 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.7.45 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.7.46 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.7.47 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.7.48 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.7.49 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8 Binary Value Object Type

The Binary Value object type defines a standardized object whose properties represent the externally visible characteristics of a binary value. A "binary value" is a control system parameter residing in the memory of the BACnet device. This parameter may assume only one of two distinct states. In this description, those states are referred to as ACTIVE and INACTIVE.

Binary Value objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

The Binary Value object and its properties are summarized in Table 12-10 and described in detail in this clause.

Table 12-10. Properties of the Binary Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetBinaryPV	R[1]
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Inactive_Text	CharacterString	O[2]
Active_Text	CharacterString	O[2]
Change_Of_State_Time	BACnetDateTime	O[3]
Change_Of_State_Count	Unsigned	O[3]
Time_Of_State_Count_Reset	BACnetDateTime	O[3]
Elapsed_Active_Time	Unsigned32	O[4]
Time_Of_Active_Time_Reset	BACnetDateTime	O[4]
Minimum_Off_Time	Unsigned32	O
Minimum_On_Time	Unsigned32	O
Priority_Array	BACnetPriorityArray	O[5]
Relinquish_Default	BACnetBinaryPV	O[5]
Time_Delay	Unsigned	O[6,8]
Notification_Class	Unsigned	O[6,8]
Alarm_Value	BACnetBinaryPV	O[6,8]
Event_Enable	BACnetEventTransitionBits	O[6,8]
Acked_Transitions	BACnetEventTransitionBits	O[6,8]
Notify_Type	BACnetNotifyType	O[6,8]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[6,8]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[8]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[8]
Event_Detection_Enable	BOOLEAN	O[6,8]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[8]
Event_Algorithm_Inhibit	BOOLEAN	O[8,9]
Time_Delay_Normal	Unsigned	O[8]
Reliability_Evaluation_Inhibit	BOOLEAN	O[10]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	O[5]
Value_Source	BACnetValueSource	O[11,13,15]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[12,14]
Last_Command_Time	BACnetTimeStamp	O[12,14]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[14]
Audit_Level	BACnetAuditLevel	O[16]
Auditable_Operations	BACnetAuditOperationFlags	O[16]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[16]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] If one of the optional properties Inactive_Text or Active_Text is present, then both of these properties shall be present.
[3] If one of the optional properties Change_Of_State_Time, Change_Of_State_Count, or Time_Of_State_Count_Reset is present, then all of these properties shall be present.
[4] If one of the optional properties Elapsed_Active_Time or Time_Of_Active_Time_Reset is present, then both of these properties shall be present.
[5] These properties are required if, and shall be present only if, Present_Value is commandable.
[6] These properties are required if the object supports intrinsic reporting.
[7] Footnote removed.
[8] These properties shall be present only if the object supports intrinsic reporting.
[9] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[10] If this property is present, then the Reliability property shall be present.
[11] This property is required if the object supports the value source mechanism.
[12] These properties are required if the object supports the value source mechanism and is commandable.
[13] This property shall be present only if the object supports the value source mechanism.
[14] These properties shall be present only if the object supports the value source mechanism and is commandable.
[15] This property shall be writable as described in Clause 19.5.
[16] This property shall be present only if the device supports audit reporting.

12.8.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.8.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.8.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be BINARY_VALUE.

12.8.4 Present_Value

This property, of type BACnetBinaryPV, reflects the logical state of the Binary Value. The logical state shall be either INACTIVE or ACTIVE. Present_Value shall be optionally commandable. If Present_Value is commandable for a given instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.8.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a binary value object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.8.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.8.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet device or operator can determine.

12.8.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.8.10 Inactive_Text

This property, of type CharacterString, characterizes the intended effect of the INACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the INACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8.11 Active_Text

This property, of type CharacterString, characterizes the intended effect of the ACTIVE state of the Present_Value property. The content of this string is a local matter, but it is intended to represent a human-readable description of the ACTIVE state. If either the Active_Text or the Inactive_Text is present, then both properties shall be present.

12.8.12 Change_Of_State_Time

This property, of type BACnetDateTime, represents the date and time at which the most recent change of state occurred. A "change of state" shall be defined as any event that alters the Present_Value property.

12.8.13 Change_Of_State_Count

This property, of type Unsigned, represents the number of times that the Present_Value property has changed state since the date and time indicated by the Time_Of_State_Count_Reset property.

When this property is set to zero, the Time_Of_State_Count_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_State_Count_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

The Change_Of_State_Count property shall have a range of 0-65535 or greater.

12.8.14 Time_Of_State_Count_Reset

This property, of type BACnetDateTime, indicates the date and time at which the counting of state changes, indicated by the Change_Of_State_Count property, started.

If the Change_Of_State_Count property accepts writes of non-zero values, then the Time_Of_State_Count_Reset property shall be writable, otherwise it shall be read-only.

12.8.15 Elapsed_Active_Time

This property, of type Unsigned32, represents the accumulated number of seconds that the Present_Value property has had the value ACTIVE since the date and time indicated by the Time_Of_Active_Time_Reset property.

When this property is set to zero, the Time_Of_Active_Time_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_Active_Time_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

12.8.16 Time_Of_Active_Time_Reset

This property, of type BACnetDateTime, indicates the date and time at which the accumulation of active time, indicated by the Elapsed_Active_Time property, has been started.

If the Elapsed_Active_Time property accepts writes of non-zero values, then the Time_Of_Active_Time_Reset property shall be writable, otherwise it shall be read-only.

12.8.17 Minimum_Off_Time

This property, of type Unsigned32, represents the minimum number of seconds that the Present_Value shall remain in the INACTIVE state after a write to the Present_Value property causes that property to assume the INACTIVE state.

If the Present_Value is commandable according to Clause 19, then the mechanism by which this is accomplished is described in Clause 19.2.3. Otherwise, the mechanism is a local matter.

12.8.18 Minimum_On_Time

This property, of type Unsigned32, represents the minimum number of seconds that the Present_Value shall remain in the ACTIVE state after a write to the Present_Value property causes that property to assume the ACTIVE state.

If the Present_Value is commandable according to Clause 19, then the mechanism by which this is accomplished is described in Clause 19.2.3. Otherwise, the mechanism is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8.19 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.8.20 Relinquish_Default

This property, of type BACnetBinaryPV, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.8.21 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.8.22 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.8.23 Alarm_Value

This property is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.8.24 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.8.25 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.8.26 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.8.27 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.8.28 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.8.29 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8.30 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.8.31 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.8.32 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.8.33 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.8.34 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.8.35 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.8.36 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.8.37 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.8.38 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.8.39 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.8.40 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.8.41 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.8.42 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.8.43 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.8.44 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.8.45 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.8.46 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.9 Calendar Object Type

The Calendar object type defines a standardized object used to describe a list of calendar dates, which might be thought of as "holidays," "special events," or simply as a list of dates. The object and its properties are summarized in Table 12-11 and described in detail in this clause.

Table 12-11. Properties of the Calendar Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BOOLEAN	R
Date_List	BACnetLIST of BACnetCalendarEntry	R
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property shall be present only if the device supports audit reporting.

12.9.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.9.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.9.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be CALENDAR.

12.9.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.9.5 Present_Value

This property, of type BOOLEAN, indicates the current value of the calendar: TRUE if the current date is in the Date_List and FALSE if it is not.

12.9.6 Date_List

This property, of type BACnetLIST of BACnetCalendarEntry, is a list where each entry is either a specific date or date pattern (Date), range of dates (BACnetDateRange), or month/week-of-month/day-of-week specification (BACnetWeekNDay). If the current date matches the calendar entry criteria, the present value of the Calendar object is TRUE.

For the calendar entry criteria to match, each of the octets in a date pattern or BACnetWeekNDay are evaluated independently and all specified criteria must match.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

As an example, if the calendar entry is a BACnetWeekNDay with an unspecified octet in the month and week-of-month fields but with a specific day-of-week, it means that the Calendar object is TRUE on that day-of-week all year long.

If a BACnet device permits writing to the Date_List property, all choices and all allowed forms of values in the BACnetCalendarEntry shall be supported.

12.9.7 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.9.8 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.9.9 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.9.10 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.9.11 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.9.12 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.10 Command Object Type

The Command object type defines a standardized object whose properties represent the externally visible characteristics of a multi-action command procedure. A Command object is used to write a set of values to a group of object properties, based on the "action code" that is written to the Present_Value of the Command object. Whenever the Present_Value property of the Command object is written to, it triggers the Command object to take a set of actions that change the values of a set of other objects' properties.

The Command object would typically be used to represent a complex context involving multiple variables. The Command object is particularly useful for representing contexts that have multiple states. For example, a particular zone of a building might have three states: UNOCCUPIED, WARMUP, and OCCUPIED. To establish the operating context for each state, numerous objects' properties may need to be changed to a collection of known values. For example, when unoccupied, the temperature setpoint might be 65°F and the lights might be off. When occupied, the setpoint might be 72°F and the lights turned on, etc.

The Command object defines the relationship between a given state and those values that shall be written to a collection of different objects' properties to realize that state. Normally, a Command object is passive. Its In_Process property is FALSE, indicating that the Command object is waiting for its Present_Value property to be written with a value. When Present_Value is written, the Command object shall begin a sequence of actions. The In_Process property shall be set to TRUE, indicating that the Command object has begun processing one of a set of action sequences that is selected based on the particular value written to the Present_Value property. If an attempt is made to write to the Present_Value property through WriteProperty services while In_Process is TRUE, then a Result(-) shall be returned with 'Error Class' = OBJECT and 'Error Code' = BUSY, rejecting the write.

The new value of the Present_Value property determines which sequence of actions the Command object shall take. These actions are specified in an array of action lists indexed by this value. The Action property contains these lists. A given list may be empty, in which case no action takes place, except that In_Process is returned to FALSE and All_Writes_Successful is set to TRUE. If the list is not empty, then for each action in the list the Command object shall write a particular value to a particular property of a particular object in a particular BACnet device. Note, however, that the capability to write to remote devices is not required.

Note also that the Command object does not guarantee that every write will be successful, and no attempt is made by the Command object to "roll back" successfully written properties to their previous values in the event that one or more writes fail. If any of the writes fail, then the All_Writes_Successful property is set to FALSE and the Write_Successful flag for that BACnetActionCommand is set to FALSE. If the Quit_On_Failure flag is TRUE for the failed BACnetActionCommand, then all subsequent BACnetActionCommands in the list shall have their Write_Successful flag set to FALSE. If an individual write succeeds, then the Write_Successful flag for that BACnetActionCommand shall be set to TRUE. If all the writes are successful, then the All_Writes_Successful property is set to TRUE. Once all the writes have been processed to completion by the Command object, the In_Process property is set back to FALSE and the Command object becomes passive again, waiting for another command.

It is important to note that the particular value that is written to the Present_Value property is not what triggers the action, but the act of writing itself. Thus if the Present_Value property has the value 5 and it is again written with the value 5, then the 5th list of actions will be performed again. Writing zero to the Present_Value causes no action to be taken and is the same as invoking an empty list of actions.

The Command object is a powerful concept with many beneficial applications. However, there are unique aspects of the Command object that can cause confusing or destructive side effects if the Command object is improperly configured. Since the Command object can manipulate other objects' properties, it is possible that a Command object could be configured to command itself. In such a case, the In_Process property acts as an interlock and protects the Command object from self-oscillation. However, it is also possible for a Command object to command another Command object that commands the first Command object and so on. The possibility exists for Command objects that command GROUP objects. In these cases of "circular referencing," it is possible for confusing side effects to occur. When references occur to objects in other BACnet devices, there is an increased possibility of time delays, which could cause oscillatory behavior between Command objects that are improperly configured in such a circular manner. Caution should be exercised when configuring Command objects that reference objects outside the BACnet device that contains them.

Command objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Command objects that support intrinsic reporting shall apply the NONE event algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The Command object and its properties are summarized in Table 12-12 and described in detail in this clause.

Table 12-12. Properties of the Command Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Unsigned	W
In_Process	BOOLEAN	R
All_Writes_Successful	BOOLEAN	R
Action	BACnetARRAY[N] of BACnetActionList	R
Action_Text	BACnetARRAY[N] of CharacterString	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Status_Flags	BACnetStatusFlags	O[1]
Event_State	BACnetEventState	O[1]
Reliability	BACnetReliability	O[1]
Event_Detection_Enable	BOOLEAN	O[1,2]
Notification_Class	Unsigned	O[1,2]
Event_Enable	BACnetEventTransitionBits	O[1,2]
Acked_Transitions	BACnetEventTransitionBits	O[1,2]
Notify_Type	BACnetNotifyType	O[1,2]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,2]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[2]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Value_Source	BACnetValueSource	O[4,5,6]
Audit_Level	BACnetAuditLevel	O[7]
Auditable_Operations	BACnetAuditOperationFlags	O[7]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.
[2] These properties shall be present only if the object supports intrinsic reporting.
[3] If this property is present, then the Reliability property shall be present.
[4] This property is required if the object supports the value source mechanism.
[5] This property shall be present only if the object supports the value source mechanism.
[6] This property shall be writable as described in Clause 19.5.
[7] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.10.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.10.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.10.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be COMMAND.

© ISO 2022 – All rights reserved

12.10.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.10.5 Present_Value

This property, of type Unsigned, indicates which action the Command object is to take or has already taken. Whenever the Present_Value property is written to, it triggers the Command object to take a set of actions that change the values of a set of other objects' properties.

The Present_Value may be written to with any value from 0 to the maximum number of actions supported by the Action property. When the Present_Value is written to, the Command object begins a sequence of actions. The new value of the Present_Value property determines which list of actions the Command object shall take. These actions are specified in the Action property, which is an array of lists of actions to take. The array is indexed by the value being written. A given list may be empty, in which case no action takes place. If the list is not empty, then for each action in the list, the Command object shall write a particular value to a particular property of a particular object in a particular BACnet device.

12.10.6 In_Process

This property, of type BOOLEAN, shall be set to TRUE when a value is written to the Present_Value property. This TRUE value indicates that the Command object has begun processing one of a set of action sequences. Once all of the writes have been attempted by the Command object, the In_Process property shall be set back to FALSE.

12.10.7 All_Writes_Successful

This property, of type BOOLEAN, indicates the success or failure of the sequence of actions that are triggered when the Present_Value property is written to. At that time, In_Process is set to TRUE and All_Writes_Successful is set to FALSE. If after the list has been executed, all of the writes have succeeded, then All_Writes_Successful is set to TRUE at the same time that In_Process is set to FALSE. Therefore, while In_Process is TRUE, the value of All_Writes_Successful is not a valid indication of the current or previous operation.

12.10.8 Action

This property, of type BACnetARRAY of BACnetActionList, specifies an array of "action lists." These action lists are indexed by the value that is written to the Present_Value property. A given list may be empty, in which case no action takes place, except that In_Process is returned to FALSE and All_Writes_Successful is set to TRUE. If the list is not empty, then for each action in the list, the Command object shall write a particular value to a particular property of a particular object in a particular BACnet device based on the specifications in each BACnetActionCommand. Each write shall occur in the order that BACnetActionCommand list elements would appear if the list was read using the ReadProperty service. The value zero is a special case that takes no action and behaves like an empty list. The number of defined action lists may be found by reading the Action property with an array index of zero. If the size of this array is changed, the size of the Action_Text array, if present, shall also be changed to the same size.

Each BACnetActionCommand is a specification of a single value to be written to a single property of a single object. BACnetActionCommands have nine parts: an optional BACnet device identifier, an object identifier, a property identifier, a conditional property array index, a value to be written, a conditional priority, an optional post-writing delay time, a premature quit flag, and a write success flag. The components and their datatypes are shown below.

Component	Datatype
Device_Identifier	BACnetObjectIdentifier (Optional)
Object_Identifier	BACnetObjectIdentifier
Property_Identifier	BACnetPropertyIdentifier
Property_Array_Index	Unsigned (Conditional)
Property_Value	Any
Priority	Unsigned (1..16) (Conditional)
Post_Delay	Unsigned (Optional)
Quit_On_Failure	BOOLEAN
Write_Successful	BOOLEAN

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the Device_Identifier is not present, then the write shall be performed on objects residing in the device that contains the Command object. A device that supports the Command object type is not required to support writing outside the device. If the Property_Identifier refers to an array property, then the Property_Array_Index shall also be present to specify the index within the array of the property to be written. If the property being written is a commandable property, then a priority value shall be supplied; otherwise it shall be omitted. If the Quit_On_Failure flag is TRUE, then if the write fails for any reason, the device shall terminate the execution of the action list prematurely. Otherwise, writing shall continue after each failure with the next element of the action list. In either case, All_Writes_Successful shall remain FALSE throughout the execution of the list. After each write, whether successful or not, if the Post_Delay is present, it shall represent a delay in seconds prior to the execution of the next write or the completion of all writing and the setting of In_Process to FALSE.

If the write fails for any reason, then the Write_Successful flag shall be set to FALSE. If the Quit_On_Failure flag is TRUE, then the first write that fails shall also terminate the execution list prematurely. In this case, the Write_Successful flag in subsequent entries in the same list shall be set to FALSE. If the write succeeds, then the Write_Successful flag shall be set to TRUE.

12.10.9 Action_Text

This property, of type BACnetARRAY of CharacterString, shall be used to indicate a text string description for each of the possible values of the Present_Value property. The content of these strings is not restricted. If the size of this array is changed, the size of the Action array shall also be changed to the same size.

12.10.10 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.10.11 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Command object. One flag is associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

12.10.12 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.10.13 Reliability

This property, of type BACnetReliability, provides an indication of whether the Command object is properly configured and is able to process write requests received. Failing property write attempts initiated by this object shall not cause this property to indicate a fault condition. Such error conditions shall be indicated in respective other properties.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.10.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.10.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.10.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.10.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.10.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.10.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.10.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.10.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.10.22 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.10.23 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.10.24 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.10.25 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.10.26 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.10.27 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.10.28 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.11 Device Object Type

The Device object type defines a standardized object whose properties represent the externally visible characteristics of a BACnet device. There shall be exactly one Device object in each BACnet device. A Device object is referenced by its Object_Identifier property, which is not only unique to the BACnet device that maintains this object but is also unique throughout the BACnet internetwork.

Device objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Device objects that support intrinsic reporting shall apply the NONE event algorithm.

The Device object type and its properties are summarized in Table 12-13 and described in detail in this clause.

Table 12-13. Properties of the Device Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
System_Status	BACnetDeviceStatus	R
Vendor_Name	CharacterString	R
Vendor_Identifier	Unsigned16	R
Model_Name	CharacterString	R
Firmware_Revision	CharacterString	R
Application_Software_Version	CharacterString	R
Location	CharacterString	O
Description	CharacterString	O
Protocol_Version	Unsigned	R
Protocol_Revision	Unsigned	R
Protocol_Services_Supported	BACnetServicesSupported	R
Protocol_Object_Types_Supported	BACnetObjectTypesSupported	R
Object_List	BACnetARRAY[N] of BACnetObjectIdentifier	R
Structured_Object_List	BACnetARRAY[N] of BACnetObjectIdentifier	O
Max_APDU_Length_Accepted	Unsigned	R
Segmentation_Supported	BACnetSegmentation	R
Max_Segments_Accepted	Unsigned	O[1]
VT_Classes_Supported	BACnetLIST of BACnetVTClass	O[2]
Active_VT_Sessions	BACnetLIST of BACnetVTSession	O[2]
Local_Time	Time	O[3,4,12]
Local_Date	Date	O[3,4,12]
UTC_Offset	INTEGER	O[4]
Daylight_Savings_Status	BOOLEAN	O[4]
APDU_Segment_Timeout	Unsigned	O[1]
APDU_Timeout	Unsigned	R
Number_Of_APDU_Retries	Unsigned	R
Time_Synchronization_Recipients	BACnetLIST of BACnetRecipient	O[5]
Max_Master	Unsigned(0..127)	O[6]
Max_Info_Frames	Unsigned	O[6]
Device_Address_Binding	BACnetLIST of BACnetAddressBinding	R
Database_Revision	Unsigned	R
Configuration_Files	BACnetARRAY[N] of BACnetObjectIdentifier	O[7]
Last_Restore_Time	BACnetTimeStamp	O[7]
Backup_Failure_Timeout	Unsigned16	O[8]
Backup_Preparation_Time	Unsigned16	O[13]
Restore_Preparation_Time	Unsigned16	O[13]
Restore_Completion_Time	Unsigned16	O[13]
Backup_And_Restore_State	BACnetBackupState	O[7]
Active_COV_Subscriptions	BACnetLIST of BACnetCOVSubscription	O[9]
Last_Restart_Reason	BACnetRestartReason	O[10]
Time_Of_Device_Restart	BACnetTimeStamp	O[10]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Restart_Notification_Recipients	BACnetLIST of BACnetRecipient	O[14]
UTC_Time_Synchronization_Recipients	BACnetLIST of BACnetRecipient	O[5]
Time_Synchronization_Interval	Unsigned	O[5]
Align_Intervals	BOOLEAN	O[5]
Interval_Offset	Unsigned	O[5]
Serial_Number	CharacterString	O[11]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Status_Flags	BACnetStatusFlags	O[15]
Event_State	BACnetEventState	O[15]
Reliability	BACnetReliability	O[15]
Event_Detection_Enable	BOOLEAN	O[15,16]
Notification_Class	Unsigned	O[15,16]
Event_Enable	BACnetEventTransitionBits	O[15,16]
Acked_Transitions	BACnetEventTransitionBits	O[15,16]
Notify_Type	BACnetNotifyType	O[15,16]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[15,16]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[16]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[16]
Reliability_Evaluation_Inhibit	BOOLEAN	O[17]
Active_COV_Multiple_Subscriptions	BACnetLIST of BACnetCOVMultipleSubscription	O[18]
Audit_Notification_Recipient	BACnetRecipient	O[19,20]
Audit_Level	BACnetAuditLevel	O[20]
Auditable_Operations	BACnetAuditOperationFlags	O[20]
Device_UUID	OCTET STRING (Size(16))	O[21]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Deployed_Profile_Location	CharacterString	O[19]
Profile_Name	CharacterString	O

[1] These properties are required if, and shall be present only if, segmentation of any kind is supported.
[2] These properties are required if, and shall be present only if, the VT Services are supported.
[3] If the device supports the execution of the TimeSynchronization service, then these properties shall be present.
[4] If the device supports the execution of the UTCTimeSynchronization service, then these properties shall be present.
[5] These properties shall be present only if all of them are present. If present, these properties shall be writable.
[6] These properties are required if the device is an MS/TP master node.
[7] These properties are required if, and shall be present only if, the device supports execution of the backup and restore procedures.
[8] This property is required if, and shall be present only if, the device supports the backup and restore procedures. If present, this property shall be writable.
[9] This property is required if, and shall be present only if, the device supports execution of either the SubscribeCOV or SubscribeCOVProperty service.
[10] These properties are required if the device supports the restart procedure as described in Clause 19.3.
[11] If the device supports the execution of the You-Are service, then this property shall be present.
[12] These properties shall be present if the device is capable of tracking date and time.
[13] These properties are required if, and shall be present only if, the device supports execution of the backup and restore procedures as described in Clause 19.1 and cannot respond to subsequent communications within the minimum value it will accept in its APDU_Timeout property.
[14] This property is required if, and shall be present only if, the device supports execution of the restart procedure as described in Clause 19.3.
[15] These properties are required if the object supports intrinsic reporting.
[16] These properties shall be present only if the object supports intrinsic reporting.
[17] If this property is present, then the Reliability property shall be present.
[18] This property is required if, and shall be present only if, the device supports execution of the SubscribeCOVPropertyMultiple service.
[19] This property is required to be writable if present.
[20] This property shall be present if, and only if, the device supports audit reporting.
[21] This property shall be present if the device supports BACnet/SC network ports.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. For the Device object, the object identifier shall be unique internetwork-wide.

12.11.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique internetwork-wide. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.11.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DEVICE.

12.11.4 System_Status

This property, of type BACnetDeviceStatus, reflects the current physical and logical status of the BACnet device. The values that may be taken on by this property are:

{OPERATIONAL, OPERATIONAL_READ_ONLY, DOWNLOAD_REQUIRED, DOWNLOAD_IN_PROGRESS, NON_OPERATIONAL, BACKUP_IN_PROGRESS}

The exact meaning of these states, except for BACKUP_IN_PROGRESS, in a given device and their synchronization with other internal operations of the device or the execution of BACnet services by the device are local matters and are not defined by this standard.

12.11.5 Vendor_Name

This property, of type CharacterString, identifies the manufacturer of the BACnet device.

12.11.6 Vendor_Identifier

This read-only property, of type Unsigned16, is a unique vendor identification code, assigned by ASHRAE, which is used to distinguish proprietary extensions to the protocol. See Clause 23.

12.11.7 Model_Name

This read-only property, of type CharacterString, is assigned by the vendor to represent the model of the BACnet device.

12.11.8 Firmware_Revision

This property, of type CharacterString, is assigned by the vendor to represent the level of firmware installed in the BACnet device.

12.11.9 Application_Software_Version

This property, of type CharacterString, identifies the version of application software installed in the machine. The content of this string is a local matter, but it could be a date-and-time stamp, a programmer's name, a host file version number, etc.

12.11.10 Location

This property, of type CharacterString, indicates the physical location of the BACnet device.

12.11.11 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.11.12 Protocol_Version

This property, of type Unsigned, represents the version of the BACnet protocol supported by this BACnet device. Every major revision of BACnet shall increase this version number by one. The initial release of BACnet shall be version 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.13 Protocol_Revision

This property, of type Unsigned, shall indicate the minor revision level of the BACnet standard. This value shall start at 1 and be incremented for any substantive change(s) to the BACnet standard that affect device communication or behavior. This value shall revert to zero upon each change to the Protocol_Version property. Changes to the values for Protocol_Version and Protocol_Revision are recorded in the History of Revisions at the end of this standard.

This property is required for all devices implementing BACnet Protocol_Version 1, Protocol_Revision 1 and above. Absence of this property shall indicate a device implemented to a version of the standard prior to the definition of the Protocol_Revision property.

12.11.14 Protocol_Services_Supported

This property, of type BACnetServicesSupported, indicates which standardized protocol services are executed by this device's protocol implementation.

12.11.15 Protocol_Object_Types_Supported

This property, of type BACnetObjectTypesSupported, indicates which standardized object types can be present in this device's protocol implementation. The list of properties present in a particular object may be acquired by use of the ReadPropertyMultiple service with a property reference of ALL (see Clause 15.7.3.1.2).

12.11.16 Object_List

This read-only property is a BACnetARRAY of BACnetObjectIdentifier, one Object_Identifier for each object within the device that is accessible through BACnet services.

12.11.17 Structured_Object_List

This property is of type BACnetARRAY of BACnetObjectIdentifier. Entries in the array reference objects chosen for use as starting points for the traversal of object hierarchies. The objects directly referenced by this property shall be restricted to Structured View and Life Safety Zone objects.

12.11.18 Max_APDU_Length_Accepted

This property, of type Unsigned, is the maximum number of octets that may be contained in a single, indivisible application layer protocol data unit. The value of this property shall be greater than or equal to 50. The value of this property is also constrained by the underlying data link technology and shall be less than or equal to the largest APDU_Length of the enabled Network Port objects used to represent the underlying data links. See Clauses 6 through 11, Annex J, Annex O, and Annex U.

If the value of this property is not encodable in the 'Max APDU Length Accepted' parameter of a ConfirmedRequest-PDU, then the value encoded shall be the highest encodable value less than the value of this property. In such cases, a responding device may ignore the encoded value in favor of the value of this property, if it is known.

12.11.19 Segmentation_Supported

This property, of type BACnetSegmentation, indicates whether the BACnet device supports segmentation of messages and, if so, whether it supports segmented transmission, reception, or both:

{SEGMENTED_BOTH, SEGMENTED_TRANSMIT, SEGMENTED_RECEIVE, NO_SEGMENTATION}

12.11.20 Max_Segments_Accepted

The Max_Segments_Accepted property, of type Unsigned, shall indicate the maximum number of segments of an APDU that this device will accept.

If the Segmentation_Supported property of the Device object has a value of SEGMENTED_TRANSMIT or NO_SEGMENTATION, this property shall have a value of 1. If the Segmentation_Supported property of the Device object has a value of SEGMENTED_BOTH or SEGMENTED_RECEIVE, then the value of this property shall have a value greater than 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.21 VT_Classes_Supported

The VT_Classes_Supported property is a BACnetLIST of BACnetVTClass each of which is an enumeration indicating a particular set of terminal characteristics. A given BACnet device may support multiple types of behaviors for differing types of terminals or differing types of operator interface programs. At a minimum, such devices shall support the "Default-terminal" VT-class defined in Clause 17.5.

12.11.22 Active_VT_Sessions

The Active_VT_Sessions property is a BACnetLIST of BACnetVTSession each of which consists of a Local VT Session Identifier, a Remote VT Session Identifier, and Remote VT Address. This property provides a network-visible indication of those virtual terminal sessions (VT-Sessions) that are active at any given time. Whenever a virtual terminal session is created with the VT-Open service, a new entry is added to the Active_VT_Sessions list. Similarly, whenever a VT-session is terminated, the corresponding entry shall be removed from the Active_VT_Sessions list.

12.11.23 Local_Time

The Local_Time property, of type Time, shall indicate the time of day to the best of the device's knowledge. If the BACnet device does not have any knowledge of time or date, then the Local_Time property shall be omitted. This property shall be present if the BACnet device is capable of tracking date and time. If the device restarts, and there is no local time available yet, then the clock shall be initialized to the time 00:00:00.00 and the device shall commence tracking the time.

12.11.24 Local_Date

The Local_Date property, of type Date, shall indicate the date to the best of the device's knowledge. If the BACnet device does not have any knowledge of time or date, then the Local_Date property shall be omitted. This property shall be present if the BACnet device is capable of tracking date and time. If the device restarts, and there is no local date available yet, then the clock shall be initialized to a date on or before January 1, 1990 and the device shall commence tracking the date.

12.11.25 UTC_Offset

The UTC_Offset property, of type INTEGER, shall indicate the number of minutes (-1440 to +1440) offset between local standard time and Universal Time Coordinated. The time zones to the west of the zero degree meridian shall be positive values, and those to the east shall be negative values. The value of the UTC_Offset property is subtracted from the UTC received in UTCTimeSynchronization service requests to calculate the correct local standard time. UTC_Offset shall be configurable and accept any multiple of 15 minutes across the full range.

12.11.26 Daylight_Savings_Status

The Daylight_Savings_Status property, of type BOOLEAN, shall indicate whether daylight savings time is in effect (TRUE) or not (FALSE) at the BACnet device's location.

12.11.27 APDU_Segment_Timeout

The APDU_Segment_Timeout property, of type Unsigned, shall indicate the amount of time in milliseconds between retransmission of an APDU segment. A suggested default value for this property is 5000 milliseconds. This value shall be non-zero if the Device object property called Number_Of_APDU_Retries is non-zero. See Clause 5.3.

In order to achieve reliable communication, it is recommended that the values of the APDU_Segment_Timeout properties of the Device objects of all intercommunicating devices should contain the same value.

12.11.28 APDU_Timeout

The APDU_Timeout property, of type Unsigned, shall indicate the amount of time in milliseconds between retransmissions of an APDU requiring acknowledgment for which no acknowledgment has been received. A suggested default value for this property is 6000 milliseconds for devices that permit modification of this parameter. Otherwise, the default value shall be 10000 milliseconds. This value shall be non-zero if the Device object property called Number_Of_APDU_Retries is non-zero. See Clause 5.3.

In order to achieve reliable communication, it is recommended that the values of the APDU_Timeout properties of the Device objects of all intercommunicating devices should contain the same value.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.29 Number_Of_APDU_Retries

The Number_Of_APDU_Retries property, of type Unsigned, shall indicate the maximum number of times that an APDU shall be retransmitted. A suggested default value for this property is 3. If this device does not perform retries, then this property shall be set to zero. If the value of this property is greater than zero, a non-zero value shall be placed in the Device object APDU_Timeout property. See Clause 5.3.

12.11.30 Deleted Clause

This clause has been removed.

12.11.31 Time_Synchronization_Recipients

This property, of type BACnetLIST of BACnetRecipient, is used to control the restrictions placed on a device's use of the TimeSynchronization service. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, or the property is not present, the device is prohibited from automatically sending a TimeSynchronization request. If the list is of length one or more, the device may automatically send a TimeSynchronization request but only to the devices or addresses listed. If this property is present, it shall be writable.

12.11.32 Max_Master

This property, of type Unsigned, shall be present if the device is a master node on an MS/TP network. The value of this property shall reflect the value of the Max_Master property of the Network Port object with the lowest object instance whose Network_Type is MSTP and whose Out_Of_Service is FALSE. See Clause 12.56.51.

The value of this property is a local matter if there are no MS/TP Network Port objects with Out_Of_Service set to FALSE.

If this property is writable, writing to this property shall cause the new value to take effect immediately, bypassing the activation functionality of the Network Port Object. See Clauses 12.56.11 and 12.56.12.

12.11.33 Max_Info_Frames

This property, of type Unsigned, shall be present if the device is a master node on an MS/TP network. The value of this property shall reflect the value of the Max_Info_Frames property of the Network Port object with the lowest object instance whose Network_Type is MSTP and whose Out_Of_Service is FALSE. See Clause 12.56.52.

The value of this property is a local matter if there are no MS/TP Network Port objects with Out_Of_Service set to FALSE.

If this property is writable, writing to this property shall cause the new value to take effect immediately, bypassing the activation functionality of the Network Port Object. See Clauses 12.56.11 and 12.56.12.

12.11.34 Device_Address_Binding

The Device_Address_Binding property is a BACnetLIST of BACnetAddressBinding each of which consists of a BACnet Object_Identifier of a Device object and a BACnet device address in the form of a BACnetAddress. Entries in the list identify the actual device addresses that will be used when the remote device must be accessed via a BACnet service request. A value of zero may be used for the network-number portion of BACnetAddress entries for other devices residing on the same network as this device. The list may be empty if no device identifier-device address bindings are currently known to the device.

12.11.35 Database_Revision

This property, of type Unsigned, is a logical revision number for the device's database. It is incremented when an object is created, an object is deleted, an object's name is changed, an object's Object_Identifier property is changed, or a restore is performed with the exception that the creation and deletion of temporary configuration files during a backup or restore procedure shall not affect this property.

12.11.36 Configuration_Files

This property is a BACnetARRAY of BACnetObjectIdentifier. Entries in the array identify the files within the device that define the device's image that can be backed up. The content of this property is only required to be valid during the backup procedure. This property must be supported if the device supports the BACnet backup and restore procedure as described in Clause 19.1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.37 Last_Restore_Time

This property, of type BACnetTimeStamp, is the time at which the device's image was last restored as described in Clause 19.1.

12.11.38 Backup_Failure_Timeout

This property, of type Unsigned16, is the time, in seconds, that the device being backed up or restored must wait before unilaterally ending the backup or restore procedure. This property must be writable with the intent that the device performing the backup, or the human operator, will configure this with an appropriate timeout.

12.11.39 Active_COV_Subscriptions

The Active_COV_Subscriptions property is a BACnetLIST of BACnetCOVSubscription, each of which consists of a Recipient, a Monitored Property Reference, an Issue Confirmed Notifications flag, a Time Remaining value and an optional COV Increment. This property provides a network-visible indication of those COV subscriptions that are active at any given time. Whenever a COV Subscription is created with the SubscribeCOV or SubscribeCOVProperty service, a new entry is added to the Active_COV_Subscriptions list. Similarly, whenever a COV Subscription is terminated, the corresponding entry shall be removed from the Active_COV_Subscriptions list. If the subscribed-to property represents a numeric quantity, the COV Increment in use for the COV subscription shall be included in the Active_COV_Subscriptions entry otherwise it is a local matter whether the COV Increment is included in the Active_COV_Subscriptions entry.

12.11.40 Last_Restart_Reason

This property, of type BACnetRestartReason, indicates the reason for the last device restart. See Clause 19.3 for a description of the restart procedure. The possible values for this property are:

UNKNOWN	The device cannot determine the cause of the last reset.
COLDSTART	A ReinitializeDevice request was received with a 'Reinitialized State of Device' of COLDSTART or the device was made to COLDSTART by some other means.
WARMSTART	A ReinitializeDevice request was received with a 'Reinitialized State of Device' of WARMSTART or the device was made to WARMSTART by some other means.
DETECTED_POWER_LOST	The device detected that incoming power was lost.
DETECTED_POWERED_OFF	The device detected that its power switch was turned off.
HARDWARE_WATCHDOG	The hardware watchdog timer reset the device.
SOFTWARE_WATCHDOG	The software watchdog timer reset the device.
SUSPENDED	The device was suspended. How the device was suspended or what it means to be suspended is a local matter.
ACTIVATE_CHANGES	A ReinitializeDevice request was received with a 'Reinitialized State of Device' of ACTIVATE_CHANGES which caused the device to restart.

12.11.41 Time_Of_Device_Restart

This property, of type BACnetTimeStamp, is the time at which the device was last restarted. See Clause 19.3 for a description of the restart procedure.

12.11.42 Restart_Notification_Recipients

This property, of type BACnetLIST of BACnetRecipient, is used to control the restrictions on which devices, if any, are to be notified when a restart occurs. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, a device is prohibited from sending a device restart notification. The default value of the property shall be a single entry representing a broadcast on the local network. If the property is not writable, then it shall contain the default value. If the list is of length one or more, a device shall send a restart notification, but only to the devices or addresses listed. See Clause 19.3 for a description of the restart procedure.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.43 UTC_Time_Synchronization_Recipients

This property, of type BACnetLIST of BACnetRecipient, is used to control the restrictions placed on a device's use of the UTCTimeSynchronization service. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, or the property is not present, the device is prohibited from automatically sending a UTCTimeSynchronization request. If the list is of length one or more, the device may automatically send a UTCTimeSynchronization request but only to the devices or addresses listed. If this property is present, it shall be writable.

12.11.44 Time_Synchronization_Interval

This property, of type Unsigned, specifies the periodic interval in minutes at which TimeSynchronization and UTCTimeSynchronization requests shall be sent. If this property has a value of zero, then periodic time synchronization is disabled. If this property is present, it shall be writable.

12.11.45 Align_Intervals

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic time synchronization is enabled. If periodic time synchronization is enabled and the time synchronization interval is a factor of (divides without remainder) an hour or day, then the beginning of the period specified for time synchronization shall be aligned to the hour or day, respectively. If this property is present, it shall be writable.

12.11.46 Interval_Offset

This property, of type Unsigned, specifies the offset in minutes from the beginning of the period specified for time synchronization until the actual time synchronization requests are sent. The offset used shall be the value of Interval_Offset modulo the value of Time_Synchronization_Interval; e.g., if Interval_Offset has the value 31 and Time_Synchronization_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals is FALSE. If this property is present, it shall be writable.

12.11.47 Backup_Preparation_Time

This property, of type Unsigned16, indicates the amount of time in seconds that the device might remain unresponsive after the sending of a ReinitializeDevice-ACK at the start of a backup procedure. The device that initiated the backup shall either wait the period of time specified by this property or be prepared to encounter communication timeouts during this period. The use of this value is described in more detail in Clause 19.1.

This property is required if the device supports execution of the backup and restore procedures and cannot complete backup preparation within the minimum value it will accept in its APDU_Timeout property.

12.11.48 Restore_Preparation_Time

This property, of type Unsigned16, indicates the amount of time in seconds that the device is allowed to remain unresponsive after the sending of a ReinitializeDevice-ACK at the start of a restore procedure. The restoring device shall either wait or be prepared to encounter communication timeouts during this period. The use of this value is described in more detail in Clause 19.1.

12.11.49 Restore_Completion_Time

This property, of type Unsigned16, indicates the amount of time in seconds that the device is allowed to remain unresponsive after the sending of a ReinitializeDevice-ACK at the end of a restore procedure. The restoring device shall either wait or be prepared to encounter communication timeouts during this period. The use of this value is described in more detail in Clause 19.1.

12.11.50 Backup_And_Restore_State

This property, of type BACnetBackupState, indicates a server device's backup and restore state. The use of this value is described in more detail in Clause 19.1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.51 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.11.52 Serial_Number

This read-only property, of type CharacterString, is assigned by the vendor to represent the serial number in a vendor-specific model series. The combination of Model_Name, Vendor_Identifier and Serial_Number uniquely identifies a device.

12.11.53 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Device object. One flag is associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN The value of this flag shall be logical FALSE (0).

OUT_OF_SERVICE The value of this flag shall be logical FALSE (0).

12.11.54 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.11.55 Reliability

This property, of type BACnetReliability, provides an indication of whether the Device object is properly configured, is able to perform the functionality it represents and to maintain properties that indicate status information.

12.11.56 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.11.57 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.11.58 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.11.59 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.11.60 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.11.61 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.11.62 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.11.63 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.11.64 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.11.65 Active_COV_Multiple_Subscriptions

This property, of type BACnetLIST of BACnetCOVMultipleSubscription, provides a network-visible indication of those COV-multiple subscriptions that are active at any given time. Each list entry constitutes a COV-multiple context and consists of a Recipient, an Issue Confirmed Notifications flag, a Time Remaining value, a Maximum Notification Delay timeout, and a list of monitored objects. For each monitored object, a list of COV Specifications is present with each containing a Monitored Property reference, an optional COV Increment, and a Timestamped flag. Only one COV-multiple context shall be present in this property for a given Recipient and form of notification indicated by the Issue Confirmed Notifications flag. A Recipient is identified by the BACnet address of the COV-client and the Subscriber Process Identifier. If the subscribed-to property in a COV specification represents a numeric quantity, the COV Increment is use for the COV subscription shall be included in the Active_COV_Multiple_Subscriptions corresponding COV specification entry otherwise it is a local matter whether the COV Increment is included in the Active_COV_Multiple_Subscriptions entry.

Whenever a COV-multiple context is created with the SubscribeCOVPropertyMultiple service, a new entry is added to the Active_COV_Multiple_Subscriptions list if no entry is present for the Recipient and form of notification. If an entry exists, COV subscription specifications are added or modified in the list of COV subscription specifications of the entry. Similarly, whenever a COV-multiple subscription is terminated, the corresponding COV subscription specifications of

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

the entry shall be removed. If no COV subscription specifications remain in the entry, or the remaining time indicated in the entry reaches zero, the entire entry is removed from the Active_COV_Multiple_Subscriptions list.

12.11.66 Audit_Notification_Recipient

This property, of type BACnetRecipient, specifies the audit logging device to which audit notifications are sent.

When this property is changed, an audit notification shall be generated and it shall be either globally broadcast, or sent to both the original audit notification recipient and the new audit notification recipient.

If this property is present, it shall be writable.

12.11.67 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.11.68 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.11.69 Device_UUID

This read-only property, of type OCTET STRING (Size(16)), represents the Universally Unique ID of the device in binary representation.

This property shall have a valid 16-octet UUID before the device is first deployed in an installation. For the generation of UUIDs and the binary representation in the OCTET STRING see RFC 4122.

12.11.70 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.11.71 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.11.72 Deployed_Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme. The referenced xdd file contains additional information about the deployed device. It is intended to be used as a supplement to the information referenced by the Profile_Location property. If present, this property shall be writable and shall, at a minimum, support storage of strings with an encoded length up to 255 octets.

12.11.73 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12 Event Enrollment Object Type

The Event Enrollment object type defines a standardized object that represents and contains the information required for algorithmic reporting of events. For the general event concepts and algorithmic event reporting, see Clause 13.2.

For the Event Enrollment object, detecting events is accomplished by performing particular event and fault algorithms on monitored values of a referenced object. The parameters for the algorithms are provided by the Event Enrollment object. The standard event algorithms are defined in Clause 13.3. The standard fault algorithms are defined in Clause 13.4. Event Enrollment objects do not modify or otherwise influence the state or operation of the referenced object.

For the reliability indication by the Reliability property of the Event Enrollment object, internal unreliable operation such as configuration error or communication failure takes precedence over reliability indication for the monitored object. Fault indications determined by the fault algorithm, if any, have least precedence.

Clause 13.2 describes the interaction between Event Enrollment objects, the Notification Class objects, and the Alarm and Event application services.

The Event Enrollment object and its properties are summarized in Table 12-14 and described in detail in this clause.

Table 12-14. Properties of the Event Enrollment Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Event_Type	BACnetEventType	R
Notify_Type	BACnetNotifyType	R
Event_Parameters	BACnetEventParameter	R
Object_Property_Reference	BACnetDeviceObjectPropertyReference	R
Event_State	BACnetEventState	R
Event_Enable	BACnetEventTransitionBits	R
Acked_Transitions	BACnetEventTransitionBits	R
Notification_Class	Unsigned	R
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	R
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O
Event_Detection_Enable	BOOLEAN	R
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O
Event_Algorithm_Inhibit	BOOLEAN	O[1]
Time_Delay_Normal	Unsigned	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Fault_Type	BACnetFaultType	O[2]
Fault_Parameters	BACnetFaultParameter	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[3]
Auditable_Operations	BACnetAuditOperationFlags	O[3]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[2] These properties are required if, and shall be present only if, the object supports fault algorithms other than NONE.
[3] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.12.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.12.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be EVENT_ENROLLMENT.

12.12.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.12.5 Event_Type

This read-only property, of type BACnetEventType, indicates the type of event algorithm that is to be used to detect the occurrence of events and the event value notification parameters conveyed in event notifications. The value of this property is an enumerated type that may have any standard BACnetEventType value except CHANGE_OF_RELIABILITY.

There is a specific relationship between each event algorithm, the event algorithm parameters, and the event type The Event_Type is determined by the chosen event parameters in the Event_Parameters property and reflects the event algorithm that is used to determine the event state. The event algorithm for each Event_Type is specified in Clause 13.3. The Event_Parameters property provides the parameters needed by the algorithm.

12.12.6 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.12.7 Event_Parameters

This property property, of type BACnetEventParameter, determines the algorithm used to monitor the referenced object and provides the parameter values needed for this event algorithm. The parameter values specified in this property serve as event algorithm parameters and as reference type parameters reference properties whose values are used as event algorithm parameters, as defined by the respective event algorithm in Clause 13.3. The mapping to the event algorithm parameters is defined in Table 12-15.

If the Event Enrollment object evaluates reliability only and does not apply an event algorithm, then the event algorithm NONE shall be set in this property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Table 12-15. Event Algorithm, Event Parameters and Event Algorithm Parameters

Event Algorithm	Event Parameters	Event Algorithm Parameters
NONE	none	none
ACCESS_EVENT	List_Of_Access_Events Access_Event_Time_Reference	pAccessEvents Referent's value is pAccessEventTime
BUFFER_READY	Notification_Threshold Previous_Notification_Count	pThreshold pPreviousCount
CHANGE_OF_BITSTRING	Time_Delay Bitmask List_Of_Bitstring_Values	pTimeDelay pBitmask pAlarmValues
CHANGE_OF_CHARACTERSTRING	Time_Delay List_Of_Alarm_Values	pTimeDelay pAlarmValues

© ISO 2022 – All rights reserved

Event Algorithm	Event Parameters	Event Algorithm Parameters
CHANGE_OF_DISCRETE_VALUE	Time_Delay	pTimeDelay
CHANGE_OF_LIFE_SAFETY	Time_Delay List_Of_Alarm_Values List_Of_Life_Safety_Alarm_Values Mode_Property_Reference	pTimeDelay pAlarmValues pLifeSafetyAlarmValues Referent's value is pMode
CHANGE_OF_STATE	Time_Delay List_Of_Values	pTimeDelay pAlarmValues
CHANGE_OF_STATUS_FLAGS	Time_Delay Selected_Flags	pTimeDelay pSelectedFlags
CHANGE_OF_TIMER	Time_Delay Alarm_Values Update_Time_Reference	pTimeDelay pAlarmValues Referent's value is pUpdateTime
CHANGE_OF_VALUE	Time_Delay Bitmask Referenced_Property_Increment	pTimeDelay pBitmask pIncrement
COMMAND_FAILURE	Time_Delay Feedback_Property_Reference	pTimeDelay Referent's value is pFeedbackValue
DOUBLE_OUT_OF_RANGE	Time_Delay Low_Limit High_Limit Deadband	pTimeDelay pLowLimit pHighLimit pDeadband
EXTENDED	Vendor_Id Extended_Event_Type Parameters	pVendorId pEventType pParameters
FLOATING_LIMIT	Time_Delay Setpoint_Reference Low_Diff_Limit High_Diff_Limit Deadband	pTimeDelay Referent's value is pSetpoint pLowDiffLimit pHighDiffLimit pDeadband
OUT_OF_RANGE	Time_Delay Low_Limit High_Limit Deadband	pTimeDelay pLowLimit pHighLimit pDeadband
SIGNED_OUT_OF_RANGE	Time_Delay Low_Limit High_Limit Deadband	pTimeDelay pLowLimit pHighLimit pDeadband
UNSIGNED_OUT_OF_RANGE	Time_Delay Low_Limit High_Limit Deadband	pTimeDelay pLowLimit pHighLimit pDeadband
UNSIGNED_RANGE	Time_Delay Low_Limit High_Limit	pTimeDelay pLowLimit pHighLimit

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.12.8 Object_Property_Reference

This property, of type BACnetDeviceObjectPropertyReference, designates the particular object and property referenced by this Event Enrollment object. The event algorithm specified by the Event_Type property is applied to the referenced property in order to determine the Event_State of the event. The value of the referenced property is used as the value of the pMonitoredValue parameter of the event algorithm. The value of the referenced property is also used as the value of the pMonitoredValue parameter of the fault algorithm, if a fault algorithm is applied.

If this property is writable, it may be restricted to only support references to objects inside of the device containing the Event Enrollment object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

© ISO 2022 – All rights reserved

If this property is set to reference an object outside the device containing the Event Enrollment object, the method used for acquisition of the referenced property value for the purpose of monitoring is a local matter. The only restriction on the method of data acquisition is that the monitoring device be capable of using ReadProperty for this purpose so as to be interoperable with all BACnet devices.

Depending on the event algorithm, the values of additional properties of the monitored object are used for particular event algorithm parameters, as specified by Table 12-15.1.

Table 12-15.1. Additional Monitored Object Properties by Event Algorithm

Event Algorithm	Additional Monitored Object Properties	Event Algorithm Parameters
NONE	none	none
ACCESS_EVENT	Status_Flags Access_Event_Tag Access_Event_Credential Access_Event_Authentication_Factor	pStatusFlags pAccessEventTag pAccessCredential pAccessFactor
BUFFER_READY	Log_Buffer	pLogBuffer
CHANGE_OF_BITSTRING	Status_Flags	pStatusFlags
CHANGE_OF_CHARACTERSTRING	Status_Flags	pStatusFlags
CHANGE_OF_DISCRETE_VALUE	Status_Flags	pStatusFlags
CHANGE_OF_LIFE_SAFETY	Status_Flags Operation_Expected	pStatusFlags pOperationExpected
CHANGE_OF_STATE	Status_Flags	pStatusFlags
CHANGE_OF_STATUS_FLAGS	Present_Value	pPresentValue
CHANGE_OF_TIMER	Status_Flags Initial_Timeout Expiration_Time Last_State_Change	pStatusFlags pInitialTimeout pExpirationTime pLastStateChange
CHANGE_OF_VALUE	Status_Flags	pStatusFlags
COMMAND_FAILURE	Status_Flags	pStatusFlags
DOUBLE_OUT_OF_RANGE	Status_Flags	pStatusFlags
EXTENDED	Defined by vendor	Defined by vendor
FLOATING_LIMIT	Status_Flags	pStatusFlags
OUT_OF_RANGE	Status_Flags	pStatusFlags
SIGNED_OUT_OF_RANGE	Status_Flags	pStatusFlags
UNSIGNED_OUT_OF_RANGE	Status_Flags	pStatusFlags
UNSIGNED_RANGE	Status_Flags	pStatusFlags

Depending on the fault algorithm, the values of additional properties of the monitored object are used for particular fault algorithm parameters, as specified by Table 12-15.2.

Table 12-15.2. Additional Monitored Object Properties by Fault Algorithm

Fault Algorithm	Additional MonitoredObject Properties	Fault Algorithm Parameters
NONE	none	none
FAULT_CHARACTERSTRING	none	none
FAULT_EXTENDED	Defined by vendor	Defined by vendor
FAULT_LIFE_SAFETY	none	none
FAULT_LISTED	none	none
FAULT_OUT_OF_RANGE	none	none
FAULT_STATE	none	none
FAULT_STATUS_FLAGS	none	none

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12.9 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.12.10 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.12.11 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.12.12 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.12.13 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.12.14 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.12.15 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.12.16 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) event reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.12.17 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12.18 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.12.19 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.12.20 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an object. Three of the flags are associated with the values of other properties of the object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

This property is not the pStatusFlags parameter for the object's event algorithm. The Status_Flags property of the object referenced by the Object_Property_Reference property shall be the pStatusFlags parameter for the event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.12.21 Reliability

This property, of type BACnetReliability, provides an indication of:

(a) the ability of the Event Enrollment object to perform its monitoring function,
(b) the reliability of the monitored object, and
(c) the result of the fault algorithm, if one is in use.

The reliability-evaluation in an Event Enrollment object is processed in the following order:

(1) Evaluate the reliability of the Event Enrollment object itself.
(2) If the reliability is NO_FAULT_DETECTED, then if the reliability of the monitored object is not NO_FAULT_DETECTED, then the reliability shall be MONITORED_OBJECT_FAULT.
(3) If the reliability is NO_FAULT_DETECTED, then evaluate the fault algorithm, if one is in use, to determine the reliability.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12.22 Fault_Type

This read-only property, of type BACnetFaultType, indicates the type of fault algorithm that is applied by the object.

There is a specific relationship between each fault algorithm, the fault algorithm parameters, and the fault type. The Fault_Type is determined by the chosen fault parameters in the Fault_Parameters property and reflects the fault algorithm that is used to monitor the referenced object for fault conditions. The fault algorithm for each Fault_Type is specified in Clause 13.4. The Fault_Parameters property provides the parameters needed by the algorithm.

12.12.23 Fault_Parameters

This property, of type BACnetFaultParameter, determines the fault algorithm used to monitor the referenced object and provides the parameter values needed for this fault algorithm. The mapping to the fault algorithm parameter values is defined in Table 12-15.3.

If the Event Enrollment object does not apply a fault algorithm, then the fault parameter choice NONE shall be set in this property.

Table 12-15.3. Fault Algorithm, Fault Parameters and Fault Algorithm Parameters

Fault Algorithm	Fault Parameters	Fault Algorithm Parameters
NONE	none	none
FAULT_CHARACTERSTRING	List_Of_Fault_Values	pFaultValues
FAULT_EXTENDED	Vendor_Id Extended_Fault_Type Parameters	pVendorId pFaultType pParameters
FAULT_LIFE_SAFETY	List_Of_Fault_Values Mode_Property_Reference	pFaultValues Referent's value is pMode
FAULT_LISTED	Fault_List_Reference	Referent's value is pMonitoredList
FAULT_OUT_OF_RANGE	Min_Normal_Value Max_Normal_Value	pMinimumNormalValue pMaximumNormalValue
FAULT_STATE	List_Of_Fault_Values	pFaultValues
FAULT_STATUS_FLAGS	Status_Flags_Property_Reference	pMonitoredValue

12.12.24 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.12.25 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.12.26 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.12.27 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.12.28 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.12.29 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.12.30 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.13 File Object Type

The File object type defines a standardized object that is used to describe properties of data files that may be accessed using File Services (see Clause 14). The file object type and its properties are summarized in Table 12-16 and described in detail in this clause.

Table 12-16. Properties of the File Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
File_Type	CharacterString	R
File_Size	Unsigned	R[1]
Modification_Date	BACnetDateTime	R
Archive	BOOLEAN	W
Read_Only	BOOLEAN	R
File_Access_Method	BACnetFileAccessMethod	R
Record_Count	Unsigned	O[2]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[3]
Auditable_Operations	BACnetAuditOperationFlags	O[3]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If the file size can be changed by writing to the file, and File_Access_Method is STREAM_ACCESS, then this property shall be writable.
[2] This property shall be present only if File_Access_Method is RECORD_ACCESS. If the number of records can be changed by writing to the file, then this property shall be writable.
[3] This property shall be present only if the device supports audit reporting.

12.13.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.13.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.13.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be FILE.

12.13.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.13.5 File_Type

This property, of type CharacterString, identifies the intended use of this file.

12.13.6 File_Size

This property, of type Unsigned, indicates the size of the file data in octets. If the size of the file can be changed by writing to the file, and File_Access_Method is STREAM_ACCESS, then this property shall be writable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Writing to the File_Size property with a value less than the current size of the file shall truncate the file at the specified position. Writing a File_Size of 0 shall delete all of the file data but not the File object itself. Writing to the File_Size property with a value greater than the current size of the file shall expand the size of the file but the value of the new octets of the file shall be a local matter.

Devices may restrict the allowed values for writes to the File_Size. Specifically, devices may allow deletion of the file contents by writing a value of zero, but not necessarily allow arbitrary truncation or expansion.

Any change to the File_Size property shall also update the Modification_Date property to reflect the date when the size was changed.

If the size of the file is unknown, an attempt to read this property shall result in an 'Error Class' of PROPERTY and an 'Error Code' of UNKNOWN_FILE_SIZE.

12.13.7 Modification_Date

This property, of type BACnetDateTime, indicates the last time this object's underlying file data or File_Size was modified. A File object shall be considered modified when any of these conditions occur:

(a) the File object is created;
(b) the underlying file data that it represents is written using AtomicWriteFile or ConfirmedPrivateTransfer or UnconfirmedPrivateTransfer services or is written by some internal process; or
(c) the File_Size property changes.

If the Local_Time and Local_Date properties are present, this property shall contain a specific datetime value; otherwise this property shall contain either a specific or an unspecified datetime value.

12.13.8 Archive

This property, of type BOOLEAN, shall be set to FALSE when the Modification_Date property changes for any reason. It is the responsibility of an archiving process to set the value of this property to TRUE when it completes. The mechanism by which archiving occurs shall be a local matter.

12.13.9 Read_Only

This property, of type BOOLEAN, indicates whether (FALSE) or not (TRUE) the file data may be changed through the use of the AtomicWriteFile service.

12.13.10 File_Access_Method

This property, of type BACnetFileAccessMethod, indicates the type(s) of file access supported for this object. The possible values for File_Access_Method are:

{RECORD_ACCESS, STREAM_ACCESS}

12.13.11 Record_Count

This property, of type Unsigned, indicates the size of the file data in records. The Record_Count property shall be present only if File_Access_Type is RECORD_ACCESS. If the number of records can be changed by writing to the file, then this property shall be writable.

Writing to the Record_Count property with a value less than the current size of the file shall truncate the file at the specified position. Writing a Record_Count of 0 shall delete all of the file data but not the File object itself. Writing to the Record_Count property with a value greater than the current size of the file shall expand the size of the file but the value of the new octets of the file shall be a local matter.

Devices may restrict the allowed values for writes to the Record_Count. Specifically, devices may allow deletion of the file contents by writing a value of zero, but not necessarily allow arbitrary truncation or expansion.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.13.12 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.13.13 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.13.14 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.13.15 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.13.16 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.13.17 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.14 Group Object Type

The Group object type defines a standardized object whose properties represent a collection of other objects and one or more of their properties. A group object is used to simplify the exchange of information between BACnet devices by providing a shorthand way to specify all members of the group at once. A group may be formed using any combination of object types. The group object and its properties are summarized in Table 12-17 and described in detail in this clause.

Table 12-17. Properties of the Group Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
List_Of_Group_Members	BACnetLIST of ReadAccessSpecification	R
Present_Value	BACnetLIST of ReadAccessResult	R
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

10 This property shall be present only if the device supports audit reporting.

12.14.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.14.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.14.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be GROUP.

12.14.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.14.5 List_Of_Group_Members

This property, of type BACnetLIST of ReadAccessSpecification, is a list of one or more read access specifications, which defines the members of the group that shall be referenced when this object is specified in a protocol transaction. Each read access specification shall consist of two parts: 1) an Object_Identifier and 2) a List Of Property References. All members of the group shall be objects that reside in the same device that maintains the Group object. See the ASN.1 production for ReadAccessSpecification in Clause 21.

Nesting of group objects is not permitted; that is, the List_Of_Group_Members shall not refer to the Present_Value property of a Group object or a Global Group object.

12.14.6 Present_Value

This property, of type BACnetLIST of ReadAccessResult, is a list that contains the values of all the properties specified in the List_Of_Group_Members. This is a "read-only" property; it cannot be used to write a set of values to the members of the group. The Present_Value list shall be reconstructed each time the property is read by fetching the member

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

properties. (NOTE: This requirement is to reduce concurrency problems that could result if the Present_Value were stored.)

12.14.7 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.14.8 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.14.9 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.14.10 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.14.11 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.14.12 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.15 Life Safety Point Object Type

The Life Safety Point object type defines a standardized object whose properties represent the externally visible characteristics associated with initiating and indicating devices in fire, life safety and security applications. The condition of a Life Safety Point object is represented by a mode and a state.

Mode changes determine the object's inner logic and, consequently, influence the evaluation of the state. Typically, the operating mode would be under operator control.

The state of the object represents the condition of the controller according to the logic internal to the device. The implementation of the logic applied to such controllers to determine the various possible states is a local matter.

Typical applications of the Life Safety Point object include automatic fire detectors, pull stations, sirens, supervised printers, etc. Similar objects can be identified in security control panels.

Life Safety Point objects that support intrinsic reporting shall apply the CHANGE_OF_LIFE_SAFETY event algorithm.

For reliability-evaluation, the FAULT_LIFE_SAFETY fault algorithm can be applied.

The Life Safety Point object type and its properties are summarized in Table 12-18 and described in detail in this clause.

Table 12-18. Properties of the Life Safety Point Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetLifeSafetyState	R
Tracking_Value	BACnetLifeSafetyState	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R[1]
Out_Of_Service	BOOLEAN	R
Mode	BACnetLifeSafetyMode	W
Accepted_Modes	BACnetLIST of BACnetLifeSafetyMode	R
Time_Delay	Unsigned	O[2,5]
Notification_Class	Unsigned	O[2,5]
Life_Safety_Alarm_Values	BACnetLIST of BACnetLifeSafetyState	O[2,5]
Alarm_Values	BACnetLIST of BACnetLifeSafetyState	O[2,5]
Fault_Values	BACnetLIST of BACnetLifeSafetyState	O
Event_Enable	BACnetEventTransitionBits	O[2,5]
Acked_Transitions	BACnetEventTransitionBits	O[2,5]
Notify_Type	BACnetNotifyType	O[2,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[2,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Silenced	BACnetSilencedState	R
Operation_Expected	BACnetLifeSafetyOperation	R
Maintenance_Required	BACnetMaintenance	O
Setting	Unsigned8	O
Direct_Reading	REAL	O[3]
Units	BACnetEngineeringUnits	O[3]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Member_Of	BACnetLIST of BACnetDeviceObjectReference	O
Floor_Number	Unsigned8	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[7,8,9]
Audit_Level	BACnetAuditLevel	O[10]
Auditable_Operations	BACnetAuditOperationFlags	O[10]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if the object supports intrinsic alarming.
[3] If either of these properties is present, then both must be present.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] This property is required if the object supports the value source mechanism.
[8] This property shall be present only if the object supports the value source mechanism.
[9] This property shall be writable as described in Clause 19.5.
[10] This property shall be present only if the device supports audit reporting.

12.15.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.15.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.15.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LIFE_SAFETY_POINT.

12.15.4 Present_Value

This property, of type BACnetLifeSafetyState, reflects the state of the Life Safety Point object. The means of deriving the Present_Value shall be a local matter. Present_Value may latch non-NORMAL state values until reset.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, then this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.15.5 Tracking_Value

This property, of type BACnetLifeSafetyState, reflects the non-latched state of the Life Safety Point object. The means of deriving the state shall be a local matter. Unlike Present_Value, which may latch non-NORMAL state values until reset, Tracking_Value shall continuously track changes in the state. The Tracking_Value property shall be writable when Out_Of_Service is TRUE.

12.15.6 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.15.7 Device_Type

This property, of type CharacterString, is a text description of the physical device that the Life Safety Point object represents.

12.15.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Life Safety Point object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical input(s) are no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical input(s). Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.9 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.15.10 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input(s) in question are "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.15.11 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the input(s) or process the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Tracking_Value property are decoupled from the input(s) or process;
(b) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the input or process;
(c) the Tracking_Value property and the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Tracking_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred to the input(s) or process.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.15.12 Mode

This property, of type BACnetLifeSafetyMode, shall convey the desired operating mode for the object.

If the object supports event reporting, then this property shall be the pMode parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.13 Accepted_Modes

This read-only property, of type BACnetLIST of BACnetLifeSafetyMode, shall specify all values the Mode property accepts when written to using BACnet services. Even though a mode is listed in this property, the write may be denied by the object due to the internal state of the object at that time. The value of the Accepted_Modes property does not depend on the internal state of the object and shall not change when the internal state changes. If a write is denied, a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned. Internal computation in the object may set the Mode property to a value other than one of those listed in the Accepted_Modes property.

12.15.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.15.16 Life_Safety_Alarm_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the pLifeSafetyAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.17 Alarm_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.18 Fault_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.15.19 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.15.20 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.15.21 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.15.22 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.15.23 Silenced

This property, of type BACnetSilencedState, shall indicate whether the most recently occurring transition for this object that has produced an audible or visual indication has been silenced by the receipt of a LifeSafetyOperation service request or a local process.

12.15.24 Operation_Expected

The Operation_Expected property, of type BACnetLifeSafetyOperation, shall specify the next operation expected by this object to handle a specific life safety situation.

If the object supports event reporting, then this property shall be the pOperationExpected parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.25 Maintenance_Required

This property, of type BACnetMaintenance, shall indicate the type of maintenance required for the life safety point. This may be periodic maintenance, or a "parameter-determined" maintenance, such as dirtiness value for an associated detector, and shall be determined locally.

12.15.26 Setting

This property, of type Unsigned8, shall be used to convey the desired setting of the input(s) or process used to determine the logical state of the Present_Value. The range of the Setting property shall be from 0 (least sensitive) to 100 (most sensitive). The interpretation of the setting and the actual number of useful settings for a given Life Safety Point object shall be a local matter.

12.15.27 Direct_Reading

This property, of type REAL, shall indicate an analog quantity that reflects the measured or calculated reading from an initiating device. The manner in which this reading is used to determine the logical state of the object shall be a local matter. If this property is present, then the Units property shall also be present.

12.15.28 Units

This property, of type BACnetEngineeringUnits, shall indicate the units of the quantity represented by the Direct_Reading property. If this property is present, then the Direct_Reading property shall also be present.

12.15.29 Member_Of

This property, of type BACnetLIST of BACnetDeviceObjectReference, shall indicate those Life Safety Zone objects of which this Life Safety Point object is considered to be a zone member. Each object in the Member_Of list shall be a Life Safety Zone object.

This property may be restricted to only support references to objects inside of the device containing the Life Safety Point object. If the property is writable and is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

12.15.30 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.15.31 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.15.32 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.15.33 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.15.34 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.15.35 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.15.36 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.15.37 Floor_Number

This property, of type Unsigned8, indicates the universal floor number of the floor on which the life safety point this object represents is present.

12.15.38 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.15.39 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Mode property. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.15.40 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.15.41 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.15.42 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.15.43 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.15.44 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.16 Life Safety Zone Object Type

The Life Safety Zone object type defines a standardized object whose properties represent the externally visible characteristics associated with an arbitrary group of BACnet Life Safety Point and Life Safety Zone objects in fire, life safety and security applications. The condition of a Life Safety Zone object is represented by a mode and a state.

Mode changes determine the object's inner logic and, consequently, influence the evaluation of the state. Typically, the operating mode would be under operator control.

The state of the object represents the condition of the controller according to the logic internal to the device. The implementation of the logic applied to such controllers to determine the various possible states is a local matter.

Typical applications of the Life Safety Zone object include fire zones, panel zones, detector lines, extinguishing controllers, remote transmission controllers, etc. Similar objects can be identified in security control panels.

Life Safety Zone objects that support intrinsic reporting shall apply the CHANGE_OF_LIFE_SAFETY event algorithm.

For reliability-evaluation, the FAULT_LIFE_SAFETY fault algorithm can be applied.

The Life Safety Zone object type and its properties are summarized in Table 12-19 and described in detail in this clause.

Table 12-19. Properties of the Life Safety Zone Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetLifeSafetyState	R
Tracking_Value	BACnetLifeSafetyState	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R[1]
Out_Of_Service	BOOLEAN	R
Mode	BACnetLifeSafetyMode	W
Accepted_Modes	BACnetLIST of BACnetLifeSafetyMode	R
Time_Delay	Unsigned	O[2,4]
Notification_Class	Unsigned	O[2,4]
Life_Safety_Alarm_Values	BACnetLIST of BACnetLifeSafetyState	O[2,4]
Alarm_Values	BACnetLIST of BACnetLifeSafetyState	O[2,4]
Fault_Values	BACnetLIST of BACnetLifeSafetyState	O
Event_Enable	BACnetEventTransitionBits	O[2,4]
Acked_Transitions	BACnetEventTransitionBits	O[2,4]
Notify_Type	BACnetNotifyType	O[2,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Event_Detection_Enable	BOOLEAN	O[2,4]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[4]
Event_Algorithm_Inhibit	BOOLEAN	O[4,5]
Time_Delay_Normal	Unsigned	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Silenced	BACnetSilencedState	R
Operation_Expected	BACnetLifeSafetyOperation	R
Maintenance_Required	BOOLEAN	O
Zone_Members	BACnetLIST of BACnetDeviceObjectReference	R
Member_Of	BACnetLIST of BACnetDeviceObjectReference	O

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Floor_Number	Unsigned8	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[6,7,8]
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if the object supports intrinsic alarming.
[3] Footnote removed.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[6] This property is required if the object supports the value source mechanism.
[7] This property shall be present only if the object supports the value source mechanism.
[8] This property shall be writable as described in Clause 19.5.
[9] This property shall be present only if the device supports audit reporting.

12.16.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.16.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.16.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LIFE_SAFETY_ZONE.

12.16.4 Present_Value

This property, of type BACnetLifeSafetyState, reflects the state of the Life Safety Zone object. The means of deriving the Present_Value shall be a local matter. Present_Value may latch non-NORMAL state values until reset.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, then this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.16.5 Tracking_Value

This property, of type BACnetLifeSafetyState, reflects the non-latched state of the Life Safety Zone object. The means of deriving the state shall be a local matter. Unlike Present_Value, which may latch non-NORMAL state values until reset, Tracking_Value shall continuously track changes in the state. The Tracking_Value property shall be writable when Out_Of_Service is TRUE.

12.16.6 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.16.7 Device_Type

This property, of type CharacterString, is a text description of the physical zone or area that the Life Safety Zone object represents. It will typically be used to describe the locale of the Life Safety Point objects that are Zone_Members of the Life Safety Zone object.

12.16.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Life Safety Zone object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical input(s) are no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical input(s). Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.9 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.16.10 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input(s) in question are "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.16.11 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the input(s) or process the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Tracking_Value property are decoupled from the input(s) or process;
(b) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the input or process;
(c) the Tracking_Value property and the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Tracking_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred to the input(s) or process.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.16.12 Mode

This property, of type BACnetLifeSafetyMode, shall convey the desired operating mode for the object.

If the object supports event reporting, then this property shall be the pMode parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.13 Accepted_Modes

This read-only property, of type BACnetLIST of BACnetLifeSafetyMode, shall specify all values the Mode property accepts when written to using BACnet services. Even though a mode is listed in this property, the write may be denied by the object due to the internal state of the object at that time. The value of the Accepted_Modes property does not depend on the internal state of the object and shall not change when the internal state changes. If a write is denied, a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned. Internal computation in the object may set the Mode property to a value other than one of those listed in the Accepted_Modes property.

12.16.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.16.16 Life_Safety_Alarm_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the pLifeSafetyAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.17 Alarm_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.18 Fault_Values

This property, of type BACnetLIST of BACnetLifeSafetyState, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.16.19 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.16.20 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.16.21 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.16.22 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.16.23 Silenced

This property, of type BACnetSilencedState, shall indicate whether the most recently occurring transition for this object that has produced an audible or visual indication has been silenced by the receipt of a LifeSafetyOperation service request or a local process.

12.16.24 Operation_Expected

The Operation_Expected property, of type BACnetLifeSafetyOperation, shall specify the next operation expected by this object to handle a specific life safety situation.

If the object supports event reporting, then this property shall be the pOperationExpected parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.25 Maintenance_Required

This property, of type BOOLEAN, shall indicate that maintenance is required for one or more of the life safety points that are members of this zone.

12.16.26 Zone_Members

This property, of type BACnetLIST of BACnetDeviceObjectReference, shall indicate which Life Safety Point and Life Safety Zone objects are members of the zone represented by this object.

This property may be restricted to only support references to objects inside of the device containing the Life Safety Zone object. If the property is writable and is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

12.16.27 Member_Of

This property, of type BACnetLIST of BACnetDeviceObjectReference, shall indicate those Life Safety Zone objects of which this Life Safety Zone object is considered to be a zone member. Each object in the Member_Of list shall be a Life Safety Zone object.

This property may be restricted to only support references to objects inside of the device containing the Life Safety Zone object. If the property is writable and is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

12.16.28 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.16.29 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.16.30 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.16.31 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.16.32 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.16.33 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.16.34 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.16.35 Floor_Number

This property, of type Unsigned8, indicates the universal floor number of the floor on which the life safety zone this object represents is present.

12.16.36 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.16.37 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Mode property. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.16.38 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.16.39 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.16.40 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.16.41 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.16.42 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17 Loop Object Type

The Loop object type defines a standardized object whose properties represent the externally visible characteristics of any form of feedback control loop. Flexibility is achieved by providing three independent gain constants with no assumed values for units. The appropriate gain units are determined by the details of the control algorithm, which is a local matter.

Loop objects that support intrinsic reporting shall apply the FLOATING_LIMIT event algorithm.

The Loop object type and its properties are summarized in Table 12-20 and described in detail in this clause. Figure 12-2 illustrates the relationship between the Loop object properties and the other objects referenced by the loop.

Table 12-20. Properties of the Loop Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	R[7]
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O[7]
Out_Of_Service	BOOLEAN	R
Update_Interval	Unsigned	O
Output_Units	BACnetEngineeringUnits	R
Manipulated_Variable_Reference	BACnetObjectPropertyReference	R
Controlled_Variable_Reference	BACnetObjectPropertyReference	R
Controlled_Variable_Value	REAL	R
Controlled_Variable_Units	BACnetEngineeringUnits	R
Setpoint_Reference	BACnetSetpointReference	R
Setpoint	REAL	R
Action	BACnetAction	R
Proportional_Constant	REAL	O[1]
Proportional_Constant_Units	BACnetEngineeringUnits	O[1]
Integral_Constant	REAL	O[2]
Integral_Constant_Units	BACnetEngineeringUnits	O[2]
Derivative_Constant	REAL	O[3]
Derivative_Constant_Units	BACnetEngineeringUnits	O[3]
Bias	REAL	O
Maximum_Output	REAL	O
Minimum_Output	REAL	O
Priority_For_Writing	Unsigned(1..16)	R
COV_Increment	REAL	O[4]
Time_Delay	Unsigned	O[5,8]
Notification_Class	Unsigned	O[5,8]
Error_Limit	REAL	O[5,8]
Deadband	REAL	O[5,8]
Event_Enable	BACnetEventTransitionBits	O[5,8]
Acked_Transitions	BACnetEventTransitionBits	O[5,8]
Notify_Type	BACnetNotifyType	O[5,8]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[5,8]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[8]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[8]
Event_Detection_Enable	BOOLEAN	O[5,8]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[8]
Event_Algorithm_Inhibit	BOOLEAN	O[8,9]
Time_Delay_Normal	Unsigned	O[8]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Reliability_Evaluation_Inhibit	BOOLEAN	O[10]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Low_Diff_Limit	BACnetOptionalREAL	O
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If one of these optional properties is present, then both of these properties shall be present.
[2] If one of these optional properties is present, then both of these properties shall be present.
[3] If one of these optional properties is present, then both of these properties shall be present.
[4] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[5] These properties are required if the object supports intrinsic reporting.
[6] Footnote removed.
[7] These properties are required to be writable when Out_Of_Service is TRUE.
[8] These properties shall be present only if the object supports intrinsic reporting.
[9] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[10] If this property is present, then the Reliability property shall be present.
[11] This property shall be present only if the device supports audit reporting.

12.17.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.17.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.17.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LOOP.

12.17.4 Present_Value

This property indicates the current output value of the loop algorithm in units of the Output_Units property. The Present_Value property shall be writable when Out_Of_Service is TRUE.

12.17.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.17.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the loop. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise Logical TRUE (1).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

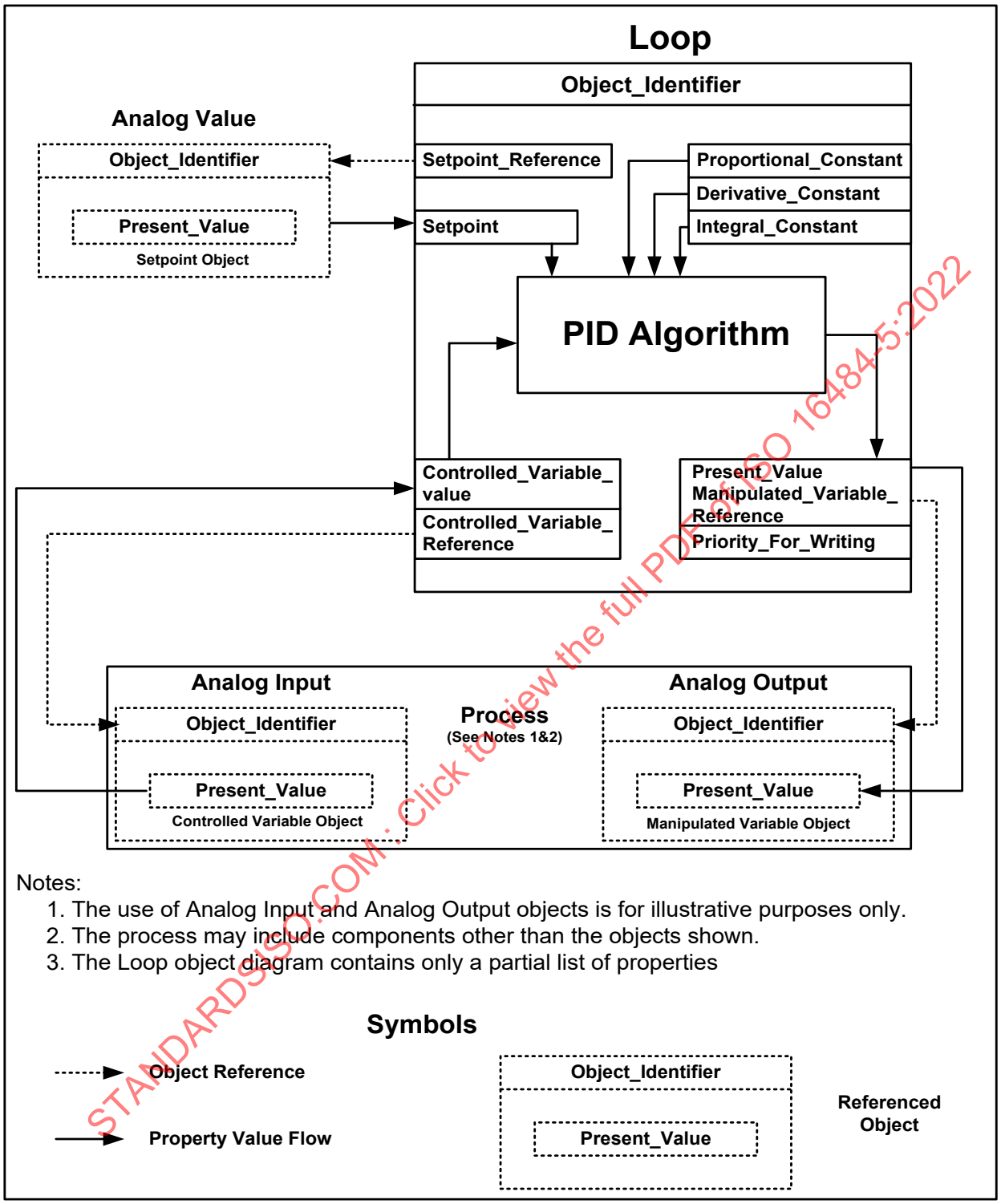
© ISO 2022 – All rights reserved

FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved



STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Figure 12-2. Loop Object Structure with its Referenced Objects.

12.17.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

© ISO 2022 – All rights reserved

12.17.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value of the loop in question is reliable as far as the BACnet device or operator can determine and, if not, why.

12.17.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the algorithm this object represents is or is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value property shall be decoupled from the algorithm;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the algorithm;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) the property referenced by Manipulated_Variable_Reference and other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties, as if those changes had been made by the algorithm.

12.17.10 Update_Interval

This property, of type Unsigned, indicates the interval in milliseconds at which the loop algorithm updates the output (Present_Value property).

NOTE: No property that represents the interval at which the process variable is sampled or the algorithm is executed is part of this object. The Update_Interval value may be the same as these other values but could also be different depending on the algorithm utilized. The sampling or execution interval is a local matter and need not be represented as part of this object.

12.17.11 Output_Units

This property, of type BACnetEngineeringUnits, indicates the engineering units for the output (Present_Value property) of this control loop.

12.17.12 Manipulated_Variable_Reference

This property is of type BACnetObjectPropertyReference. The output (Present_Value) of the control loop is written to the object and property designated by the Manipulated_Variable_Reference. It is normally the Present_Value of an Analog Output object used to position a device, but it could also be another object or property, such as that used to stage multiple Binary Outputs.

12.17.13 Controlled_Variable_Reference

This property is of type BACnetObjectPropertyReference. The Controlled_Variable_Reference identifies the property used to set the Controlled_Variable_Value property of the Loop object. It is normally the Present_Value property of an Analog Input object used for measuring a process variable, temperature, for example, but it could also be another object, such as an Analog Value, which calculates a minimum or maximum of a group of Analog Inputs for use in discriminator control.

12.17.14 Controlled_Variable_Value

This property, of type REAL, is the value of the property of the object referenced by the Controlled_Variable_Reference property. This control loop compares the Controlled_Variable_Value with the Setpoint to calculate the error.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.17.15 Controlled_Variable_Units

This property, of type BACnetEngineeringUnits, indicates the engineering units for the Controlled_Variable_Value property of this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17.16 Setpoint_Reference

This property, of type BACnetSetpointReference, contains zero or one references. The absence of a reference indicates that the setpoint for this control loop is fixed and is contained in the Setpoint property. The presence of a reference indicates that the property of another object contains the setpoint value used for this Loop object and the reference specifies that property.

12.17.17 Setpoint

This property, of type REAL, is the value of the loop setpoint or of the property of the object referenced by the Setpoint_Reference, expressed in units of the Controlled_Variable_Units property.

If the object supports event reporting, then this property shall be the pSetpoint parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.17.18 Action

This property, of type BACnetAction, defines whether the loop is DIRECT or REVERSE acting.

12.17.19 Proportional_Constant

This property, of type REAL, is the value of the proportional gain parameter used by the loop algorithm. It may be used to represent any of the various forms of gain for the proportional control mode, such as overall gain, throttling range, or proportional band. If either the Proportional_Constant property or the Proportional_Constant_Units property are present, then both of these properties shall be present.

12.17.20 Proportional_Constant_Units

This property, of type BACnetEngineeringUnits, indicates the engineering units of the Proportional_Constant property of this object. If either the Proportional_Constant_Units property or the Proportional_Constant property are present, then both of these properties shall be present.

12.17.21 Integral_Constant

This property, of type REAL, is the value of the integral gain parameter used by the loop algorithm. It may be used to represent any of the various forms of gain for the integral control mode, such as reset time or rate. If either the Integral_Constant property or the Proportional_Constant_Units property are present, then both of these properties shall be present.

12.17.22 Integral_Constant_Units

This property, of type BACnetEngineeringUnits, indicates the engineering units of the Integral_Constant property of this object. If either the Integral_Constant_Units property or the Proportional_Constant property are present, then both of these properties shall be present.

12.17.23 Derivative_Constant

This property, of type REAL, is the value of the derivative gain parameter used by the loop algorithm. It may be used to represent any of the various forms of gain for the derivative control mode, such as derivative time or rate time. If either the Derivative_Constant property or the Derivative_Constant_Units property are present, then both of these properties shall be present.

12.17.24 Derivative_Constant_Units

This property, of type BACnetEngineeringUnits, indicates the engineering units of the Derivative_Constant property of this object. If either the Derivative_Constant_Units property or the Derivative_Constant property are present, then both of these properties shall be present.

12.17.25 Bias

This property, of type REAL, is the bias value used by the loop algorithm expressed in units of the Output_Units property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17.26 Maximum_Output

This property, of type REAL, is the maximum value of the Present_Value property as limited by the PID loop algorithm. It is normally used to prevent the algorithm from controlling beyond the range of the controlled device and to prevent integral term "windup."

12.17.27 Minimum_Output

This property, of type REAL, is the minimum value of the Present_Value property as limited by the loop algorithm. It is normally used to prevent the algorithm from controlling beyond the range of the controlled device and to prevent integral term "windup."

12.17.28 Priority_For_Writing

Loop objects may be used to control the commandable property of an object. This property, of type Unsigned, provides a priority to be used by the command prioritization mechanism. It identifies the particular priority slot in the Priority_Array of the Manipulated_Variable_Reference that is controlled by this loop. It shall have a value in the range 1..16.

12.17.29 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.17.30 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.17.31 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.17.32 Error_Limit

This property is both the pLowDiffLimit and pHighDiffLimit parameter for the object's event algorithm if the property Low_Diff_Limit is not present or has a value of NULL.

If the property Low_Diff_Limit is present and has a value other than NULL, then Error_Limit is the pHighDiffLimit parameter for the object's event algorithm.

See Clause 13.3 for event algorithm parameter descriptions.

12.17.33 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.17.34 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.17.35 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17.36 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.17.37 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.17.38 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.17.39 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.17.40 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.17.41 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.17.42 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.17.43 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17.44 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.17.45 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.17.46 Low_Diff_Limit

This property, of type BACnetOptionalREAL, is the pLowDiffLimit parameter for the object's FLOATING_LIMIT event algorithm if it has a value other than NULL.

If this property has the value NULL, then the value of the Error_Limit property is used as the pLowDiffLimit parameter for the object's FLOATING_LIMIT event algorithm.

See Clause 13.3 for event algorithm parameter descriptions.

12.17.47 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.17.48 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.17.49 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.17.50 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.17.51 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.18 Multi-state Input Object Type

The Multi-state Input object type defines a standardized object whose Present_Value represents the result of an algorithmic process within the BACnet device in which the object resides. The algorithmic process itself is a local matter and is not defined by the protocol. For example, the Present_Value or state of the Multi-state Input object may be the result of a logical combination of multiple binary inputs or the threshold of one or more analog inputs or the result of a mathematical computation. The Present_Value property is an integer number representing the state. The State_Text property associates a description with each state.

Multi-state Input objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

For reliability-evaluation, the FAULT_STATE fault algorithm can be applied.

The Multi-state Input object type and its properties are summarized in Table 12-21 and described in detail in this clause.

Table 12-21. Properties of the Multi-state Input Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Number_Of_States	Unsigned	R
State_Text	BACnetARRAY[N] of CharacterString	O
Time_Delay	Unsigned	O[3,5]
Notification_Class	Unsigned	O[3,5]
Alarm_Values	BACnetLIST of Unsigned	O[3,5]
Fault_Values	BACnetLIST of Unsigned	O[7]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalUnsigned	O
Audit_Level	BACnetAuditLevel	O[8]
Auditable_Operations	BACnetAuditOperationFlags	O[8]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] Footnote removed.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.
[8] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.18.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.18.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.18.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be MULTISTATE_INPUT.

12.18.4 Present_Value

This property, of type Unsigned, reflects the logical state of the input. The logical state of the input shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. The means used to determine the current state is a local matter. The Present_Value property shall always have a value greater than zero. The Present_Value property shall be writable when Out_Of_Service is TRUE. Any local modification of the value of the Present_Value when the Number_Of_States property is changed is a local matter.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.18.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.18.6 Device_Type

This property, of type CharacterString, is a text description of the multi-state input. It will typically be used to describe the type of device attached to the multi-state input.

12.18.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the multi-state input. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.18.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.18.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical inputs in question are "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.18.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical input that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the physical input and will not track changes to the physical input;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical input;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred in the physical input.

12.18.11 Number_Of_States

This property, of type Unsigned, defines the number of states the Present_Value may have. The Number_Of_States property shall always have a value greater than zero. If the value of this property is changed, the size of the State_Text array, if present, shall also be changed to the same value. If the Number_of_States property value becomes less than the value of the Present_Value, the object shall have a Reliability of MULTI_STATE_OUT_OF_RANGE as long as this situation remains, unless the object is out of service. It is a local matter whether Present_Value is modified when the Number_of_States property value becomes less than its current value.

12.18.12 State_Text

This property, of type BACnetARRAY of CharacterString, contains strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array. If the size of this array is changed, the Number_Of_States property shall also be changed to the same value.

12.18.13 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.18.14 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.18.15 Alarm_Values

This property, of type BACnetLIST of Unsigned, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.18.16 Fault_Values

This property, of type BACnetLIST of Unsigned, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.18.17 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.18.18 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.18.19 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.18.20 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.18.21 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.18.22 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.18.23 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.18.24 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.18.25 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.18.26 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.18.27 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.18.28 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.18.29 Interface_Value

This read-only property, of type BACnetOptionalUnsigned, reflects the logical state of the physical input. If the BACnet device is not capable of knowing the logical state of the physical input, then the value of this property shall be NULL. Otherwise, the logical state of this property shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. The Interface_Value property shall always have a value greater than zero, if not NULL.

12.18.30 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.18.31 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.18.32 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.18.33 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.18.34 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.19 Multi-state Output Object Type

The Multi-state Output object type defines a standardized object whose properties represent the desired state of one or more physical outputs or processes within the BACnet device in which the object resides. The actual functions associated with a specific state are a local matter and not specified by the protocol. For example, a particular state may represent the active/inactive condition of several physical outputs or perhaps the value of an analog output. The Present_Value property is an unsigned integer number representing the state. The State_Text property associates a description with each state.

Multi-state Output objects that support intrinsic reporting shall apply the COMMAND_FAILURE event algorithm.

The Multi-state Output object type and its properties are summarized in Table 12-22 and described in detail in this clause.

Table 12-22. Properties of the Multi-state Output Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	W
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Number_Of_States	Unsigned	R
State_Text	BACnetARRAY[N] of CharacterString	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	Unsigned	R
Time_Delay	Unsigned	O[1,3]
Notification_Class	Unsigned	O[1,3]
Feedback_Value	Unsigned	O[1]
Event_Enable	BACnetEventTransitionBits	O[1,3]
Acked_Transitions	BACnetEventTransitionBits	O[1,3]
Notify_Type	BACnetNotifyType	O[1,3]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,3]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[3]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[3]
Event_Detection_Enable	BOOLEAN	O[1,3]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[3]
Event_Algorithm_Inhibit	BOOLEAN	O[3,4]
Time_Delay_Normal	Unsigned	O[3]
Reliability_Evaluation_Inhibit	BOOLEAN	O[5]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Interface_Value	BACnetOptionalUnsigned	O
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[2] Footnote removed.
[3] These properties shall be present only if the object supports intrinsic reporting.
[4] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[5] If this property is present, then the Reliability property shall be present.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.
[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.19.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.19.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.19.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be MULTISTATE_OUTPUT.

12.19.4 Present_Value (Commandable)

This property, of type Unsigned, reflects the logical state of an output. The logical state of the output shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. How the Present_Value is used is a local matter. Any local modification of the value of the Present_Value when the Number_Of_States property is changed is a local matter. The Present_Value property shall always have a value greater than zero.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.19.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.19.6 Device_Type

This property, of type CharacterString, is a text description of the physical device connected to the multi-state output. It will typically be used to describe the type of device attached to the multi-state output.

12.19.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the multi-state output. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.19.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.19.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical outputs in question are "reliable" as far as the BACnet device or operator can determine and, if not, why.

12.19.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical point that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the physical output;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical output;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred to the physical output;
(e) the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

12.19.11 Number_Of_States

This property, of type Unsigned, defines the number of states the Present_Value may have. The Number_Of_States property shall always have a value greater than zero. If the value of this property is changed, the size of the State_Text array, if present, shall also be changed to the same value. If the Number_of_States property value becomes less than the value of the Present_Value, the object shall have a Reliability of MULTI_STATE_OUT_OF_RANGE as long as this situation remains, unless the object is out of service. It is a local matter whether Priority_Array, Relinquish_Default, Present_Value, and Feedback_Value properties are modified when the Number_of_States property value becomes less than their current values. If any of those properties other than Present_Value are out of range, the value of the Reliability property shall remain CONFIGURATION_ERROR, unless the object is out of service.

12.19.12 State_Text

This property, of type BACnetARRAY of CharacterString, contains strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array. If the size of this array is changed, the Number_Of_States property shall also be changed to the same value.

12.19.13 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism. Any local modification of the values in the Priority_Array when the Number_Of_States property is changed is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.19.14 Relinquish_Default

This property, of type Unsigned, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2. Any local modification of the value of the Relinquish_Default when the Number_Of_States property is changed is a local matter.

12.19.15 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.19.16 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.19.17 Feedback_Value

This property is an indication of the actual value of the entity controlled by Present_Value. The manner by which the Feedback_Value is determined shall be a local matter.

If the object supports event reporting, then this property is the pFeedback parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.19.18 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.19.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.19.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.19.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.19.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.19.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.19.24 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.19.25 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.19.26 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.19.27 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.19.28 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.19.29 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.19.30 Interface_Value

This read-only property, of type BACnetOptionalUnsigned, reflects the logical state of the physical output. If the BACnet device is not capable of knowing the logical state of the physical output, then the value of this property shall be NULL. Otherwise, the logical state of this property shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. The Interface_Value property shall always have a value greater than zero, if not NULL.

12.19.31 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.19.32 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.19.33 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.19.34 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.19.35 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.19.36 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.19.37 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.19.38 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.19.39 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.19.40 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.19.41 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.20 Multi-state Value Object Type

The Multi-state Value object type defines a standardized object whose properties represent the externally visible characteristics of a multi-state value. A "multi-state value" is a control system parameter residing in the memory of the BACnet device. The actual functions associated with a specific state are a local matter and not specified by the protocol. For example, a particular state may represent the active/inactive condition of several physical inputs and outputs or perhaps the value of an analog input or output. The Present_Value property is an unsigned integer number representing the state. The State_Text property associates a description with each state.

Multi-state Value objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

For reliability-evaluation, the FAULT_STATE fault algorithm can be applied.

The Multi-state Value object type and its properties are summarized in Table 12-23 and described in detail in this clause.

Table 12-23. Properties of the Multi-state Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	R[1]
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Number_Of_States	Unsigned	R
State_Text	BACnetARRAY[N] of CharacterString	O
Priority_Array	BACnetPriorityArray	O[3]
Relinquish_Default	Unsigned	O[3]
Time_Delay	Unsigned	O[4,6]
Notification_Class	Unsigned	O[4,6]
Alarm_Values	BACnetLIST of Unsigned	O[4,6]
Fault_Values	BACnetLIST of Unsigned	O[8]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	O[3]
Value_Source	BACnetValueSource	O[9,11,13]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[10,12]
Last_Command_Time	BACnetTimeStamp	O[10,12]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[12]
Audit_Level	BACnetAuditLevel	O[14]
Auditable_Operations	BACnetAuditOperationFlags	O[14]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[14]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[1] If Present_Value is commandable, then it is required to also be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] Footnote removed.
[3] These properties are required if, and shall be present only if, Present_Value is commandable.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] This property is required if the object supports the value source mechanism.
[10] These properties are required if the object supports the value source mechanism and is commandable.
[11] This property shall be present only if the object supports the value source mechanism.
[12] These properties shall be present only if the object supports the value source mechanism and is commandable.
[13] This property shall be writable as described in Clause 19.5.
[14] This property shall be present only if the device supports audit reporting.

12.20.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.20.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.20.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be MULTISTATE_VALUE.

12.20.4 Present_Value

This property, of type Unsigned, reflects the logical state of the multi-state value. The logical state of the multi-state value shall be one of 'n' states, where 'n' is the number of states defined in the Number_Of_States property. How the Present_Value is used is a local matter. The Present_Value property shall always have a value greater than zero. Present_Value shall be optionally commandable. If Present_Value is commandable for a given object instance, then the Priority_Array and Relinquish_Default properties shall also be present for that instance. The Present_Value property shall be writable when Out_Of_Service is TRUE. Any local modification of the value of the Present_Value when the Number_Of_States property is changed is a local matter.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, then this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.20.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.20.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the multi-state value. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the physical output. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.20.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.20.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause13.4 for fault algorithm parameter descriptions.

12.20.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.20.10 Number_Of_States

This property, of type Unsigned, defines the number of states the Present_Value may have. The Number_Of_States property shall always have a value greater than zero. If the value of this property is changed, the size of the State_Text array, if present, shall also be changed to the same value. If the Number_of_States property value becomes less than the value of the Present_Value, the object shall have a Reliability of MULTI_STATE_OUT_OF_RANGE as long as this situation remains, unless the object is out of service. It is a local matter whether Priority_Array, Relinquish_Default, Present_Value, Alarm_Values, and Fault_Values properties, if present, are modified when the Number_of_States property value becomes less than their current values. If any of those properties other than Present_Value are out of range, the value of the Reliability property shall remain CONFIGURATION_ERROR, unless the object is out of service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.20.11 State_Text

This property, of type BACnetARRAY of CharacterString, contains strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array. If the size of this array is changed, the Number_Of_States property shall also be changed to the same value.

12.20.12 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism. Any local modification of the values in the Priority_Array when the Number_Of_States property is changed is a local matter.

12.20.13 Relinquish_Default

This property, of type Unsigned, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2. Any local modification of the value of the Relinquish_Default when the Number_Of_States property is changed is a local matter.

12.20.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.20.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.20.16 Alarm_Values

This property, of type BACnetLIST of Unsigned, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.20.17 Fault_Values

This property, of type BACnetLIST of Unsigned, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.20.18 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.20.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.20.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.20.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.20.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.20.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.20.24 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.20.25 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.20.26 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.20.27 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.20.28 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.20.29 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.20.30 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.20.31 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.20.32 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.20.33 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.20.34 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.20.35 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.20.36 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.20.37 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.20.38 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.20.39 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.20.40 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.21 Notification Class Object Type

The Notification Class object type defines a standardized object that represents and contains information required for the distribution of event notifications within BACnet systems. Notification Classes are useful for event-initiating objects that have identical needs in terms of how their notifications should be handled, what the destination(s) for their notifications should be, and how they should be acknowledged.

A notification class defines how event notifications shall be prioritized in their handling according to TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events; whether these categories of events require acknowledgment (nearly always by a human operator); and what destination devices or processes should receive notifications.

The purpose of prioritization is to provide a means to ensure that alarms or event notifications with critical time considerations are not unnecessarily delayed. The possible range of priorities is 0 - 255. A lower number indicates a higher priority. The priority and the Network Priority (Clause 6.2.2) are associated as defined in Table 13-6. Priorities may be assigned to TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events individually within a notification class.

The purpose of acknowledgment is to provide assurance that a notification has been acted upon by some other agent, rather than simply having been received correctly by another device. In most cases, acknowledgments come from human operators. TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events may, or may not, require individual acknowledgment within a notification class.

It is often necessary for event notifications to be sent to multiple destinations or to different destinations based on the time of day or day of week. Notification Classes may specify a list of destinations, each of which is qualified by time, day of week, and type of handling. A destination specifies a set of days of the week (Monday through Sunday) during which the destination is considered viable by the Notification Class object. In addition, each destination has a FromTime and ToTime, which specify a window using specific times, on those days of the week, during which the destination is viable. If an event that uses a Notification Class object occurs and the day is one of the days of the week that is valid for a given destination and the time is within the window specified in the destination, then the destination shall be sent a notification. Destinations may be further qualified, as applicable, by any combination of the three event transitions TO_OFFNORMAL, TO_FAULT, or TO_NORMAL.

The destination also defines the recipient device to receive the notification and a process within the device. Processes are identified by numeric handles that are only meaningful to the destination device. The administration of these handles is a local matter. The recipient device may be specified by either its unique Device Object_Identifier or its BACnetAddress. In the latter case, a specific node address, a multicast address, or a broadcast address may be used. The destination further specifies whether the notification shall be sent using a confirmed or unconfirmed event notification.

Notification Class objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Notification Class objects that support intrinsic reporting shall apply the NONE event algorithm.

The Notification Class object and its properties are summarized in Table 12-24 and described in detail in this clause.

Table 12-24. Properties of the Notification Class Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Notification_Class	Unsigned	R
Priority	BACnetARRAY[3] of Unsigned	R
Ack_Required	BACnetEventTransitionBits	R
Recipient_List	BACnetLIST of BACnetDestination	R
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Status_Flags	BACnetStatusFlags	O[1]
Event_State	BACnetEventState	O[1]
Reliability	BACnetReliability	O[1]
Event_Detection_Enable	BOOLEAN	O[1,2]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Property Identifier	Property Datatype	Conformance Code
Event_Enable	BACnetEventTransitionBits	O[1,2]
Acked_Transitions	BACnetEventTransitionBits	O[1,2]
Notify_Type	BACnetNotifyType	O[1,2]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,2]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[2]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Audit_Level	BACnetAuditLevel	O[4]
Auditable_Operations	BACnetAuditOperationFlags	O[4]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.
[2] These properties shall be present only if the object supports intrinsic reporting.
[3] If this property is present, then the Reliability property shall be present.
[4] This property shall be present only if the device supports audit reporting.

12.21.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.21.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.21.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be NOTIFICATION_CLASS.

12.21.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.21.5 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.21.6 Priority

This property, of type BACnetARRAY[3] of Unsigned, shall convey the priority to be used for event notifications for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively. Priorities shall range from 0 - 255 inclusive. A lower number indicates a higher priority. The priority and the Network Priority (see Clause 6.2.2) are associated as defined in Table 13-6.

12.21.7 Ack_Required

This property, of type BACnetEventTransitionBits, shall convey three separate flags that represent whether acknowledgment shall be required in notifications generated for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL event transitions, respectively, except in the case of the Access Point or Alert Enrollment object types which behave as directed in Clauses 12.31.40 and 12.52.8.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.21.8 Recipient_List

This property, of type BACnetLIST of BACnetDestination, shall convey a list of zero or more recipient destinations to which notifications shall be sent when event-initiating objects using this class detect the occurrence of an event. These recipient destinations are intended to be relatively permanent, do not expire, and shall be maintained through a power failure or device "restart." The destinations themselves define a structure of parameters that is summarized in Table 12-25.

Table 12-25. Components of a BACnetDestination

Parameter	Type	Description
Valid Days	BACnetDaysOfWeek	The set of days of the week on which this destination may be used between From Time and To Time
From Time, To Time	Time	The window of time (inclusive) during which the destination is viable on the days of the week specified by Valid Days. These values shall be specific times.
Recipient	BACnetRecipient	The destination device(s) to receive notifications
Process Identifier	Unsigned32	The handle of a process within the recipient device that is to receive the event notification
Issue Confirmed Notifications	Boolean	(TRUE) if confirmed notifications are to be sent and (FALSE) if unconfirmed notifications are to be sent
Transitions	BACnet Event Transition Bits	A set of three flags that indicate those transitions {TO_OFFNORMAL, TO_FAULT, TO_NORMAL} for which this recipient is suitable

When combined with local Notification Forwarder objects, a device is able to contain a large number of Notification Class objects while centralizing the Recipient_List information in a small number of local Notification Forwarder objects. If local Notification Forwarder objects are being used to specify all of the recipients for the Notification Class, the Recipient_List property is allowed to be read-only. In this case, the read-only Recipient_List shall contain at least one entry, and all entries in the Recipient_List shall refer to the local device.

If the Recipient_List is not writable and the device is not using local Notification Forwarder objects, then the Recipient_List shall have a length of 1 with the Recipient set to a local broadcast, Valid Days set to all days, From Time set to 00:00:00.0, To_Time set to 23:59:59.99, Process Identifier set to 0, Issue Confirmed Notifications set to FALSE, and all bits in Transitions set to TRUE. When deploying devices configured in this manner, it is expected that a Notification Forwarder will be installed in a different device on the same BACnet network to forward the notifications to recipients that are not on the local network. Note that this implementation choice should not be chosen for devices on data links that are severely impacted by broadcasts.

For writable Recipient_List properties, devices are allowed to restrict the values for the Valid Days, From Time, To_Time, and the Transitions field such that they only accept the configuration that results in all transitions being sent without regard to the current time or date. In such cases, the Valid Days shall be all days, From Time shall be 0:00:00.0, To Time shall be 23:59:59.99, and Transitions shall be (TRUE, TRUE, TRUE). A device shall not otherwise restrict the value of Recipient_List entries.

12.21.9 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.21.10 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Notification Class object. One flag is associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0)
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

12.21.11 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.21.12 Reliability

This property, of type BACnetReliability, provides an indication of whether the Notification Class object is properly configured and is able to perform the functionality it represents. Failing event notifications initiated by this object shall not cause this property to indicate a fault condition.

12.21.13 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.21.14 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.21.15 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.21.16 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.21.17 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.21.18 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.21.19 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.21.20 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.21.21 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.21.22 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.21.23 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.21.24 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.21.25 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.22 Program Object Type

The Program object type defines a standardized object whose properties represent the externally visible characteristics of an application program. In this context, an application program is an abstract representation of a process within a BACnet device, which is executing a particular body of instructions that act upon a particular collection of data structures. The logic that is embodied in these instructions and the form and content of these data structures are local matters.

The Program object provides a network-visible view of selected parameters of an application program in the form of properties of the Program object. Some of these properties are specified in the standard and exhibit a consistent behavior across different BACnet devices. The operating state of the process that executes the application program may be viewed and controlled through these standardized properties, which are required for all Program objects. In addition to these standardized properties, a Program object may also provide vendor-specific properties. These vendor-specific properties may serve as inputs to the program, outputs from the program, or both. However, these vendor-specific properties may not be present at all. If any vendor-specific properties are present, the standard does not define what they are or how they work, as this is specific to the particular application program and vendor.

Program objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Program objects that support intrinsic reporting shall apply the NONE event algorithm.

The Program object type and its standardized properties are summarized in Table 12-26 and described in detail in this clause.

Table 12-26. Properties of the Program Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Program_State	BACnetProgramState	R
Program_Change	BACnetProgramRequest	W
Reason_For_Halt	BACnetProgramError	O[1]
Description_Of_Halt	CharacterString	O[1]
Program_Location	CharacterString	O
Description	CharacterString	O
Instance_Of	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Event_Detection_Enable	BOOLEAN	O[2,3]
Notification_Class	Unsigned	O[2,3]
Event_Enable	BACnetEventTransitionBits	O[2,3]
Event_State	BACnetEventState	O[2,3]
Acked_Transitions	BACnetEventTransitionBits	O[2,3]
Notify_Type	BACnetNotifyType	O[2,3]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,3]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[3]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[3]
Reliability_Evaluation_Inhibit	BOOLEAN	O[4]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[5]
Auditable_Operations	BACnetAuditOperationFlags	O[5]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If one of the optional properties Reason_For_Halt or Description_Of_Halt is present, then both of these properties shall be present.

[2] These properties are required if the object supports intrinsic reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

[3] These properties shall be present only if the object supports intrinsic reporting.
[4] If this property is present, then the Reliability property shall be present.
[5] This property shall be present only if the device supports audit reporting.

12.22.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.22.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.22.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be PROGRAM.

12.22.4 Program_State

This property, of type BACnetProgramState, reflects the current logical state of the process executing the application program this object represents. This property is Read-Only. The values that may be taken on by this property are:

IDLE	process is not executing
LOADING	application program being loaded
RUNNING	process is currently executing
WAITING	process is waiting for some external event
HALTED	process is halted because of some error condition
UNLOADING	process has been requested to terminate

12.22.5 Program_Change

This property, of type BACnetProgramRequest, is used to request changes to the operating state of the process this object represents. The Program_Change property provides one means for changing the operating state of this process. The process may change its own state as a consequence of execution as well. The values that may be taken on by this property are:

READY	ready for change request (the normal state)
LOAD	request that the application program be loaded, if not already loaded
RUN	request that the process begin executing, if not already running
HALT	request that the process halt execution
RESTART	request that the process restart at its initialization point
UNLOAD	request that the process halt execution and unload

Normally the value of the Program_Change property will be READY, meaning that the program is ready to accept a new request to change its operating state. If the Program_Change property is not READY, then it may not be written to and any attempt to write to the property shall return a Result(-). If it has one of the other enumerated values, then a previous request to change state has not yet been honored, so new requests cannot be accepted. When the request to change state is finally honored, then the Program_Change property value shall become READY and the new state shall be reflected in the Program_State property. Depending on the current Program_State, certain requested values for Program_Change may be invalid and would also return a Result(-) if an attempt were made to write them. Figure 12-3 shows the valid state transitions and the resulting new Program_State.

It is important to note that program loading could be terminated either due to an error or a request to HALT that occurs during loading. In either case, it is possible to have Program_State=HALTED and yet not have a complete or operable program in place. In this case, a request to RESTART is taken to mean LOAD instead. If a complete program is loaded but HALTED for any reason, then RESTART simply reenters program execution at its initialization entry point.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

There may be BACnet devices that support Program objects but do not require "loading" of the application programs, as these applications may be built in. In these cases, loading is taken to mean "preparing for execution," the specifics of which are a local matter.

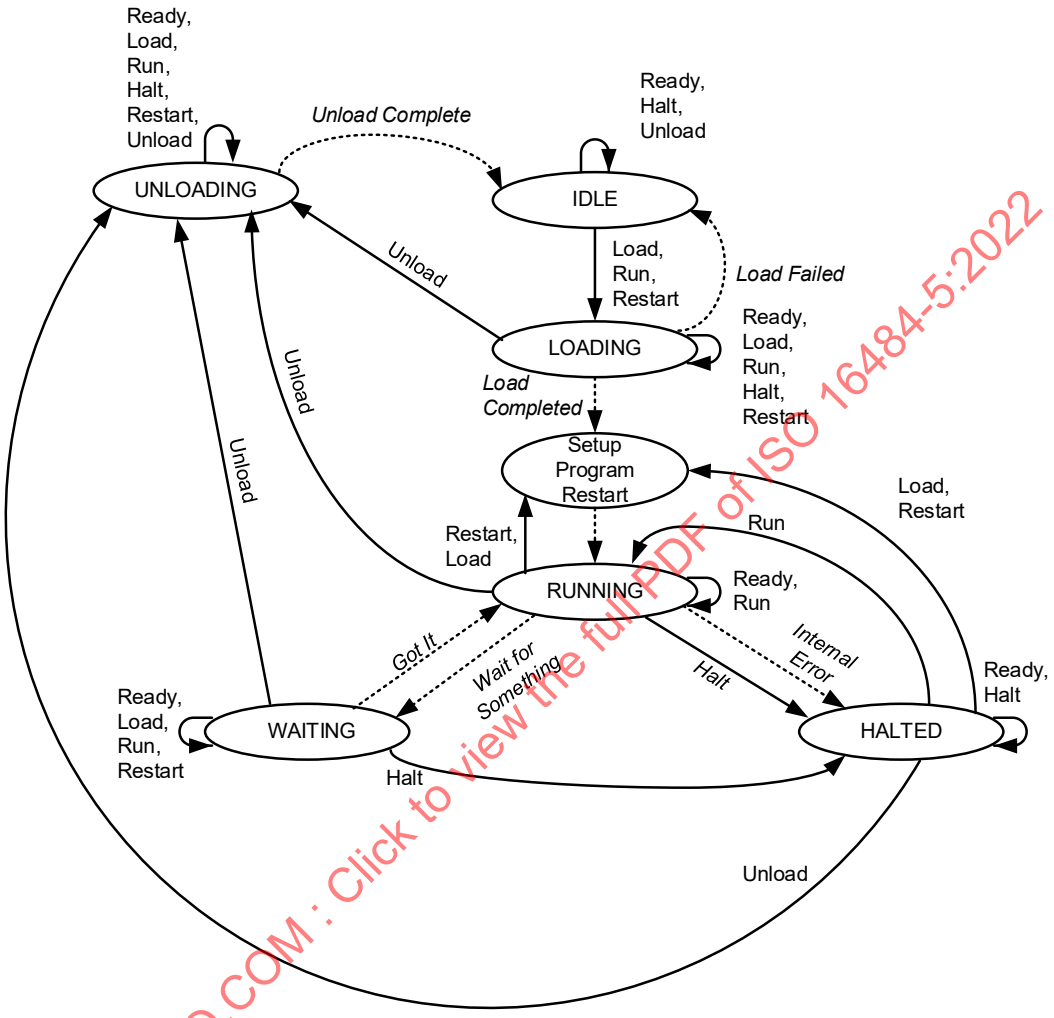


Figure 12-3. State Transitions for the program object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.22.6 Reason_For_Halt

If the process executing the application program this object represents encounters any type of error that causes process execution to be halted, then this property shall reflect the reason why the process was halted. The Reason_For_Halt property shall be an enumerated type called BACnetProgramError. The values that may be taken on by this property are:

NORMAL	process is not halted due to any error condition
LOAD_FAILED	the application program could not complete loading
INTERNAL	process is halted by some internal mechanism
PROGRAM	process is halted by Program_Change request
OTHER	process is halted for some other reason

If one of the optional properties Reason_For_Halt or Description_Of_Halt is present, then both of these properties shall be present.

© ISO 2022 – All rights reserved

12.22.7 Description_Of_Halt

This property, of type CharacterString, is a string of printable characters that may be used to describe the reason why a program has been halted. This property provides essentially the same information as the Reason_For_Halt property, except in a human-readable form. The content of this string is a local matter. If one of the optional properties Reason_For_Halt or Description_Of_Halt is present, then both of these properties shall be present.

12.22.8 Program_Location

This property, of type CharacterString, is a string of printable characters that may be used by the application program to indicate its location within the program code, for example, a line number or program label or section name. The content of this string is a local matter.

12.22.9 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.22.10 Instance_Of

This property, of type CharacterString, is a string of printable characters that is the local name of the application program being executed by this process. The content of this string is a local matter.

12.22.11 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the program. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the program has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that neither the Program_Change, Program_State nor any other program-specific property may be changed through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.22.12 Reliability

This property, of type BACnetReliability, provides an indication of whether the application-specific properties of the program object or the process executing the application program are "reliable" as far as the BACnet device can determine and, if not, why.

12.22.13 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the process this object represents is not in service. In this case, "in service" means that the application program is properly loaded and initialized, although the process may or may not be actually executing. If the Program_State property has the value IDLE, then Out_Of_Service shall be TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When Out_Of_Service is TRUE:

(a) the application program process is not executing;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the program;
(c) the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Reliability property shall respond to changes made to this property, as if those changes had occurred to the application program process.

12.22.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.22.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.22.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.22.17 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object.

12.22.18 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.22.19 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.22.20 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.22.21 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.22.22 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.22.23 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.22.24 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.22.25 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.22.26 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.22.27 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.22.28 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.22.29 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.23 Pulse Converter Object Type

The Pulse Converter object type defines a standardized object that represents a process whereby ongoing measurements made of some quantity, such as electric power or water or natural gas usage, and represented by pulses or counts, might be monitored over some time interval for applications such as peak load management, where it is necessary to make periodic measurements but where a precise accounting of every input pulse or count is not required.

The Pulse Converter object might represent a physical input. As an alternative, it might acquire the data from the Present_Value of an Accumulator object, representing an input in the same device as the Pulse Converter object. This linkage is illustrated by the dotted line in Figure 12-4. Every time the Present_Value property of the Accumulator object is incremented, the Count property of the Pulse Converter object is also incremented.

The Present_Value property of the Pulse Converter object can be adjusted at any time by writing to the Adjust_Value property, which causes the Count property to be adjusted, and the Present_Value recomputed from Count. In the illustration in Figure 12-4, the Count property of the Pulse Converter was adjusted down to 0 when the Total_Count of the Accumulator object had the value 0070.

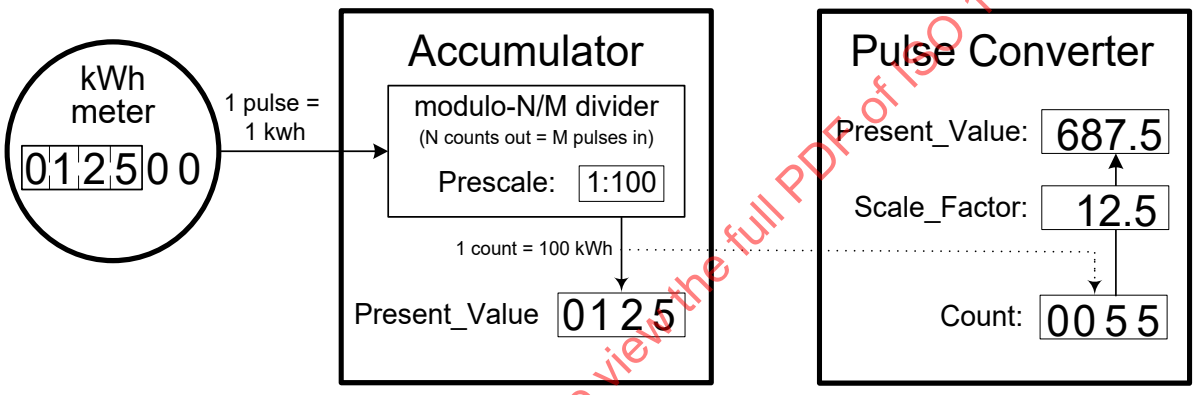


Figure 12-4. Relationship between the Pulse Converter and Accumulator objects.

Pulse Converter objects that support intrinsic reporting shall apply the OUT_OF_RANGE event algorithm.

The object and its properties are summarized in Table 12-27 and described in detail in this clause.

Table 12-27. Properties of the Pulse Converter Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	REAL	R[1]
Input_Reference	BACnetObjectPropertyReference	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Units	BACnetEngineeringUnits	R
Scale_Factor	REAL	R
Adjust_Value	REAL	W
Count	Unsigned	R
Update_Time	BACnetDateTime	R
Count_Change_Time	BACnetDateTime	R
Count_Before_Change	Unsigned	R
COV_Increment	REAL	O[2]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
COV_Period	Unsigned	O[2]
Notification_Class	Unsigned	O[3,5]
Time_Delay	Unsigned	O[3,5]
High_Limit	REAL	O[3,5]
Low_Limit	REAL	O[3,5]
Deadband	REAL	O[3,5]
Limit_Enable	BACnetLimitEnable	O[3,5]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[8]
Auditable_Operations	BACnetAuditOperationFlags	O[8]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.
[8] This property shall be present only if the device supports audit reporting.

12.23.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.23.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.23.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be PULSE_CONVERTER.

12.23.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.23.5 Present_Value

This property, of type REAL, indicates the accumulated value of the input being measured. It is computed by multiplying the current value of the Count property by the value of the Scale_Factor property. The value of the Present_Value

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

property may be adjusted by writing to the Adjust_Value property. The Present_Value property shall be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.6 Input_Reference

This property, of type BACnetObjectPropertyReference, indicates the object and property (typically an Accumulator object's Present_Value property) representing the actual physical input that is to be measured and presented by the Pulse Converter object. The referenced property should have a datatype of INTEGER or Unsigned.

If this property is not present, the Pulse Converter object directly represents the physical input.

12.23.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Pulse Converter. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the program has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value, Count and Reliability properties are no longer tracking changes to the input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.23.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value and/or Count properties or the operation of the physical input in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

If Input_Reference is configured to reference a property that is not of datatype Unsigned or INTEGER, or is otherwise not supported as an input source for this object, the Reliability property shall indicate CONFIGURATION_ERROR.

12.23.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the input that the object directly represents, if any, is not in service. ("Directly represents" means that the Input_Reference property is not present in this object.)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the Count property and will not track changes to the input;
(b) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the input;
(c) the Present_Value property and the Reliability property , if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties while Out_Of_Service is TRUE as if those changes had occurred in the input.

If the Input_Reference property is present, the state of the Out_Of_Service property of the object referenced by Input_Reference shall not be indicated by the Out_Of_Service property of the Pulse Converter object.

12.23.11 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.23.12 Scale_Factor

This property, of type REAL, provides the conversion factor for computing Present_Value. It represents the change in Present_Value resulting from changing the value of Count by one.

12.23.13 Adjust_Value

This property, of type REAL, is written to adjust the Present_Value property (and thus the Count property also) by the amount written to Adjust_Value.

The following series of operations shall be performed atomically when this property is written:

(1) The value written to Adjust_Value shall be stored in the Adjust_Value property.
(2) The value of Count shall be copied to the Count_Before_Change property.
(3) The value of Count shall be decremented by the value calculated by performing the integer division (Adjust_Value/Scale_Factor) and discarding the remainder.
(4) The current date and time shall be stored in the Count_Change_Time property.

A write to this property results in a change in the value of Present_Value. Whether the new value is computed as part of the atomic series of operations or when Present_Value is read is a local matter.

An attempt to write Adjust_Value with a value that would cause an overflow or underflow condition in Count shall result in a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If Adjust_Value has never been written, it shall have a value of zero.

12.23.14 Count

This read-only property, of type Unsigned, indicates the count of the input pulses as acquired from the physical input or the property referenced by the Input_Reference property.

If the property referenced by Input_Reference property is present, has datatype Unsigned or INTEGER, and is supported as an input source for this object, the value of the Count property is derived from the referenced property. An increment by one count in the referenced property is reflected by an increment of one count in the Count property. The means by which this is done shall be a local matter. Because the value of the Pulse Converter object Count property may be changed by a write to the Adjust_Value property, the value of the Count property can be different from the value of the referenced property.

12.23.15 Update_Time

This read-only property, of type BACnetDateTime, reflects the date and time of the most recent change to the Count property as a result of input pulse accumulation and is updated atomically with the Count property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.23.16 Count_Change_Time

This read-only property, of type BACnetDateTime, represents the date and time of the most recent occurrence of a write to the Adjust_Value property.

12.23.17 Count_Before_Change

This property, of type Unsigned, indicates the value of the Count property just prior to the most recent write to the Adjust_Value property. If no such write has yet occurred, this property shall have the value zero.

12.23.18 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.23.19 COV_Period

The COV_Period property, of type Unsigned, shall indicate the amount of time in seconds between the periodic COV notifications performed by this object. This property is required if COV reporting is supported by this object. See Clause 13.1.

12.23.20 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.23.21 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.22 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.23 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.24 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.25 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.23.26 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.23.27 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.23.28 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.23.29 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.23.30 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.23.31 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.23.32 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.23.33 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.23.34 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.23.35 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.23.36 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.23.37 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.23.38 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.23.39 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.23.40 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.23.41 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.23.42 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.24 Schedule Object Type

The Schedule object type defines a standardized object used to describe a periodic schedule that may recur during a range of dates, with optional exceptions at arbitrary times on arbitrary dates. The Schedule object also serves as a binding between these scheduled times and the writing of specified "values" to specific properties of specific objects at those times. The Schedule object type and its properties are summarized in Table 12-28 and described in detail in this clause.

Schedules are divided into days, of which there are two types: normal days within a week and exception days. Both types of days can specify scheduling events for either the full day or portions of a day, and a priority mechanism defines which scheduled event is in control at any given time.

The current state of the Schedule object is represented by the value of its Present_Value property, which is normally calculated using the time/value pairs from the Weekly_Schedule and Exception_Schedule properties, with a default value for use when no schedules are in effect. Details of this calculation are provided in the description of the Present_Value property.

Versions of the Schedule object prior to Protocol_Revision 4 only support schedules that define an entire day, from midnight to midnight. For compatibility with these versions, this whole day behavior can be achieved by using a specific schedule format. Weekly_Schedule and Exception_Schedule values that begin at 00:00, and do not use any NULL values, will define schedules for the entire day. Property values in this format will produce the same results in all versions of the Schedule object.

Schedule objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Schedule objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-28. Properties of the Schedule Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Any	R
Description	CharacterString	O
Effective_Period	BACnetDateRange	R
Weekly_Schedule	BACnetARRAY[7] of BACnetDailySchedule	O[1]
Exception_Schedule	BACnetARRAY[N] of BACnetSpecialEvent	O[1]
Schedule_Default	Any	R
List_Of_Object_Property_References	BACnetLIST of BACnetDeviceObjectPropertyReference	R
Priority_For_Writing	Unsigned(1..16)	R
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Event_Detection_Enable	BOOLEAN	O[2,3]
Notification_Class	Unsigned	O[2,3]
Event_Enable	BACnetEventTransitionBits	O[2,3]
Event_State	BACnetEventState	O[2,3]
Acked_Transitions	BACnetEventTransitionBits	O[2,3]
Notify_Type	BACnetNotifyType	O[2,3]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,3]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[3]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[3]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[4]
Auditable_Operations	BACnetAuditOperationFlags	O[4]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[1] At least one of these properties is required.
[2] These properties are required if the object supports intrinsic reporting.
[3] These properties shall be present only if the object supports intrinsic reporting.
[4] This property shall be present only if the device supports audit reporting.

12.24.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.24.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.24.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be SCHEDULE.

12.24.4 Present_Value

This property indicates the current value of the schedule, which may be any primitive datatype. As a result, most analog, binary, and enumerated values may be scheduled. This property shall be writable when Out_Of_Service is TRUE (see Clause 12.24.14).

Any change in the value of this property shall be written to all members of the List_Of_Object_Property_References property. An error writing to any member of the list shall not stop the Schedule object from writing to the remaining members.

The normal calculation of the value of the Present_Value property is illustrated as follows (the actual algorithm used is a local matter but shall yield the same results as this one):

(1) Find the highest relative priority (as defined by Clause 12.24.8) Exception_Schedule array element that is in effect for the current day and whose current value (see method below) is not NULL, and assign that value to the Present_Value property.
(2) If the Present_Value was not assigned in the previous step, then evaluate the current value of the Weekly_Schedule array element for the current day and if that value is not NULL, assign it to the Present_Value property.
(3) If the Present_Value was not assigned in the previous steps, then assign the value of the Schedule_Default property to the Present_Value property.

The method for evaluating the current value of a schedule (either exception or weekly) is to find the latest element in the list of BACnetTimeValues that occurs on or before the current time, and then use that element's value as the current value for the schedule. If no such element is found, then the current value for the schedule shall be NULL.

These calculations are such that they can be performed at any time and the correct value of Present_Value property will result. These calculations shall be performed at 00:00 each day, whenever the device resets, whenever properties that can affect the results are changed, whenever the time in the device changes by an amount that may have an effect on the calculation result, and at other times, as required, to maintain the correct value of the Present_Value property through the normal passage of time.

Note that the Present_Value property will be assigned the value of the Schedule_Default property at 00:00 of any given day, unless there is an entry for 00:00 in effect for that day. If a scheduled event logically begins on one day and ends on another, an entry at 00:00 shall be placed in the schedule that is in effect for the second day, and for any subsequent days of the event's duration, to ensure the correct result whenever Present_Value is calculated.

12.24.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.24.6 Effective_Period

This property specifies the range of dates within which the Schedule object is active. Seasonal scheduling may be achieved by defining several SCHEDULE objects with non-overlapping Effective_Periods to control the same property references. Upon entering its effective period, the object shall calculate its Present_Value and write that value to all members of the List_Of_Object_Property_References property. An error writing to any member of the list shall not stop the Schedule object from writing to the remaining members.

12.24.7 Weekly_Schedule

This property, of type BACnetARRAY[7] of BACnetDailySchedule, contains exactly 7 elements each of which contain a BACnetDailySchedule. A BACnetDailySchedule consists of a list of BACnetTimeValues that are (time, value) pairs, which describe the sequence of schedule actions on one day of the week when no Exception_Schedule is in effect. The time portion of each BACnetTimeValue shall contain a specific time. The array elements 1-7 correspond to the days Monday - Sunday, respectively. The Weekly_Schedule is an optional property, but either the Weekly_Schedule or a non-empty Exception_Schedule shall be supported in every instance of a Schedule object.

If the Weekly_Schedule property is written with a schedule item containing a datatype not supported by this instance of the Schedule object (e.g., the List_Of_Object_Property_References property cannot be configured to reference a property of the unsupported datatype), the device may return a Result(-) response specifying an 'Error Class' of PROPERTY and an 'Error Code' of DATATYPE_NOT_SUPPORTED.

If the Weekly_Schedule property is written with a value containing a duplicate time within any single day, the device shall return a Result(-) response specifying an 'Error Class' of PROPERTY and an 'Error Code' of DUPLICATE_ENTRY.

12.24.8 Exception_Schedule

This property, of type BACnetARRAY of BACnetSpecialEvents, describes a sequence of schedule actions that takes precedence over the normal day's behavior on a specific day or days.

BACnetSpecialEvent ::= (Period, list of BACnetTimeValue, EventPriority)

Period ::= Choice of {BACnetCalendarEntry | CalendarReference}

EventPriority ::= Unsigned (1..16)

The Period may be a BACnetCalendarEntry or it may refer to a Calendar object. A BACnetCalendarEntry would be used if the BACnetSpecialEvent is specific to this Schedule object, while Calendar objects might be defined for common holidays to be referenced by multiple Schedule objects. Each BACnetCalendarEntry is either a specific date or date pattern (Date), range of dates (BACnetDateRange), or month/week-of-month/day-of-week specification (BACnetWeekNDay). If the current date matches any of the calendar entry criteria, the BACnetSpecialEvent would be activated and the list of BACnetTimeValue items would be enabled for use.

For the calendar entry criteria to match, each of the octets in a date pattern or BACnetWeekNDay are evaluated independently and all specified criteria must match.

As an example, if the calendar entry is a BACnetWeekNDay with an unspecified octet for month and week-of-month fields but with a specific day-of-week, it means the BACnetSpecialEvent applies on that day-of-week all year long.

Each item in the list of BACnetTimeValue specifies a time and a value of primitive datatype that is used at the time specified by the time portion. The time portion shall contain a specific time.

Each BACnetSpecialEvent contains an EventPriority that determines its importance relative to other BACnetSpecialEvent elements within the same Exception_Schedule array. Since BACnetSpecialEvent elements within the same Exception_Schedule array may have overlapping periods, it is necessary to have a mechanism to determine the relative priorities for the BACnetSpecialEvent elements that apply on any given day. If more than one BACnetSpecialEvent applies to a given day, the relative priority of the BACnetSpecialEvent elements shall be determined by their EventPriority values. If multiple overlapping BACnetSpecialEvent elements have the same EventPriority value, then the BACnetSpecialEvent with the lowest index number in the array shall have higher relative priority. The highest

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

EventPriority is 1 and the lowest is 16. The EventPriority is not related to the Priority_For_Writing property of the Schedule object.

If a BACnet device supports writing to the Exception_Schedule property, all possible choices in the BACnetSpecialEvent elements shall be supported. If the size of this array is increased by writing to array index zero, each newly created array element shall be a BACnetSpecialEvent containing an empty list of BACnetTimeValue.

If the Exception_Schedule property is written with a schedule item containing a datatype not supported by this instance of the Schedule object (e.g., the List_Of_Object_Property_References property cannot be configured to reference a property of the unsupported datatype), the device may return a Result(-) response, specifying an 'Error Class' of PROPERTY and an 'Error Code' of DATATYPE_NOT_SUPPORTED.

If the Exception_Schedule property is written with a BACnetSpecialEvent containing a duplicate time in any particular SEQUENCE OF TimeValue, the device shall return a Result(-) response specifying an 'Error Class' of PROPERTY and an 'Error Code' of DUPLICATE_ENTRY.

12.24.9 Schedule_Default

This property holds a default value to be used for the Present_Value property when no other scheduled value is in effect (see Clause 12.24.4). This may be any primitive datatype.

If the Schedule_Default property is written with a value containing a datatype not supported by this instance of the Schedule object (e.g., the List_Of_Object_Property_References property cannot be configured to reference a property of the unsupported datatype), the device may return a Result(-) response specifying an 'Error Class' of PROPERTY and an 'Error Code' of DATATYPE_NOT_SUPPORTED.

12.24.10 List_Of_Object_Property_References

This property specifies the Device Identifiers, Object Identifiers and Property Identifiers of the properties to be written with specific values at specific times on specific days.

If this property is writable, it may be restricted to only support references to objects inside of the device containing the Schedule object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property is set to reference an object outside the device containing the Schedule object, the method used for writing to the referenced property value for the purpose of controlling the property is a local matter. The only restriction on the method of writing to the referenced property is that the scheduling device be capable of using WriteProperty for this purpose so as to be interoperable with all BACnet devices.

12.24.11 Priority_For_Writing

This property, of type Unsigned, defines the priority at which the referenced properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. The value 1 is considered the highest priority and 16 the lowest. See Clause 19.2.

12.24.12 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the schedule object. Two of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the schedule object has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.24.13 Reliability

This property, of type BACnetReliability, provides an indication that the properties of the schedule object are in a consistent state. All non-NULL values used in the Weekly_Schedule, the Exception_Schedule, and the Schedule_Default properties shall be of the same datatype, and all members of the List_Of_Object_Property_References shall be writable with that datatype. If these conditions are not met, then this property shall have the value CONFIGURATION_ERROR.

If the List_Of_Object_Property_References contains a member that references a property in a remote device, the detection of a configuration error may be delayed until an attempt is made to write a scheduled value.

12.24.14 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the internal calculations of the schedule object are used to determine the value of the Present_Value property.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the internal calculations and will not track changes to other properties;
(b) other functions that depend on the state of the Present_Value, such as writing to the members of the List_Of_Object_Property_References, shall respond to changes made to that property, as if those changes had occurred by internal calculations.

12.24.15 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.24.16 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.24.17 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.24.18 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.24.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.24.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.24.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.24.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.24.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.24.24 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.24.25 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.24.26 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.24.27 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.24.28 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.24.29 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.24.30 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.25 Trend Log Object Type

A Trend Log object monitors a property of a referenced object and, when predefined conditions are met, saves ("logs") the value of the property and a timestamp in an internal log buffer for subsequent retrieval. The data may be logged periodically, upon a change of value or when "triggered" by a write to the Trigger property. The Trigger property allows the acquisition of samples to be controlled by network write operations or internal processes. Errors that prevent the acquisition of the data, as well as changes in the status or operation of the logging process itself, are also recorded. Each timestamped buffer entry is called a "log record."

The referenced object may reside in the same device as the Trend Log object or in an external device. The referenced property's value may be recorded upon COV subscription or periodic poll. If the value of the monitored object's Status_Flags property is available, then it may optionally be recorded along with the value of the referenced property.

Each Trend Log object maintains an internal, optionally fixed-size log buffer. This log buffer fills or grows as log records are added. If the log buffer becomes full, the least recent log record is overwritten when a new log record is added, or collection may be set to stop. Log buffers are transferred as a list of BACnetLogRecord using the ReadRange service. The log buffer may be cleared by writing a zero to the Record_Count property. Each log record in the log buffer has an implied SequenceNumber which is equal to the value of the Total_Record_Count property immediately after the log record is added.

Several datatypes are defined for storage in the log records. Data stored in the log buffer may be optionally restricted in size to 32 bits, as in the case of bit strings, to facilitate implementation in devices with strict storage requirements.

Logging may be enabled and disabled through the Enable property and at dates and times specified by the Start_Time and Stop_Time properties. Trend Log enabling and disabling is recorded in the log buffer.

Event reporting (notification) may be provided to facilitate automatic fetching of log records by processes on other devices such as fileservers. Support is provided for algorithmic reporting; optionally, intrinsic reporting may be provided. Trend Log objects that support intrinsic reporting shall apply the BUFFER_READY event algorithm.

In intrinsic reporting, when the number of log records specified by the Notification_Threshold property have been collected since the previous notification (or startup), a new notification is sent to recipients enrolled in the respective Notification Class object.

In response to a notification, recipients may fetch all of the new records. If a recipient needs to fetch all of the new log records, it should use the 'By Sequence Number' form of the ReadRange service request.

A missed notification may be detected by a recipient if the 'Current Notification' parameter received in the previous BUFFER_READY notification is different than the 'Previous Notification' parameter of the current BUFFER_READY notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has the FIRST_ITEM bit of the 'Result Flags' parameter set to TRUE, log records have probably been missed by this recipient.

The acquisition of log records by remote devices has no effect upon the state of the Trend Log object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its records at any time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Table 12-29. Properties of the Trend Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Enable	BOOLEAN	W
Start_Time	BACnetDateTime	O[1,2]
Stop_Time	BACnetDateTime	O[1,2]
Log_DeviceObjectProperty	BACnetDeviceObjectPropertyReference	O[8]
Log_Interval	Unsigned	O[1,3]
COV_Resubscription_Interval	Unsigned	O

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Client_COV_Increment	BACnetClientCOV	O
Stop_When_Full	BOOLEAN	R
Buffer_Size	Unsigned32	R
Log_Buffer	BACnetLIST of BACnetLogRecord	R
Record_Count	Unsigned32	W
Total_Record_Count	Unsigned32	R
Logging_Type	BACnetLoggingType	R
Align_Intervals	BOOLEAN	O[5]
Interval_Offset	Unsigned	O[5]
Trigger	BOOLEAN	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O
Notification_Threshold	Unsigned32	O[4,7]
Records_Since_Notification	Unsigned32	O[4,7]
Last_Notify_Record	Unsigned32	O[4,7]
Event_State	BACnetEventState	R
Notification_Class	Unsigned	O[4,7]
Event_Enable	BACnetEventTransitionBits	O[4,7]
Acked_Transitions	BACnetEventTransitionBits	O[4,7]
Notify_Type	BACnetNotifyType	O[4,7]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,7]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[7]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[7]
Event_Detection_Enable	BOOLEAN	O[4,7]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[7]
Event_Algorithm_Inhibit	BOOLEAN	O[7,9]
Reliability_Evaluation_Inhibit	BOOLEAN	O[10]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the monitored property is a BACnet property.
[2] If present, these properties are required to be writable.
[3] If present, this property is required to be writable when Logging_Type has the value POLLED or the value COV. Also, if present this property is required to be read-only if Logging_Type has the value TRIGGERED.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties are required if, and shall be present only if, the object supports clock-aligned logging.
[6] Footnote removed.
[7] These properties shall be present only if the object supports intrinsic reporting.
[8] This property is required if, and shall be present only if, the monitored property is a BACnet property.
[9] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[10] If this property is present, then the Reliability property shall be present.
[11] This property shall be present only if the device supports audit reporting.

12.25.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.25.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.25.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TREND_LOG.

12.25.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.25.5 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains an unspecified datetime, then it shall be considered equal to 'the start of time'. If Stop_Time contains an unspecified datetime, then it shall be considered equal to 'the end of time'. As an exception, log records of type log-status in which the BUFFER_PURGED flag is set to TRUE shall always be recorded.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and either Record_Count is equal to Buffer_Size, or Buffer_Size is 2^{32}-1 and there is no space for another record, shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

12.25.6 Start_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. This property shall be writable if present.

When Start_Time is reached, the value of the Enable property is not changed.

12.25.7 Stop_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be disabled by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. This property shall be writable if present.

When Stop_Time is reached, the value of the Enable property is not changed.

12.25.8 Log_DeviceObjectProperty

This property, of type BACnetDeviceObjectPropertyReference, specifies the Device Identifier, Object Identifier and Property Identifier of the property to be trend logged.

A change of the value of this property shall cause all records in the log buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported as a BUFFER_PURGED log-status log record in the log buffer as the initial entry regardless of the state of the log buffer before the change.

If this property is writable, it may be restricted to reference only objects inside the device containing the Trend Log object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned.

12.25.9 Log_Interval

This property, of type Unsigned, specifies the periodic interval in hundredths of seconds for which the referenced property is to be logged when Logging_Type has the value POLLED. If the Logging_Type property has either of the values COV or TRIGGERED, then the value of the Log_Interval property shall be zero and ignored.

To maintain compatibility with previous versions of the standard, devices supporting COV data collection shall support switching between COV and polled modes in response to writes to Log_Interval. If the Logging_Type property has the value POLLED, changing the Log_Interval property from a non-zero value to the value zero shall change the value of Logging_Type to COV, and shall cause the Trend Log object to issue COV subscriptions for the referenced property. If

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

the Logging_Type property has the value COV, writing a non-zero value to the Log_Interval property shall change the value of Logging_Type to POLLED, and shall cause the Trend Log object to periodically poll the monitored property.

If present, this property shall be writable if Logging_Type has either the value POLLED or the value COV. This property shall be read-only if Logging_Type has the value TRIGGERED.

12.25.10 COV_Resubscription_Interval

If the Trend Log is acquiring data from a remote device by COV subscription, this property, of type Unsigned, specifies the number of seconds between COV resubscriptions, provided that COV subscription is in effect. COV subscription requests shall specify twice this lifetime for the subscription and shall specify the issuance of confirmed notifications. If COV subscriptions are in effect, the first COV subscription is issued when the Trend Log object begins operation or when Enable becomes TRUE. If present, the value of this property shall be non-zero.

A device may use a single COV subscription for multiple Trend Log objects only if they reference the same remote property, the resubscription interval is equal, and the COV increment is equal in the respective Trend Log objects. Otherwise, individual COV subscriptions shall be maintained for each Trend Log object.

If this property is not present, then COV subscription shall not be attempted.

12.25.11 Client_COV_Increment

If the Trend Log is acquiring COV data, this property, of type BACnetClientCOV, specifies the increment to be used in determining that a change of value has occurred. If the referenced object and property supports COV reporting according to Clause 13.1, this property may have the value NULL; in this case change of value is determined by the criteria of Clause 13.1.

12.25.12 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the log buffer is full. When logging ceases because the addition of one more log record would cause the log buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest log record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

If Stop_When_Full is TRUE and Buffer_Size is 2^{32}-1 and the addition of a new record fails due to a lack of space, then the oldest Log_Buffer record shall be discarded, the Enable property shall be set to FALSE and the event is recorded.

12.25.13 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter. If all records are deleted when the Buffer_Size is written then the object shall act as if the Record_Count was set to zero.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

12.25.14 Log_Buffer

This property, of type BACnetLIST of BACnetLogRecord, is a list of up to Buffer_Size timestamped log records of datatype BACnetLogRecord, each of which conveys a recorded data value, an error related to data-collection, or status changes in the Trend Log object. Each log record has data fields as follows:

Timestamp When Logging_Type is POLLED or TRIGGERED the timestamp shall be the date and time the logging object attempts to acquire the record. When Logging_Type is COV, the timestamp shall be the date and time included within the notification if supplied, otherwise the date and time when the notification was received.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

LogDatum The data value read from the monitored object and property, an error encountered in an attempt to read a value, or a change in status or operation of the Trend Log object itself.

StatusFlags If this field is present in the log record, then it shall contain the value of the Status_Flags property of the monitored object. If the monitored object is in a different device than the Trend Log object, then it is recommended that the Status_Flags and the data value in the monitored object property be acquired together with a single service request, such as ReadPropertyMultiple or one of the COV notification services.

The choices available for the LogDatum are listed below:

log-status This choice represents a change in the status or operation of the Trend Log object. Whenever one of the events represented by the flags listed below occurs, a log record shall be appended to the log buffer.

- LOG_DISABLED This flag is changed whenever collection of log records by the Trend Log object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this log record will cause the log buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.
- BUFFER_PURGED This flag shall be set to TRUE whenever the buffer is cleared by writing zero to the Record_Count property or by a change to the Log_DeviceObjectProperty property. After this value is recorded in the log buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the log buffer shall be placed into the log buffer even if logging is disabled, the local time is outside of the time range defined by the Start_Time and Stop_Time properties, or the log buffer was completely empty before the request for clearing.
- LOG_INTERRUPTED This flag indicates that the collection of log records by the Trend Log object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that samples prior to this log record might have been missed.

boolean-value
real-value
enumerated-value
unsigned-value
integer-value
bitstring-value
null-value
These choices represent the data values and datatypes read from the monitored object and property.

failure This choice represents an error encountered in an attempt to read a data value from the monitored object. If the error is conveyed by an error response from a remote device the Error Class and Error Code in the response shall be recorded.

time-change This choice represents a change in the clock setting in the device; it records the number of seconds and fraction of a second by which the clock changed. If, and only if, the number is not known, such as when the clock is initialized for the first time, the value recorded shall be 0.0. This log record shall be recorded after changing the local time of the device and the timestamp shall reflect the new local time of the device.

any-value This choice represents the data values and datatypes read from the monitored object and property that are not specifically listed in LogDatum.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Also associated with each log record is an implied log record number, the value of which is equal to Total_Record_Count at the point where the log record has been added into the log buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the Trend Log object is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one.

The log buffer is not network accessible except through the use of the ReadRange service, in order to avoid problems with record sequencing when segmentation is required.

If the object supports event reporting, then a reference to this property shall be the pLogBuffer parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.25.15 Record_Count

This property, of type Unsigned32, shall represent the number of log records currently resident in the log buffer. A write of the value zero to this property shall cause all log records in the log buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported as a BUFFER_PURGED log-status log record in the log buffer as the initial entry regardless of the state of the log buffer before the write.

12.25.16 Total_Record_Count

This property, of type Unsigned32, shall represent the total number of log records collected by the Trend Log object since creation. When the value of Total_Record_Count reaches its maximum possible value of 2^{32} - 1, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost but its use in generating notifications remains.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.25.17 Notification_Threshold

This property, of type Unsigned32, is the pThreshold parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.25.18 Records_Since_Notification

This property, of type Unsigned32, represents the number of log records collected since the previous notification, or since the beginning of logging if no previous notification has occurred.

12.25.19 Last_Notify_Record

This property, of type Unsigned32, is the pPreviousCount parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.25.20 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.25.21 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.25.22 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.25.23 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.25.24 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.25.25 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.25.26 Logging_Type

This property, of type BACnetLoggingType, specifies whether the Trend Log object collects log records using polling, COV or triggered acquisition.

If this property is writable, an attempt to write a value not supported by the object into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property has the value COV, then the Trend Log shall issue COV subscriptions for the referenced property, and shall log the COV notifications if they indicate a changed value.

If this property has the value POLLED, then the Trend Log shall periodically poll the monitored property on the interval defined by the Log_Interval, Align_Intervals, and Interval_Offset properties.

If the value POLLED is written to this property when the value of Log_Interval is zero, then the Log_Interval property shall be updated with an appropriate default polling interval. Determination of the appropriate default polling interval is a local matter.

If either of the values COV or TRIGGERED is written to this property when Log_Interval has a non-zero value, then the Log_Interval property shall be updated with the value zero.

12.25.27 Align_Intervals

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic logging is enabled. If clock-aligned periodic logging is enabled and the value of Log_Interval is a factor of (i.e. it divides into without a remainder) a second, minute, hour or day, then the beginning of the period specified for logging shall be aligned to the second, minute, hour or day, respectively.

This property has no effect on the behavior of the Trend Log object if the Logging_Type property has a value other than POLLED.

12.25.28 Interval_Offset

This property, of type Unsigned, specifies the offset in hundredths of seconds from the beginning of the period specified for logging until the actual acquisition of a log record begins. The offset used shall be the value of Interval_Offset modulo the value of Log_Interval; i.e. if Interval_Offset has the value 31 and Log_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals is FALSE.

12.25.29 Trigger

This property, of type BOOLEAN, shall cause the Trend Log object to acquire a log record whenever the value of this property is changed from FALSE to TRUE. It shall remain TRUE while the Trend Log object is acquiring the data items

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

for a log record. When all data items have been collected or it has been determined that all outstanding data requests will not be fulfilled, the Trend Log object shall reset the value to FALSE.

If the value of the Logging_Type property is not TRIGGERED and an attempt is made to write the value TRUE to the Trigger property, it is left as a local matter whether to execute the logging operation or not. For devices that choose not to execute triggered logging when Logging_Type is not equal to TRIGGERED, attempts to write the value TRUE to this property when Logging_Type has a value other than TRIGGERED shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NOT_CONFIGURED_FOR_TRIGGERED_LOGGING.

Writing to the Trigger property is not restricted to network-visible write operations; internal processes may control the acquisition of samples by writing to this property.

12.25.30 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Trend Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT — Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN — The value of this flag shall be Logical FALSE (0).

OUT_OF_SERVICE — The value of this flag shall be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.25.31 Reliability

This property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet device can determine.

12.25.32 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.25.33 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.25.34 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.25.35 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.25.36 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.25.37 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.25.38 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.25.39 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.25.40 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.25.41 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.25.42 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.25.43 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.26 Access Door Object Type

The Access Door object type is an abstract interface to a physical door whose properties represent the externally visible characteristics of an access control door. The Access Door is comprised of a collection of physical door hardware, such as a door lock, a door contact, and a Request-To-Exit device, which together comprise a door for access control. The individual hardware components of the door may or may not be exposed through this object.

Access Door objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

For reliability-evaluation, the FAULT_STATE fault algorithm can be applied.

The object and its properties are summarized in Table 12-30 and described in detail in this clause.

Table 12-30. Properties of the Access Door Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetDoorValue	W
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	BACnetDoorValue	R
Door_Status	BACnetDoorStatus	O[1,2]
Lock_Status	BACnetLockStatus	O[1]
Secured_Status	BACnetDoorSecuredStatus	O
Door_Members	BACnetARRAY[N] of BACnetDeviceObjectReference	O
Door_Pulse_Time	Unsigned	R
Door_Extended_Pulse_Time	Unsigned	R
Door_Unlock_Delay_Time	Unsigned	O
Door_Open_Too_Long_Time	Unsigned	R
Door_Alarm_State	BACnetDoorAlarmState	O[1, 3]
Masked_Alarm_Values	BACnetLIST of BACnetDoorAlarmState	O
Maintenance_Required	BACnetMaintenance	O
Time_Delay	Unsigned	O[3,5]
Notification_Class	Unsigned	O[3,5]
Alarm_Values	BACnetLIST of BACnetDoorAlarmState	O[3,5]
Fault_Values	BACnetLIST of BACnetDoorAlarmState	O
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[7,9,11]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[8,10]
Last_Command_Time	BACnetTimeStamp	O[8,10]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[10]
Audit_Level	BACnetAuditLevel	O[12]
Auditable_Operations	BACnetAuditOperationFlags	O[12]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[12]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties, when present, shall be writable when Out_Of_Service is TRUE.
[2] This property is required if the property Secured_Status is present.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] This property is required if the object supports the value source mechanism.
[8] These properties are required if the object supports the value source mechanism and is commandable.
[9] This property shall be present only if the object supports the value source mechanism.
[10] These properties shall be present only if the object supports the value source mechanism and is commandable.
[11] This property shall be writable as described in Clause 19.5.
[12] This property shall be present only if the device supports audit reporting.

12.26.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.26.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.26.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_DOOR.

12.26.4 Present_Value (Commandable)

This property, of type BACnetDoorValue, reflects the current active command of the access door object. The Present_Value is commandable and has one of the following values:

LOCK — The door is commanded to the locked state.

UNLOCK — The door is commanded to the unlocked state.

PULSE_UNLOCK — The door will be commanded to the unlocked state for a maximum of the time specified by Door_Pulse_Time, after which the value will be automatically relinquished from the priority array at the commanded priority. It is permissible for the local controller to relinquish the value from the priority array before the time specified by Door_Pulse_Time has expired. The conditions when this may occur are considered a local matter.

If an unlock delay is in effect when the value of PULSE_UNLOCK is written at the given priority, then the door shall remain in the locked state for Door_Unlock_Delay_Time tenths of seconds before it is commanded to the unlocked state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a value of PULSE_UNLOCK is written at a given priority and the Present_Value is currently being commanded, at any value, at a higher priority then the lower priority value will be relinquished immediately.

EXTENDED_PULSE_UNLOCK The door will be commanded to the unlocked state for a maximum of the time specified by Door_Extended_Pulse_Time, after which the value will be automatically relinquished from the priority array at the commanded priority. It is permissible for the local controller to relinquish the value from the priority array before the time specified by Door_Extended_Pulse_Time has expired. The conditions when this may occur are considered a local matter.

If an unlock delay is in effect when the value of EXTENDED_PULSE_UNLOCK is written at the given priority, then the door shall remain in the locked state for Door_Unlock_Delay_Time tenths of seconds before it is commanded to the unlocked state.

If a value of EXTENDED_PULSE_UNLOCK is written at a given priority and the Present_Value is currently being commanded, at any value, at a higher priority then the lower priority value will be relinquished immediately.

Note that the present value represents the commanded state of the door, which does not necessarily correspond to the physical state of the door lock.

The present value of the Access Door is defined for a standard access controlled door, where the control operation is to lock or unlock. However, this does not exclude motorized devices such as sliding doors, parking gates, etc., where the operation is to open or close. In these cases, locked shall be equivalent to closed and unlocked shall be equivalent to open.

12.26.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.26.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the physical door. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

Where:

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN Logical TRUE (1) if the object has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the physical door is no longer tracking changes to the Present_Value property and the Reliability property is no longer a reflection of the reliability of the physical inputs(s) and output(s). Otherwise, the value is logical FALSE (0).

OUT_OF_SERVICE Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.26.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.26.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical inputs or outputs which comprise this door are "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.26.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the logical door which this object represents is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled from the physical door and will not track changes to the physical door;
(b) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical door;
(c) the Present_Value property and the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, and if present, the Door_Status, Lock_Status and Door_Alarm_State properties shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties, and if present the Door_Status, Lock_Status and Door_Alarm_State properties, shall respond to changes made to these properties, as if those changes had occurred to the physical door.

12.26.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.26.11 Relinquish_Default

This property, of type BACnetDoorValue, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2. The acceptable values for this property are either LOCK or UNLOCK and the property shall not take on either of the values PULSE_UNLOCK or EXTENDED_PULSE_UNLOCK.

12.26.12 Door_Status

This property, of type BACnetDoorStatus, represents the open or closed state of the physical door. The values that may be taken on by this property are:

CLOSED	The door is closed.
OPENED	The door is open or partially open.
UNKNOWN	It is unknown whether the door is opened or closed.
DOOR_FAULT	The door status input associated with the physical door is unreliable.
UNUSED	There is no door status input associated with the door.

This property, if present, is required to be writable when Out_Of_Service is TRUE.

12.26.13 Lock_Status

This property, of type BACnetLockStatus, represents the monitored (as opposed to the commanded) status of the door lock. The values that may be taken on by this property are:

LOCKED	The door lock is locked.
UNLOCKED	The door lock is unlocked.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

UNKNOWN	It is unknown whether the door lock is locked or unlocked.
LOCK_FAULT	The lock status input associated with the door lock is unreliable.
UNUSED	There is no lock status input associated with the door.

This property, if present, is required to be writable when Out_Of_Service is TRUE.

12.26.14 Secured_Status

This property, of type BACnetDoorSecuredStatus, represents whether or not the physical door is in a secured state. This property shall have a value of SECURED if, and only if, all of the following conditions are met:

(a) the IN_ALARM flag of the Status_Flags property is FALSE, and
(b) the Masked_Alarm_Values list, if it exists, is empty , and
(c) the Door_Status property has a value of CLOSED or UNUSED, and
(d) the Present_Value property has a value of LOCK, and
(e) the Lock_Status property, if it exists, has a value of LOCKED or UNUSED.

If one or more of the previous conditions are not met, the property shall have a value of UNSECURED. If the device cannot determine any of the previous conditions, then the property shall have a value of UNKNOWN.

12.26.15 Door_Members

This property, of type BACnetARRAY of BACnetDeviceObjectReference holds an array of references to BACnet objects which represent I/O devices, authentication devices, schedules, programs, or other objects that are associated with the physical door. It is a local matter as to how this array is used and which objects are referenced in this array. The array may be empty or not present if the vendor does not wish to expose the individual objects that make up this physical door.

12.26.16 Door_Pulse_Time

This property, of type Unsigned, is the maximum duration of time, in tenths of seconds, for which the door will be unlocked when the Present_Value has a value of PULSE_UNLOCK, after which time the Present_Value shall be automatically relinquished at the priority that established the PULSE_UNLOCK command.

12.26.17 Door_Extended_Pulse_Time

This property, of type Unsigned, is the maximum amount of time, in tenths of seconds, which the door will be unlocked when the Present_Value has a value of EXTENDED_PULSE_UNLOCK, after which time the Present_Value shall be automatically relinquished at the priority that established the EXTENDED_PULSE_UNLOCK command.

12.26.18 Door_Unlock_Delay_Time

This property, of type Unsigned, is the duration of time, in tenths of seconds, which the physical door lock will delay unlocking when the Present_Value changes to a value of PULSE_UNLOCK or EXTENDED_PULSE_UNLOCK.

12.26.19 Door_Open_Too_Long_Time

This property, of type Unsigned, is the time, in tenths of seconds, to delay before setting the Door_Alarm_State to DOOR_OPEN_TOO_LONG after it is determined that a door-open-too-long condition exists. A door-open-too-long condition occurs when the Present_Value has a value of LOCK and one of the following conditions exist:

(a) The Present_Value had a previous value of PULSE_UNLOCK and the door has been in a continual open state for the time specified by Door_Open_Too_Long_Time after the Door_Pulse_Time has expired.
(b) The Present_Value had a previous value of EXTENDED_PULSE_UNLOCK and the door has been in a continual open state for the time specified by Door_Open_Too_Long_Time after the Door_Extended_Pulse_Time has expired.
(c) The Present_Value had a previous value of UNLOCK and the door has been in a continual open state for the time specified by Door_Open_Too_Long_Time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.26.20 Door_Alarm_State

This property, of type BACnetDoorAlarmState, is the alarm state for the physical door and is restricted to the values NORMAL and those contained in Alarm_Values and Fault_Values. When no alarm or fault condition exists for this object, this property shall take on the value NORMAL. It is considered a local matter as to when this property is set to a non-normal value. It is up to the internal control logic to take Lock_Status, Door_Status, Present_Value and information from other objects into account when calculating the proper alarm state. However, this property cannot take on any value which is also in the Masked_Alarm_Values list. If the property is currently set to a specific state and that state is written to the Masked_Alarm_Values list, then the Door_Alarm_State will immediately return to the NORMAL state.

This property, if present, is required to be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, then this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.26.21 Masked_Alarm_Values

This property, of type BACnetLIST of BACnetDoorAlarmState, shall specify any alarm and/or fault states which are masked. An alarm state which is currently masked will prevent the Door_Alarm_State property from being equal to that state.

12.26.22 Maintenance_Required

This property, of type BACnetMaintenance, shall indicate the type of maintenance required for the Access Door. This may be periodic maintenance, or a "parameter-determined" maintenance, such as maximum duty-cycle for a door lock, and shall be determined locally.

12.26.23 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.26.24 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.26.25 Alarm_Values

This property, of type BACnetLIST of BACnetDoorAlarmState, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.26.26 Fault_Values

This property, of type BACnetLIST of BACnetDoorAlarmState, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.26.27 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.26.28 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.26.29 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.26.30 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.26.31 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.26.32 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.26.33 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.26.34 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.26.35 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.26.36 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.26.37 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.26.38 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.26.39 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.26.40 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.26.41 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.26.42 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.26.43 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.26.44 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.26.45 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.26.46 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.26.47 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.26.48 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.26.49 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.27 Event Log Object Type

An Event Log object records event notifications with timestamps and other pertinent data in an internal buffer for subsequent retrieval. Each timestamped buffer entry is called a "log record."

Each Event Log object maintains an internal, optionally fixed-size buffer. This log buffer fills or grows as event log records are added. If the log buffer becomes full, the least recent log records are overwritten when new log records are added, or collection may be set to stop. Log buffers are transferred as a list of BACnetEventLogRecord using the ReadRange service. The log buffer may be cleared by writing a zero to the Record_Count property. The determination of which event notifications are placed into the log buffer is a local matter, including those never transmitted as an APDU. Each log record in the log buffer has an implied SequenceNumber that is equal to the value of the Total_Record_Count property immediately after the log record is added.

Logging may be enabled and disabled through the Enable property and at dates and times specified by the Start_Time and Stop_Time properties. Event Log enabling and disabling is recorded in the log buffer.

Event reporting (notification) may be provided to facilitate automatic fetching of log records by processes on other devices such as fileservers. Support is provided for algorithmic reporting; optionally, intrinsic reporting may be provided. Event Log objects that support intrinsic reporting shall apply the BUFFER_READY event algorithm.

In intrinsic reporting, when the number of records specified by the Notification_Threshold property has been collected since the previous notification (or startup), a new notification is sent to all recipients enrolled in the respective Notification Class object.

In response to a notification, recipients may fetch all of the new log records. If a recipient needs to fetch all of the new log records, it should use the 'By Sequence Number' form of the ReadRange service request.

A missed notification may be detected by a recipient if the 'Current Notification' parameter received in the previous BUFFER_READY notification is different than the 'Previous Notification' parameter of the current BUFFER_READY notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has the FIRST_ITEM bit of the 'Result Flags' parameter set to TRUE, log records have probably been missed by this recipient.

The acquisition of log records by remote devices has no effect upon the state of the Event Log object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its log records at any time.

Table 12-31. Properties of the Event Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Start_Time	BACnetDateTime	O[1, 2]
Stop_Time	BACnetDateTime	O[1, 2]
Stop_When_Full	BOOLEAN	R
Buffer_Size	Unsigned32	R
Log_Buffer	BACnetLIST of BACnetEventLogRecord	R
Record_Count	Unsigned32	W
Total_Record_Count	Unsigned32	R
Notification_Threshold	Unsigned32	O[3,5]
Records_Since_Notification	Unsigned32	O[3,5]
Last_Notify_Record	Unsigned32	O[3,5]
Notification_Class	Unsigned	O[3,5]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Audit_Level	BACnetAuditLevel	O[8]
Auditable_Operations	BACnetAuditOperationFlags	O[8]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If present, these properties are required to be writable.
[2] If one of these properties is present, then all shall be present.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.
[8] This property shall be present only if the device supports audit reporting.

12.27.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.27.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.27.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be EVENT_LOG.

12.27.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.27.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Event Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

FAULT — Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

OVERRIDDEN	The value of this flag shall be Logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.27.6 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.27.7 Reliability

This property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet device can determine.

12.27.8 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains an unspecified datetime, then it shall be considered equal to 'the start of time'. If Stop_Time contains an unspecified datetime, then it shall be considered equal to 'the end of time'. As an exception, log records of type log-status in which the BUFFER_PURGED flag is set to TRUE shall always be recorded.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and either Record_Count is equal to Buffer_Size, or Buffer_Size is 2^{32}-1 and there is no space for another record, shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

12.27.9 Start_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. This property shall be writable if present.

When Start_Time is reached, the value of the Enable property is not changed.

12.27.10 Stop_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be disabled by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. This property shall be writable if present.

When Stop_Time is reached, the value of the Enable property is not changed.

12.27.11 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the log buffer is full. When logging ceases because the addition of one more log record would cause the log buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest log record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

If Stop_When_Full is TRUE and Buffer_Size is 2^{32}-1 and the addition of a new record fails due to a lack of space, then the oldest Log_Buffer record shall be discarded, the Enable property shall be set to FALSE and the event is recorded.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.27.12 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter. If all records are deleted when the Buffer_Size is written then the object shall act as if the Record_Count was set to zero.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

12.27.13 Log_Buffer

This property, of type BACnetLIST of BACnetEventLogRecord, is a list of up to Buffer_Size timestamped log records of datatype BACnetEventLogRecord, each of which conveys the event notification parameters or status changes in the Event Log object. Each log record has data fields as follows:

Timestamp The local date and time that the log record was placed into the log buffer.

LogDatum The notification information, or a change in status or operation of the Event Log object itself.

The choices available for the LogDatum are listed below:

log-status — This choice represents a change in the status or operation of the Event Log object. Whenever one of the events represented by the flags listed below occurs, a record shall be appended to the log buffer.

- LOG_DISABLED — This flag is changed whenever collection of log records by the Event Log object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this log record will cause the log buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.
- BUFFER_PURGED — This flag shall be set to TRUE whenever the buffer is cleared by writing zero to the Record_Count property. After this value is recorded in the log buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the log buffer shall be placed into the log buffer even if logging is disabled, the local time is outside of the time range defined by the Start_Time and Stop_Time properties, or the log buffer was completely empty before the request for clearing.
- LOG_INTERRUPTED — This flag indicates that the collection of log records by the Event Log object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that events prior to this log record might have been missed.

notification — This choice represents an event notification that was received. It consists of the body of the ConfirmedEventNotification or UnconfirmedEventNotification. If the event was generated locally, this shall hold what would be received if the Event Log object existed on a remote device. In such a case the value of the Process Identifier parameter is a local matter.

time-change — This choice represents a change in the clock setting in the device; it records the number of seconds and fraction of a second by which the clock changed. If, and only if, the number is not known, such as when the clock is initialized for the first time, the value recorded shall be 0.0. This record shall be recorded after changing the local time of the device and the timestamp shall reflect the new local time of the device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Also associated with each log record is an implied log record number, the value of which is equal to Total_Record_Count at the point where the record has been added into the log buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the Event Log object is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one.

The log buffer is not network accessible except through the use of the ReadRange service in order to avoid problems with log record sequencing when segmentation is required.

If the object supports event reporting, then a reference to this property shall be the pLogBuffer parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.27.14 Record_Count

This property, of type Unsigned32, shall represent the number of log records currently resident in the log buffer. A write of the value zero to this property shall cause all log records in the log buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported as a BUFFER_PURGED log-status log record in the log buffer as the initial entry regardless of the state of the log buffer before the write.

12.27.15 Total_Record_Count

This property, of type Unsigned32, shall represent the total number of log records collected by the Event Log object since creation. When the value of Total_Record_Count reaches its maximum possible value of 2^{32} - 1, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost but its use in generating notifications remains.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.27.16 Notification_Threshold

This property, of type Unsigned32, is the pThreshold parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.27.17 Records_Since_Notification

This property, of type Unsigned32, represents the number of log records collected since the previous notification, or since the beginning of logging if no previous notification has occurred.

12.27.18 Last_Notify_Record

This property, of type Unsigned32, is the pPreviousCount parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.27.19 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.27.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.27.21 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.27.22 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.27.23 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.27.24 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.27.25 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.27.26 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.27.27 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.27.28 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.27.29 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.27.30 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.27.31 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.27.32 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.27.33 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.27.34 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.27.35 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.28 Load Control Object Type

The Load Control object type defines a standardized object whose properties represent the externally visible characteristics of a mechanism for controlling load requirements. A BACnet device can use a Load Control object to allow external control over the shedding of a load that it controls. The mechanisms by which the loads are shed are not visible to the BACnet client. One or more objects may be used in the device to allow independent control over different sub-loads. The Load Control object may also be used in a hierarchical fashion to control other Load Control objects in other BACnet devices.

A BACnet client (controller) can request that the Load Control object shed a portion of its load for a specified time by writing to the four properties: Requested_Shed_Level, Start_Time, Shed_Duration, and Duty_Window. For any given shed request, which may arrive while a previous request is pending or active, each of these parameters is optional except for Start_Time, which must be written if no shed request is pending or active. If no shed request is pending or active, only the writing of Start_Time will cause the Load Control object to become active. Modification of these shed request parameters serves to configure the load shed command. Initial values of these properties, and the values taken at the completion of a shed command execution, are as specified in the individual property descriptions.

The Load Control object shed mechanism follows a state machine whose operation is displayed in Figure 12-5. This state machine only describes the behavior of the Load Control object when the Enable property has the value TRUE. See Clause 12.28.14 for a description of the effect of this property. The state machine captures the transitions that occur within the Load Control object.

If the device is unable to comply fully with the shed request by shedding the entire amount of load requested, it is a local matter whether the device sheds as much load as it can or whether it does not shed any of its loads. Determination of compliance with a client's load shed request may also be affected by other factors, such as the definition of the baseline usage, synchronization of time between the client and the device containing the Load Control object, and any intrinsic limits on shed amounts that the device may have. If these factors are not in agreement, the client's determination of compliance may not match the object's determination.

The activity of a Load Control object in the SHED_REQUEST_PENDING state will vary. For a Load Control object controlling only one or more direct loads that it can shed instantly, the activity will be simply waiting for the first duty window to arrive, at which point it will monitor the clock and cycle on/off or begin modulation of loads or reduce loads by some other means. The object may need to begin shedding some of the loads before Start_Time in order to meet the shed target by Start_Time, in which case it will enter the SHED_NON_COMPLIANT state. For a Load Control object controlling other Load Control objects subordinate to it, the shedding activity will begin prior to Start_Time by communicating the shed request (possibly modified) to these other Load Control objects. There may be some Load Control objects that indicate an inability to comply with the request, which may lead to requests for increased load reduction from these or other Load Control objects.

While the Load Control object is designed to allow independent operation, it is possible that there will exist within a building (or even within a device) a hierarchy of Load Control objects, where one Load Control object receives a load shed command, possibly from a non-BACnet client (e.g., a utility), and the controller which hosts that object (the master) in turn will be responsible for managing and issuing requests to other Load Control objects. There may be a negotiation between the master and its subordinate Load Control objects. The master uses WriteProperty or WritePropertyMultiple to set shed request parameters in the subordinate Load Control objects. A subordinate Load Control object would then set its Expected_Shed_Level property to the value that it expects to be able to achieve after Start_Time. Before Start_Time, the master object can read the Expected_Shed_Level properties of its subordinates to determine expected compliance with the request. After Start_Time plus Duty_Window, the Actual_Shed_Level properties of the subordinate objects will reflect the actual amount shed in the past Duty_Window. If by reading these properties the master Load Control object determines that one or more subordinate objects cannot completely comply with the request, the master may choose to modify the shed requests to subordinates, such that the overall shed target is achieved. For instance, it may request that another object shed a greater amount of its load or it may choose to request that the noncompliant device shed a greater amount. This negotiation could be repeated at each successive level in the hierarchy. If the subordinate Load Control objects also support intrinsic reporting, expected or actual instances of non-compliance can be reported to the master object using event notifications.

Where large loads are concerned, it is expected that the master Load Control object will employ sequencing to distribute the startup and shutdown of managed loads. When the load control master is used in a gateway to a non-BACnet load

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

control client, such as a utility company, the gateway shall accept and process any start randomization commands and accordingly distribute the initiation of load control requests to its subordinate Load Control objects.

The Load Control object shall exhibit restorative behavior across a restart or time change of the BACnet device in which it resides. The shed request property values shall be maintained across a device restart. Upon device restart or a time change, the object shall behave as if Start_Time were written and shall re-evaluate the state machine's state.

Load Control objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm.

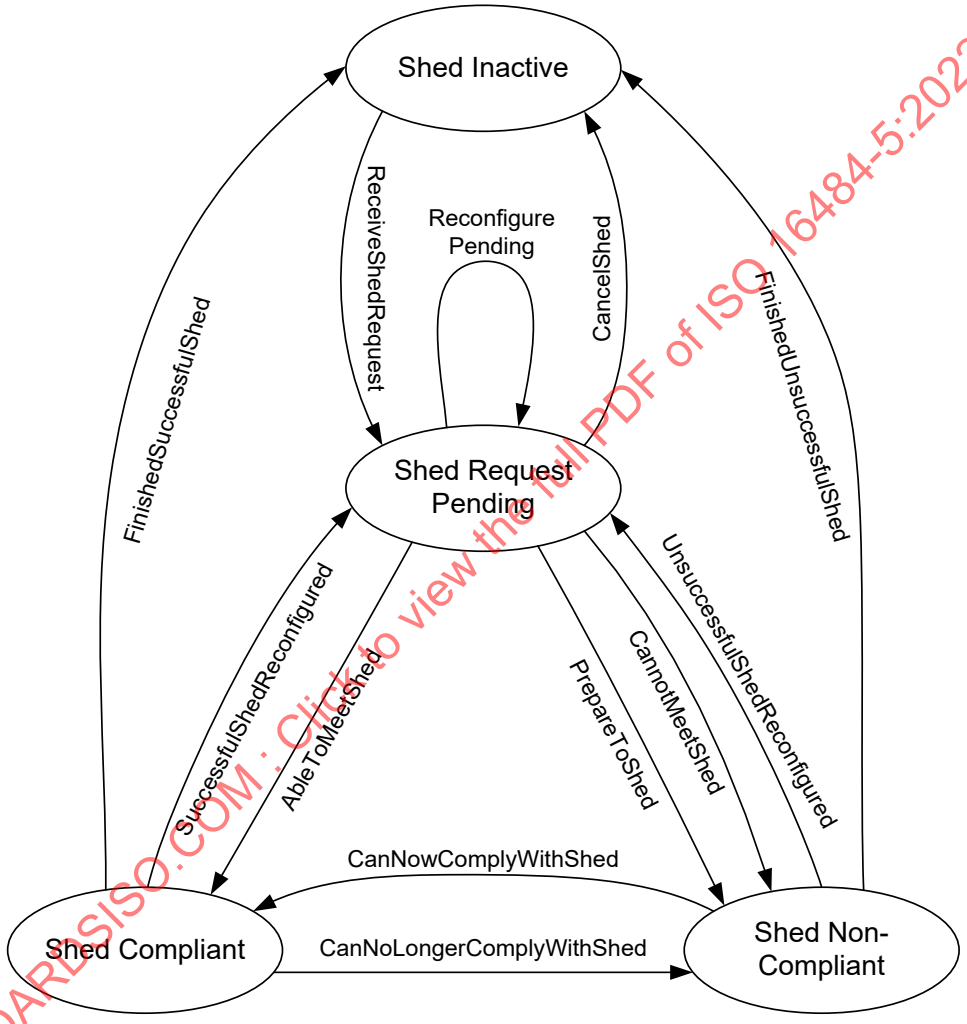


Figure 12-5. State Diagram for Load Control Object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

SHED_INACTIVE

In the SHED_INACTIVE state, the Load Control object waits for a shed request.

ReceiveShedRequest

If Start_Time is written, the object shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_REQUEST_PENDING state.

SHED_REQUEST_PENDING

In the SHED_REQUEST_PENDING state, the object makes a determination from the newly written shed parameters whether the shed request needs to be executed immediately or at some time in the future.

© ISO 2022 – All rights reserved

CancelShed
If the current time is after Start_Time plus Shed_Duration, this request is for an invalid time and is ignored. The object shall stop shedding and enter the SHED_INACTIVE state.

If Requested_Shed_Level is equal to the default value for the choice, or Start_Time contains an unspecified datetime, then this is a cancellation of shedding. The object shall stop shedding and enter the SHED_INACTIVE state.

ReconfigurePending
If the current time is prior to Start_Time, and a new write is received for Requested_Shed_Level, Shed_Duration, Duty_Window, or Start_Time, this is a reconfiguration of the shed request. The object shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_REQUEST_PENDING state.

PrepareToShed
If the current time is prior to Start_Time, but the loads to be shed require time to decrease usage to the requested shed level, the object may choose to initiate shedding of its subordinates prior to Start_Time in order to be in compliance by Start_Time. If this approach is followed, the object shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_NON_COMPLIANT state.

CannotMeetShed
If the current time is after Start_Time, and the object is unable to meet the shed request immediately, it shall begin shedding its loads, calculate Expected_Shed_Level and Actual_Shed_Level, and enter the SHED_NON_COMPLIANT state.

AbleToMeetShed
If the current time is after Start_Time and the object is able to achieve the shed request immediately, it shall shed its loads, calculate Expected_Shed_Level and Actual_Shed_Level, and enter the SHED_COMPLIANT state.

If the current time is before Start_Time, and the object has initiated shedding prior to Start_Time in order to be in compliance by Start_Time, and the object has achieved the requested shed level, it shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_COMPLIANT state.

SHED_NON_COMPLIANT

In the SHED_NON_COMPLIANT state, the object attempts to meet the shed request until the shed is achieved, the object is reconfigured, or the request has completed unsuccessfully.

FinishedUnsuccessfulShed
If the current time is after Start_Time plus Shed_Duration, the shed request has completed unsuccessfully. The object shall stop shedding and enter the SHED_INACTIVE state.

UnsuccessfulShedReconfigured
If the object receives a write to any of the properties Requested_Shed_Level, Shed_Duration, Duty_Window, or Start_Time, the object shall enter the SHED_REQUEST_PENDING state.

CanNowComplyWithShed
If the object has achieved the Requested_Shed_Level, it shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_COMPLIANT state.

SHED_COMPLIANT

In the SHED_COMPLIANT state, the object continues meeting the shed request until the shed is either reconfigured or completes, or conditions change and the object is no longer able to maintain the requested shed level.

FinishedSuccessfulShed
If the current time is after Start_Time plus Shed_Duration, the shed request has completed successfully. The object shall stop shedding, set Start_Time to an unspecified datetime, and enter the SHED_INACTIVE state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

SuccessfulShedReconfigured

If the object receives a write to any of the properties Requested_Shed_Level, Shed_Duration, Duty_Window, or Start_Time, the object shall enter the SHED_REQUEST_PENDING state.

CanNoLongerComplyWithShed

If the object is no longer able to maintain the Requested_Shed_Level, it shall calculate Expected_Shed_Level and Actual_Shed_Level and enter the SHED_NON_COMPLIANT state.

Table 12-32. Properties of the Load Control Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BACnetShedState	R
State_Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Requested_Shed_Level	BACnetShedLevel	W
Start_Time	BACnetDateTime	W
Shed_Duration	Unsigned	W
Duty_Window	Unsigned	W
Enable	BOOLEAN	W
Full_Duty_Baseline	REAL	O
Expected_Shed_Level	BACnetShedLevel	R
Actual_Shed_Level	BACnetShedLevel	R
Shed_Levels	BACnetARRAY[N] of Unsigned	W[1]
Shed_Level_Descriptions	BACnetARRAY[N] of CharacterString	R
Notification_Class	Unsigned	O[2,4]
Time_Delay	Unsigned	O[2,4]
Event_Enable	BACnetEventTransitionBits	O[2,4]
Acked_Transitions	BACnetEventTransitionBits	O[2,4]
Notify_Type	BACnetNotifyType	O[2,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Event_Detection_Enable	BOOLEAN	O[2,4]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[4]
Event_Algorithm_Inhibit	BOOLEAN	O[4,5]
Time_Delay_Normal	Unsigned	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[6]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[7,8,9]
Audit_Level	BACnetAuditLevel	O[10]
Auditable_Operations	BACnetAuditOperationFlags	O[10]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] The elements of this array are required to be writable, although the array is not required to be resizable.
[2] These properties are required if the object supports intrinsic reporting.
[3] Footnote removed.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[6] If this property is present, then the Reliability property shall be present.
[7] This property is required if the object supports the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[8] This property shall be present only if the object supports the value source mechanism.
[9] This property shall be writable as described in Clause 19.5.
[10] This property shall be present only if the device supports audit reporting.

12.28.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.28.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.28.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LOAD_CONTROL.

12.28.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.28.5 Present_Value

This property, of type BACnetShedState, indicates the current load shedding state of the object. See Figure 12-5 for a diagram of the state machine governing the value of Present_Value.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm and the pAlarmValues parameter shall have the value SHED_NON_COMPLIANT. See Clause 13.3 for event algorithm parameter descriptions.

12.28.6 State_Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted. The State_Description provides additional information for human operators about the shed state of the Load Control object.

12.28.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Load Control object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

IN_ALARM Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

FAULT Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device, otherwise logical FALSE (0).

OUT_OF_SERVICE This bit shall always be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.28.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.28.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Load Control object is reliably reporting its compliance with any load shed requests.

12.28.10 Requested_Shed_Level

This property, of type BACnetShedLevel, indicates the desired load shedding. Table 12-33 describes the default values and power targets for the different choices of Requested_Shed_Level.

If the choice for Requested_Shed_Level is PERCENT, the value of Requested_Shed_Level is interpreted as a requested percentage of Full Duty to which the device is to attempt to reduce its load. The determination of the Full Duty rating (or some alternative baseline power usage) is a local matter. It may be determined from the Full_Duty_Baseline property, if present.

If the choice for Requested_Shed_Level is LEVEL, the value of Requested_Shed_Level is used to set a preconfigured level of load shedding.

The Load Control object's available shed actions are described by the Shed_Level_Descriptions array and are mapped to the BACnet visible values of Requested_Shed_Level by the Shed_Levels array. The SHED_INACTIVE state shall always be represented by the value 0, which is not represented in the Shed_Levels or Shed_Level_Descriptions arrays. If Requested_Shed_Level choice is AMOUNT, the value of Requested_Shed_Level shall be interpreted as an amount, in kilowatts, by which to reduce power usage. Load Control objects are required to support the LEVEL choice. Support for the PERCENT and AMOUNT choices is optional. This allows a master to be guaranteed the ability to write to the Load Control object by using the LEVEL choice.

If a load control command has been issued, and execution of the command has completed, Requested_Shed_Level shall be reset to the default value appropriate to the choice of Requested_Shed_Level used for the last command.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Table 12-33. Requested_Shed_Level Default Values and Power Targets

Choice	Default Requested_Shed_Level value	Power load target in kW
PERCENT	100	(current baseline) * Requested_Shed_Level / 100
LEVEL	0	locally pre-specified shed target for the given level
AMOUNT	0.0	(current baseline) - Requested_Shed_Level

12.28.11 Start_Time

This property, of type BACnetDateTime, indicates the start of the duty window in which the load controlled by the Load Control object must be compliant with the requested shed. Load shedding (or determination of loads to shed) may need to begin before Start_Time in order to be compliant with the shed request by Start_Time. If no shed request is pending

© ISO 2022 – All rights reserved

or active, Start_Time shall contain an unspecified datetime value. If a load control command has been issued, and execution of the command has completed, Start_Time shall be reset by the device to contain an unspecified datetime value. If a client wishes to initiate an immediate shed, it can set Start_Time to a value prior to the device's current time.

12.28.12 Shed_Duration

This property, of type Unsigned, indicates the duration of the load shed action, starting at Start_Time. The units for Shed_Duration are minutes. If no shed request is pending or active, Shed_Duration shall be zero. If a load control command has been issued, and execution of the command has completed, Shed_Duration shall be reset by the device to zero.

12.28.13 Duty_Window

This property, of type Unsigned, indicates the time window used for load shed accounting. The units for Duty_Window are minutes. Duty_Window is used for performance measurement or compliance purposes. The average power consumption across a duty window must be less than or equal to the requested reduced consumption. It is a local matter whether this window is fixed or sliding. The first Duty_Window begins at Start_Time. If a shed request is received with no value written to this property, Duty_Window shall be set to some pre-agreed upon value. If a load control command has been issued, and execution of the command has completed, Duty_Window shall be reset by the device to this pre-agreed value.

12.28.14 Enable

This property, of type BOOLEAN, indicates and controls whether the Load Control object is currently enabled to respond to load shed requests. If Enable is TRUE, the object will respond to load shed requests normally and follow the state machine described in Figure 12-5. If Enable is FALSE, the object will transition to the SHED_INACTIVE state if necessary and remain in that state. It shall not respond to any load shed request while Enable is FALSE.

12.28.15 Full_Duty_Baseline

This property, of type REAL, indicates the baseline power consumption value for the sheddable load controlled by this object, if a fixed baseline is used. Shed requests may be made with respect to this baseline, that is, to "percent of baseline" and "amount off baseline". The units of Full_Duty_Baseline are kilowatts.

12.28.16 Expected_Shed_Level

This property, of type BACnetShedLevel, indicates the amount of power that the object expects to be able to shed in response to a load shed request. When the object is in the SHED_INACTIVE state, this value shall be equal to the default value of Requested_Shed_Level. When a shed request is pending or active, Expected_Shed_Level shall be equal to the shed level the object expects to be able to achieve at Start_Time. Expected_Shed_Level allows a client (e.g., a master-level Load Control object) to determine if a pending shed request needs to be modified in order to achieve the requested shed level, in the event that Expected_Shed_Level is less than the Requested_Shed_Level. The units for Expected_Shed_Level are the same as the units for Requested_Shed_Level.

12.28.17 Actual_Shed_Level

This property, of type BACnetShedLevel, indicates the actual amount of power being shed in response to a load shed request. When the object is in the SHED_INACTIVE state, this value shall be equal to the default value of Requested_Shed_Level. After Start_Time plus Duty_Window has elapsed, this value shall be the actual shed amount as calculated based on the average value over the previous duty window. The units for Actual_Shed_Level are the same as the units for Requested_Shed_Level.

12.28.18 Shed_Levels

This property, of type BACnetARRAY of Unsigned, represents the shed levels for the LEVEL choice of BACnetShedLevel that have meaning for this particular Load Control object. The array shall be ordered by increasing shed amount. When commanded with the LEVEL choice, the Load Control object shall take a shedding action described by the corresponding element in the Shed_Level_Descriptions array. If the Load Control object is commanded to go to a level that is not in the Shed_Levels array, it shall go to the Shed_Level whose entry in the Shed_Levels array has the nearest numerically lower value. The elements of the array are required to be writable, allowing local configuration of how this Load Control object will participate in load shedding for the facility. This array is not required to be resizable through BACnet write services.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The size of this array shall be equal to the size of the Shed_Level_Descriptions array. The behavior of this object when the Shed_Levels array contains duplicate entries is a local matter.

12.28.19 Shed_Level_Descriptions

This property, of type BACnetARRAY of CharacterString, contains a description of the shed levels that the Load Control object can take on. This allows a local configuration tool to provide to a user an understanding of what each shed level in this Load Control object's load shedding algorithm will do. The level at which each shed action will occur can then be configured by writing to the Shed_Levels property.

12.28.20 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.28.21 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.28.22 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.28.23 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.28.24 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.28.25 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.28.26 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.28.27 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.28.28 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.28.29 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.28.30 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.28.31 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.28.32 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.28.33 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.28.34 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.28.35 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.28.36 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.28.37 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.28.38 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.28.39 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.29 Structured View Object Type

The Structured View object type defines a standardized object that provides a container to hold references to subordinate objects, which may include other Structured View objects, thereby allowing multilevel hierarchies to be created. The hierarchies are intended to convey a structure or organization such as a geographical distribution or application organization. Subordinate objects may reside in the same device as the Structured View object or in other devices on the network.

The Structured View object and its properties are summarized in Table 12-34 and described in detail in this clause.

Table 12-34. Properties of the Structured View Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Node_Type	BACnetNodeType	R
Node_Subtype	CharacterString	O
Subordinate_List	BACnetARRAY[N] of BACnetDeviceObjectReference	R
Subordinate_Annotations	BACnetARRAY[N] of CharacterString	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Subordinate_Tags	BACnetARRAY[N] of BACnetNameValueCollection	O
Subordinate_Node_Types	BACnetARRAY[N] of BACnetNodeType	O
Subordinate_Relationships	BACnetARRAY[N] of BACnetRelationship	O
Default_Subordinate_Relationship	BACnetRelationship	O
Represents	BACnetDeviceObjectReference	O
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property shall be present only if the device supports audit reporting.

12.29.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.29.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.29.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be STRUCTURED_VIEW.

12.29.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.29.5 Node_Type

This property, of type BACnetNodeType, provides a general classification of the object in the hierarchy of objects.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

It is intended as a general suggestion to a client application about the contents of a Structured View object, and is not intended to convey an exact definition. Further refinement of classification is provided by the Node_Subtype property. The allowable values for this property are:

{UNKNOWN, SYSTEM, SUBSYSTEM, NETWORK, DEVICE, ORGANIZATIONAL, AREA, BUILDING, FLOOR, ROOM, ZONE, EQUIPMENT, SECTION, MODULE, POINT, COLLECTION, PROPERTY, FUNCTIONAL, TREE, MEMBER, PROTOCOL, OTHER}

Where the following are suggested interpretations:

UNKNOWN	Indicates that a value for Node_Type is not available or has not been configured at this time
SYSTEM	An entire mechanical system
SUBSYSTEM	A portion of a mechanical system
NETWORK	A communications network
DEVICE	Contains a set of elements which collectively represents a BACnet device, a logical device, or a physical device
ORGANIZATIONAL	Business concepts such as departments or people
AREA	Geographical concept such as a region, area, campus, etc.
BUILDING	Geographical concept of a building
FLOOR	Geographical concept of a floor of a building
ROOM	Geographical concept of a room of a building
ZONE	Abstract concept that may span or subdivide geographical boundaries for a particular purpose such as HVAC, lighting, security, fire, etc.
EQUIPMENT	Single piece of equipment that may be a collection of "Points"
SECTION	A section of a piece of equipment that may be a collection of "Points" or other "Sections"
MODULE	A modular part of a device (physical or logical)
POINT	Contains a set of elements which collectively defines a single point of data, either a physical input or output of a control or monitoring device, or a software calculation or configuration setting
COLLECTION	A generic container used to group things together, such as a collection of references to all space temperatures in a building
PROPERTY	Defines a characteristic or parameter of the parent node
FUNCTIONAL	Single system component such as a control module or a logical component such as a function block
TREE	A logical hierarchical arrangement
MEMBER	Part of a collection
PROTOCOL	A grouping of data reachable by a single protocol
OTHER	Everything that does not fit into one of these broad categories

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.29.6 Node_Subtype

This property, of type CharacterString, is a string of printable characters whose content is not restricted. It provides a more specific classification of the object in the hierarchy of objects, providing a short description of the item represented by the node.

12.29.7 Subordinate_List

This property, of type BACnetARRAY of BACnetDeviceObjectReference, defines the members of the current Structured View.

By including references to 'child' Structured View objects, multilevel hierarchies may be created.

If the optional device identifier is not present for a particular Subordinate_List member, then that member resides in the same device that maintains the Structured View object. If Subordinate_List is writable using WriteProperty services, the Subordinate_List may optionally be restricted to reference-only objects in the local device. To avoid recursion, it is suggested that a single Structured View object should be referenced only once in the hierarchy.

If the size of the Subordinate_List array is changed, the size of the Subordinate_Annotations, Subordinate_Tags, Subordinate_Relationships, and Subordinate_Node_Types arrays, if present, shall also be changed to the same size. Uninitialized Subordinate_List array elements shall be given the instance number 4194303.

A Subordinate_List array element whose instance number is equal to 4194303 shall be considered uninitialized and shall be ignored.

12.29.8 Subordinate_Annotations

This property, of type BACnetARRAY of Characterstring, shall be used to define a text string description for each member of the Subordinate_List. The content of these strings is not restricted.

If the size of this array is changed, the size of the Subordinate_List, Subordinate_Tags, Subordinate_Relationships, and Subordinate_Node_Types arrays shall also be changed to the same size.

12.29.9 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.29.10 Subordinate_Tags

This property, of type BACnetARRAY of BACnetNameValueCollection, provides individual collections of tags for each of the subordinates. See Clause 21.4 for restrictions on the string values used for the names of these tags and a general description of tagging and the mechanism by which tags are defined.

If the size of this array is changed, the size of the Subordinate_List, Subordinate_Annotations, Subordinate_Node_Types, and Subordinate_Relationships arrays, if present, shall also be changed to the same size. Uninitialized Subordinate_Tags array elements shall be empty collections.

12.29.11 Subordinate_Node_Types

This property, of type BACnetARRAY of BACnetNodeType, shall be used to provide node type (Clause 12.29.5) information for each member of the Subordinate_List. If the subordinate object has its own Node_Type, it is intended that the local value in this property logically override the subordinate's Node_Type unless the local value is "UNKNOWN", in which case, the subordinate's Node_Type is used.

If the size of this array is changed, the size of the Subordinate_List, Subordinate_Annotations, Subordinate_Tags, and Subordinate_Relationships arrays, if present, shall also be changed to the same size. Uninitialized Subordinate_Node_Types array elements shall have the value UNKNOWN.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.29.12 Subordinate_Relationships

This property, of type BACnetARRAY of BACnetRelationship, shall be used to describe the relationship to each member of the Subordinate_List. If this property is absent, then relationship to each of the subordinates is equal to the value of the Default_Subordinate_Relationship property, if present, else equal to UNKNOWN.

If the size of this array is changed, the size of the Subordinate_List, Subordinate_Annotations, Subordinate_Tags, and Subordinate_Node_Types arrays, if present, shall also be changed to the same size. Uninitialized Subordinate_Relationships array elements shall be equal to DEFAULT.

12.29.13 Default_Subordinate_Relationship

This property, of type BACnetRelationship, shall be used to describe the default relationship to each member of the Subordinate_List, unless overridden by a member of the Subordinate_Relationships property. If this property is absent and the Subordinate_Relationships is also absent, the relationship to the subordinates is equal to UNKNOWN.

12.29.14 Represents

This property, of type BACnetDeviceObjectReference, can be used to indicate the entity for which this Structured View is modeling the subordinates.

In some cases, the Structure View object will abstractly represent this entity by itself, and this property will either be absent, unconfigured, or point to itself. For example, this could be a Structured View object that abstractly represents a simple rooftop unit, where the subordinates are its points. This Structured View therefore logically "is" the rooftop unit and thus is the parent of the subordinates.

In other cases, there may be a separate concrete object that represents the entity. This Structured View therefore logically provides the subordinates for that other object. For example, the other object could be another Structured View, a Program object, a Load Control object, or any other object that best represents the entity. In this case, the Represents property can be configured to point to that object and the subordinates configured in this Structured View are then logically subordinates of that object.

In this manner, multiple Structured View objects can be used to provide multiple sets of subordinates for a single entity, thereby providing deployment flexibility. For example, if an entity has relationship X with one set of subordinates and has relationship Y with another set of subordinates, deployments can, among other possibilities:

(a) use one Structured View object with all the subordinates combined into a single array and use the Subordinate_Relationships property to select between X and Y for each of the subordinates, or
(b) use two Structured View objects, one for the subordinates with relationship X and another for those with relationship Y. Since the relationships are segregated, the Default_Subordinate_Relationship property can be used in each of the two views to select the relationship for all of that view's subordinates, and the Represents property of the two views can be used to tie them to a common entity.

12.29.15 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.29.16 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.29.17 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.29.18 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.29.19 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30 Trend Log Multiple Object Type

A Trend Log Multiple object monitors one or more properties of one or more referenced objects, either in the same device as the Trend Log Multiple object or in an external device. When predefined conditions are met, the object saves ("logs") the value of the properties and a timestamp into an internal log buffer for subsequent retrieval. The data may be logged periodically or when "triggered" by a write to the Trigger property. Errors that prevent the acquisition of the data, as well as changes in the status or operation of the logging process itself, are also recorded. Each timestamped log buffer entry is called a "log record".

The Log_DeviceObjectProperty array holds the list of properties to be monitored and logged. If an element of the Log_DeviceObjectProperty array has an object or device instance number equal to 4194303, this indicates that the element is 'empty' or 'uninitialized'. For empty or uninitialized elements, an indication that no property was specified shall be written to the corresponding entry in each log record.

Each Trend Log Multiple object maintains an internal, optionally fixed-size, log buffer. This log buffer fills or grows as log records are added. If the log buffer becomes full, the least recent log record is overwritten when a new log record is added, or collection may be set to stop. Log buffers are transferred as a list of BACnetLogMultipleRecord using the ReadRange service. The log buffer may be cleared by writing a zero to the Record_Count property. Each log record in the log buffer has an implied SequenceNumber that is equal to the value of the Total_Record_Count property immediately after the log record is added.

Logging may be enabled and disabled through the Enable property and at dates and times specified by the Start_Time and Stop_Time properties. The enabling and disabling of log record collection is recorded in the log buffer.

Event reporting (notification) may be provided to facilitate automatic fetching of log records by processes on other devices such as fileservers. Mechanisms for both algorithmic and intrinsic reporting are provided. Trend Log Multiple objects that support intrinsic reporting shall apply the BUFFER_READY event algorithm.

In intrinsic reporting, when the number of log records specified by the Notification_Threshold property has been collected since the previous notification (or startup), a new notification is sent to all recipients enrolled in the respective Notification Class object.

In response to a notification, recipients may fetch all of the new log records. If a recipient needs to fetch all of the new log records, it should use the 'By Sequence Number' form of the ReadRange service request.

A missed notification may be detected by a recipient if the 'Current Notification' parameter received in the previous BUFFER_READY notification is different than the 'Previous Notification' parameter of the current BUFFER_READY notification. If the ReadRange-ACK response to the ReadRange request issued under these conditions has the FIRST_ITEM bit of the 'Result Flags' parameter set to TRUE, log records have probably been missed by this recipient.

The acquisition of log records by remote devices has no effect upon the state of the Trend Log Multiple object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its records at any time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Table 12-35. Properties of the Trend Log Multiple Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Start_Time	BACnetDateTime	O[1]
Stop_Time	BACnetDateTime	O[1]
Log_DeviceObjectProperty	BACnetARRAY[N] of BACnetDeviceObjectPropertyReference	R
Logging_Type	BACnetLoggingType	R

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Log_Interval	Unsigned	R[2]
Align_Intervals	BOOLEAN	O[3]
Interval_Offset	Unsigned	O[3]
Trigger	BOOLEAN	O
Stop_When_Full	BOOLEAN	R
Buffer_Size	Unsigned32	R
Log_Buffer	BACnetLIST of BACnetLogMultipleRecord	R
Record_Count	Unsigned32	W
Total_Record_Count	Unsigned32	R
Notification_Threshold	Unsigned32	O[4,6]
Records_Since_Notification	Unsigned32	O[4,6]
Last_Notify_Record	Unsigned32	O[4,6]
Notification_Class	Unsigned	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If present, these properties are required to be writable.
[2] This property is required to be writable when Logging_Type has the value POLLED and is required to be read-only when Logging_Type has the value TRIGGERED.
[3] These properties are required if, and shall be present only if, the object supports clock-aligned logging.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] This property shall be present only if the device supports audit reporting.

12.30.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.30.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.30.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TREND_LOG_MULTIPLE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.30.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Trend Log Multiple object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be Logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.30.6 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.30.7 Reliability

This property, of type BACnetReliability, provides an indication of whether the application-specific properties of the object or the process executing the application program are "reliable" as far as the BACnet device can determine.

12.30.8 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains an unspecified datetime, then it shall be considered equal to 'the start of time'. If Stop_Time contains an unspecified datetime, then it shall be considered equal to 'the end of time'. As an exception, log records of type log-status in which the BUFFER_PURGED flag is set to TRUE shall always be recorded.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and either Record_Count is equal to Buffer_Size, or Buffer_Size is 2^{32}-1 and there is no space for another record, shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

12.30.9 Start_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be enabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be enabled by Start_Time shall be ignored. If Start_Time specifies a date and time after Stop_Time, then logging shall be disabled. This property shall be writable if present.

When Start_Time is reached, the value of the Enable property is not changed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30.10 Stop_Time

This property, of type BACnetDateTime, specifies the date and time at or after which logging shall be disabled by this property. If this property contains an unspecified datetime, then the conditions for logging to be disabled by Stop_Time shall be ignored. If Stop_Time specifies a date and time earlier than Start_Time, then logging shall be disabled. This property shall be writable if present.

When Stop_Time is reached, the value of the Enable property is not changed.

12.30.11 Log_DeviceObjectProperty

This property, of type BACnetARRAY of BACnetDeviceObjectPropertyReference, specifies the properties to be logged.

If this property is writable, it may be restricted to reference-only objects inside the device containing the Trend Log Multiple object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

Elements of the Log_DeviceObjectProperty array containing object or device instance numbers equal to 4194303 are considered to be 'empty' or 'uninitialized'. An error shall be written to log record entries corresponding to these empty or uninitialized array elements, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NO_PROPERTY_SPECIFIED.

If this property is changed, one of the following actions shall be taken:

Either the log buffer shall be purged, Records_Since_Notification shall be reset to zero, and a BUFFER_PURGED log-status log record shall be recorded as the initial entry regardless of the state of the log buffer before the change, or

Log data values corresponding to changed elements of the Log_DeviceObjectProperty array shall be purged by replacing the relevant log data values in each log record with errors, specifying an 'Error Class' of PROPERTY and an 'Error Code' of LOGGED_VALUE_PURGED.

The selection of which action to take is a local matter.

12.30.12 Logging_Type

This property, of type BACnetLoggingType, specifies whether the Trend Log Multiple object collects log records using polling or triggered acquisition.

COV Logging is not allowed for a Trend Log Multiple object. If this property is writable, an attempt to write the value COV into this property shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the value POLLED is written to this property when the value of Log_Interval is zero, the Log_Interval property shall be updated with an appropriate default polling interval. Determination of the appropriate default polling interval is a local matter.

If the value TRIGGERED is written to this property when Log_Interval has a non-zero value, the Log_Interval property shall be updated with the value zero.

12.30.13 Log_Interval

This property, of type Unsigned, specifies the periodic interval in hundredths of seconds for which the referenced properties are to be logged when Logging_Type has the value POLLED. If Logging_Type has the value TRIGGERED, then the value of this property shall be zero and ignored.

This property shall be writable if Logging_Type has the value POLLED, and shall be read-only if Logging_Type has the value TRIGGERED.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30.14 Align_Intervals

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) clock-aligned periodic logging is enabled. If clock-aligned periodic logging is enabled and the value of Log_Interval is a factor of (i.e. it divides into without a remainder) a second, minute, hour or day, then the beginning of the period specified for logging shall be aligned to the second, minute, hour or day, respectively.

This property has no effect on the behavior of the Trend Log Multiple object if the Logging_Type property has a value other than POLLED.

12.30.15 Interval_Offset

This property, of type Unsigned, specifies the offset in hundredths of seconds from the beginning of the period specified for logging until the actual acquisition of a log record begins. The offset used shall be the value of Interval_Offset modulo the value of Log_Interval; i.e. if Interval_Offset has the value 31 and Log_Interval is 30, the offset used shall be 1. Interval_Offset shall have no effect if Align_Intervals is FALSE.

12.30.16 Trigger

This property, of type BOOLEAN, shall cause the Trend Log Multiple object to acquire a log record whenever the value of this property is changed from FALSE to TRUE. It shall remain TRUE while the Trend Log Multiple object is acquiring the data items for a log record. When all log data items have been collected or it has been determined that all outstanding data requests will not be fulfilled, the Trend Log Multiple object shall reset the value to FALSE.

If the value of the Logging_Type property is not TRIGGERED and an attempt is made to write the value TRUE to the Trigger property, it is left as a local matter whether to execute the logging operation or not. For devices that choose not to execute triggered logging when Logging_Type is not equal to TRIGGERED, attempts to write the value TRUE to this property when Logging_Type has a value other than TRIGGERED shall cause a Result(-) response to be issued, specifying an 'Error Class' of PROPERTY and an 'Error Code' of NOT_CONFIGURED_FOR_TRIGGERED_LOGGING.

Writing to the Trigger property is not restricted to network visible write operations; internal processes may control the acquisition of samples by writing to this property.

12.30.17 Stop_When_Full

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the log buffer is full. When logging ceases because the addition of one more log record would cause the log buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest log record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

If Stop_When_Full is TRUE and Buffer_Size is 2^{32}-1 and the addition of a new record fails due to a lack of space, then the oldest Log_Buffer record shall be discarded, the Enable property shall be set to FALSE and the event is recorded.

12.30.18 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter. If all records are deleted when the Buffer_Size is written then the object shall act as if the Record_Count was set to zero.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

12.30.19 Log_Buffer

This property, of type BACnetLIST of BACnetLogMultipleRecord, is a list of log records where each log record conveys either a set of recorded data values or errors related to data-collection, a status change in the Trend Log Multiple object, or an indication that the time and/or date was changed in the device hosting the Trend Log Multiple object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each log record has data fields as follows:

Timestamp The local date and time when the logging object attempts to aquire the record.

LogDataA set of recorded data values or errors related to data-collection, a status change in the Trend Log Multiple object itself, or an indication that the time and/or date was changed in the device hosting the Trend Log Multiple object.

The choices available for LogData are listed below:

log-status — This choice represents a change in the status or operation of the Trend Log Multiple object. Whenever one of the events represented by the flags listed below occurs, a log record shall be appended to the log buffer.

- LOG_DISABLED — This flag is changed whenever collection of log records by the Trend Log Multiple object is enabled or disabled. It shall be TRUE if Enable is FALSE, or the local time is outside the range defined by Start_Time and Stop_Time, or the addition of this log record will cause the log buffer to be full and Stop_When_Full is TRUE; otherwise it shall be FALSE.
- BUFFER_PURGED — This flag shall be set to TRUE whenever the log buffer is cleared by writing zero to the Record_Count property, or due to a change to the Log_DeviceObjectProperty property. After this value is recorded in the Log buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the log buffer shall be placed into the log buffer even if logging is disabled, the local time is outside of the time range defined by the Start_Time and Stop_Time properties, or the log buffer was completely empty before the request for clearing.
- LOG_INTERRUPTED — This flag indicates that the collection of log records by the Trend Log Multiple object was interrupted by a power failure, device reset, object reconfiguration or other such disruption, such that samples prior to this log record might have been missed.

log-data — The set of logged values. The order of logged values shall correspond to the order of the Log_DeviceObjectProperty array.

- boolean-value, real-value, enumerated-value, unsigned-value, integer-value, bitstring-value, null-value — These choices represent the data values and datatypes read from the monitored object and property.
- any-value — This choice represents the data values and datatypes read from the monitored object and property that are not specifically listed in LogDatum.
- failure — This choice indicates either that an entry in the Log_DeviceObjectProperty array contains an object or device instance equal to 4194303, that a previously logged value was purged, or that an error was encountered in an attempt to read a data value from the monitored object. If the error is conveyed by an error response from a remote device, the Error Class and Error Code in the response shall be recorded.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

time-change — This choice represents a change in the clock setting in the device; it records the number of seconds and fraction of a second by which the clock changed. If, and only if, the number is not known, such as when the clock is initialized for the first time, the value recorded shall be zero. This record shall be recorded after changing the local time of the device and the timestamp shall reflect the new local time of the device.

Also associated with each log record is an implied log record number, the value of which is equal to Total_Record_Count at the point where the log record has been added into the log buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the log buffer is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one.

The log buffer is not network accessible except through the use of the ReadRange service, in order to avoid problems with record sequencing when segmentation is required.

If the object supports event reporting, then a reference to this property shall be the pLogBuffer parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.30.20 Record_Count

This property, of type Unsigned32, shall represent the number of log records currently resident in the log buffer. A write of the value zero to this property shall cause all log records in the log buffer to be deleted and Records_Since_Notification to be reset to zero. Upon completion, this event shall be reported as a BUFFER_PURGED log-status log record in the log buffer as the initial entry regardless of the state of the log buffer before the write.

12.30.21 Total_Record_Count

This property, of type Unsigned32, shall represent the total number of log records collected by the Trend Log Multiple object since creation. When the value of Total_Record_Count reaches its maximum possible value of 2^{32}-1, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost but its use in generating notifications remains.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.30.22 Notification_Threshold

This property, of type Unsigned32, is the pThreshold parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.30.23 Records_Since_Notification

This property, of type Unsigned32, represents the number of log records collected since the previous notification, or since the beginning of logging if no previous notification has occurred.

12.30.24 Last_Notify_Record

This property, of type Unsigned32, is the pPreviousCount parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.30.25 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.30.26 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30.27 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.30.28 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.30.29 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.30.30 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.30.31 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.30.32 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.30.33 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.30.34 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.30.35 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.30.36 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.30.37 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.30.38 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.30.39 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.30.40 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.30.41 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.31 Access Point Object Type

The Access Point object type defines a standardized object whose properties represent the externally visible characteristics associated with the authentication and authorization process of an access controlled point. (e.g., door, gate, turnstile). Access through this point is directional in that it represents access in one direction only. A door, in which access is controlled in both directions, is represented by two separate Access Point objects.

Authentication is the process of verifying the identity of an access user requesting access through an access-controlled door. This can be an authentication policy as simple as a single-factor authentication, in which one authentication factor (i.e. magnetic-stripe card, proximity-card, smart card) is used to identify a known user. In a multi-factor authentication policy, a combination of two or more authentication factors (e.g., card + PIN, card + biometric) are used to verify the identity of the access user. The Access Point object supports the definition of single-factor and multi-factor authentication policies, including the functionality to switch the policy in effect. On false attempts to authenticate, the Access Point may lock-down, for an infinite or specific amount of time.

Authorization is the process of determining whether the access user is permitted to access the zone that he or she has requested to enter. Once the access user has been authenticated successfully, a list of criteria is checked to determine whether access can be granted. If one or more of the authorization criteria fail, then the access user is denied access. Once the access user is granted access, the door will be commanded unlocked at the specified command priority and the access user can access the zone. The door which is controlled is specified in the Access Point. Authorization criteria supported by the Access Point are authorization modes, occupancy enforcement, external verification and threat level.

Authentication and authorization begins when an access user presents an authentication factor at the access controlled point. The process this object represents consumes the authentication factors from the corresponding Credential Data Input objects and performs the authentication and authorization functions. The result is to grant or deny access and to generate corresponding access events. If the object is out of service or not reliable, then these functions are not performed.

The Access Point object generates access events. Access events are stateless (i.e. NORMAL to NORMAL transitions only). A single access transaction, such as a request to enter or an operator action, can result in one or more access events. All access events that belong to the same access transaction have the same access event tag.

For intrinsic reporting, the ACCESS_EVENT algorithm is applied for both Access Alarm Events and Access Transaction Events:

Access Alarm Events: These are events requiring operator attention and handling, and the Access Point object may request human operator acknowledgment of these events.

Access Transaction Events: These are events that are to be logged, not requiring immediate operator attention. The Access Point object does not request human operator acknowledgment of these events.

Access Points which authorize entrance to an access controlled zone are entry points of that zone. Access Points which authorize exit from an access controlled zone are exit points of that zone. In the typical case a specific Access Point is an exit point from one zone and an entry point to an adjacent zone. If the Access Point leads from an Access Zone to no zone (i.e. outside), then the Access Point is an exit point only. If the Access Point leads from no zone (i.e. outside) to an Access Zone, then the Access Point is an entry point only. If the Access Point does not lead from or to an Access Zone (e.g., internal check point or muster point), then the Access Point is neither an entry nor an exit point.

Table 12-36. Properties of the Access Point Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Authentication_Status	BACnetAuthenticationStatus	R
Active_Authentication_Policy	Unsigned	R
Number_Of_Authentication_Policies	Unsigned	R
Authentication_Policy_List	BACnetARRAY[N] of BACnetAuthenticationPolicy	O[1]
Authentication_Policy_Names	BACnetARRAY[N] of CharacterString	O[1]
Authorization_Mode	BACnetAuthorizationMode	R
Verification_Time	Unsigned	O
Lockout	BOOLEAN	O[2]
Lockout_Relinquish_Time	Unsigned	O
Failed_Attempts	Unsigned	O
Failed_Attempt_Events	BACnetLIST of BACnetAccessEvent	O
Max_Failed_Attempts	Unsigned	O[3]
Failed_Attempts_Time	Unsigned	O[3]
Threat_Level	BACnetAccessThreatLevel	O
Occupancy_Upper_Limit_Enforced	BOOLEAN	O
Occupancy_Lower_Limit_Enforced	BOOLEAN	O
Occupancy_Count_Adjust	BOOLEAN	O
Accompaniment_Time	Unsigned	O
Access_Event	BACnetAccessEvent	R
Access_Event_Tag	Unsigned	R
Access_Event_Time	BACnetTimeStamp	R
Access_Event_Credential	BACnetDeviceObjectReference	R
Access_Event_Authentication_Factor	BACnetAuthenticationFactor	O
Access_Doors	BACnetARRAY[N] of BACnetDeviceObjectReference	R
Priority_For_Writing	Unsigned (1..16)	R
Muster_Point	BOOLEAN	O
Zone_To	BACnetDeviceObjectReference	O
Zone_From	BACnetDeviceObjectReference	O
Notification_Class	Unsigned	O[4,6]
Transaction_Notification_Class	Unsigned	O
Access_Alarm_Events	BACnetLIST of BACnetAccessEvent	O[4,6]
Access_Transaction_Events	BACnetLIST of BACnetAccessEvent	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[8]
Auditable_Operations	BACnetAuditOperationFlags	O[8]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] The size of this array shall equal the value of the Number_Of_Authentication_Policies property.
[2] This property is required to be present if Lockout_Relinquish_Time is present.
[3] If this property is present, then the Failed_Attempts property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.31.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.31.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.31.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_POINT.

12.31.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.31.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Access Point. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the Access Point has been overridden by some mechanism local to the BACnet device. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.31.6 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.31.7 Reliability

This property, of type BACnetReliability, provides an indication of whether the authentication and authorization process this object represents is "reliable" as far as the BACnet device can determine and, if not, why.

If Reliability has a value other than NO_FAULT_DETECTED, the process that this object represents shall not perform any authentication or authorization. No access events are generated in this case.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.31.8 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the authentication and authorization process this object represents is out of service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When Out_Of_Service is TRUE:

(a) the process that this object represents shall not perform any authentication or authorization;
(b) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the associated process this object represents;
(c) the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Reliability property shall respond to changes made to this property, as if those changes had occurred to the associated process this object represents.

When this property changes from FALSE to TRUE, then the Access_Event property shall be set to OUT_OF_SERVICE. When this property changes from TRUE to FALSE, then the Access_Event property shall be set to OUT_OF_SERVICE_RELINQUISHED.

12.31.9 Authentication_Status

This property, of type BACnetAuthenticationStatus, shall indicate the current status of the authentication process. This is an enumeration with the following status values:

NOT_READY	The authentication process is not ready to perform a new authentication. This indicates a temporary condition due to processing of the current authentication factor, initialization during startup or other internal processing.
READY	The authentication process is ready to start a new authentication
DISABLED	The authentication process has been disabled. The property shall take on this status when the Out_Of_Service property is TRUE.
WAITING_FOR_AUTHENTICATION_FACTOR	The authentication process is waiting for an additional authentication factor for a multi-factor authentication.
WAITING_FOR_ACCOMPANIMENT	The authentication process is waiting for the authentication of the accompanying credential.
WAITING_FOR_VERIFICATION	The authentication process is waiting for the verification of the credential by an external process.
IN_PROGRESS	The authentication process is currently performing an authentication.

12.31.10 Active_Authentication_Policy

This property, of type Unsigned, shall specify the active authentication policy. The active authentication policy of this object shall be one of 'n' authentication policies, where 'n' is the number of authentication policies defined in the Number_Of_Authentication_Policies property.

The value of the Number_Of_Authentication_Policies property specifies the maximum value that can be written to Active_Authentication_Policy. If a value is written greater than the maximum value or specifies an index of an invalid entry in the Authentication_Policy_List property, if present, then the write attempt is denied and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

If the Authentication_Policy_List property is present, then the value of Active_Authentication_Policy property is an array index that corresponds to the authentication policy as specified in the Authentication_Policy_List property. If no valid authentication policy exists or the current active authentication policy is not valid (i.e. the Authentication_Policy_List entry is empty or not well formed), then this property shall take a value of zero. If the value of the Number_Of_Authentication_Policies property becomes less than the value of this property, then this property shall take a value of zero.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property has the value of zero, then the Reliability property shall have the value CONFIGURATION_ERROR.

12.31.11 Number_Of_Authentication_Policies

This property, of type Unsigned, shall specify the number of specified authentication policies. This property shall always have a value greater than zero. If the value of this property is changed, the size of the Authentication_Policy_List array and the size of the Authentication_Policy_Names array, if present, shall also be changed to the same value.

12.31.12 Authentication_Policy_List

This property, of type BACnetARRAY of BACnetAuthenticationPolicy, specifies the authentication policies defined for this Access Point.

Each element in the array defines an authentication policy for this access point. The BACnetAuthenticationPolicy structure contains the following fields:

Policy — This field is a list of sequence elements that specifies the Credential Data Input objects which are used for this authentication policy. Each element of the list has the following fields:

- Credential-Data-Input — This field, of type BACnetDeviceObjectReference, contains a reference to an object of type Credential Data Input where the authentication factor value is read from the physical device.
- Index — This field, of type Unsigned, indicates the order in which the authentication process expects to receive authentication factors from Credential Data Input objects. The value shall start with the value 1 and continue in increasing sequence.
  If two or more entries of the Policy list have the same index value this indicates that there is a choice between any of the authentication factors supported by the Credential Data Input objects referenced by these entries. In this case the user may present any one of these authentication factors.

Order-Enforced — If TRUE, then the ordering sequence, as specified by the Index fields of this Policy list, is enforced. If FALSE, then the order is not enforced.

Timeout — This field, of type Unsigned, specifies the maximum time in seconds for which all authentication factors, as defined by this policy, must be presented. A value of zero indicates an unlimited time to present all authentication factors. If not all authentication factors are presented in the allotted time, then a timeout occurs and authentication fails.

An Authentication_Policy_List array element shall be considered invalid if the Policy field is empty or if it is not well formed. In this case, the Reliability property shall have the value CONFIGURATION_ERROR. If a write to the Authentication_Policy_List property would cause an array element to be invalid, then a Result (-) shall be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If this property is not present, then the authentication policies are a local matter.

The size of this array shall equal the value of the Number_Of_Authentication_Policies property.

12.31.12.1 Reading Authentication Factors

The authentication factors to be read are determined by the current authentication policy in effect. If the Authentication_Policy_List property is present, the authentication policy is explicitly defined and specifies which Credential Data Input objects the authentication factors are read from. When any authentication factor is read, then the Access_Event property shall be set to AUTHENTICATION_FACTOR_READ.

If the authentication factor read does not match any known authentication factor or the authentication factor read has an error (i.e. has a format type of ERROR), then authentication shall fail and access shall not be granted. In the case where the authentication factor is unknown the Access Event property shall be set to DENIED_UNKNOWN_CREDENTIAL. In the case where the authentication factor has an error, then the Access_Event property shall be set to DENIED_AUTHENTICATION_FACTOR_ERROR.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

It may be possible with certain credential readers to signal a duress code when reading an authentication factor. Determining when a duress code has been read is a local matter. In this case the Access_Event property shall be set to DURESS.

12.31.12.1.1 Single-Factor Authentication

In single-factor authentication, only one authentication factor is required to identify and authenticate the access credential. Depending on the current authentication policy, the access user may have a choice of multiple credentials to use.

12.31.12.1.2 Multi-Factor Authentication

In multi-factor authentication, two or more authentication factors are used for authentication. Typically when multiple factors are used, the first authentication factor is used to identify the credential while the subsequent authentication factors are used to validate the identity. All authentication factors of a multi-factor authentication are expected to be configured in the same Access Credential object.

If a timeout is specified for the current authentication policy and not all authentication factors are read within that time, then authentication shall fail and access shall not be granted. In this case the Access_Event property is set to DENIED_AUTHENTICATION_FACTOR_TIMEOUT.

If one of the authentication factors presented is not the value expected or is presented in an incorrect order, then it is a local matter as to whether authentication fails immediately or whether the access user is given subsequent chances to present the correct authentication factor. If authentication fails immediately, then the Access_Event property shall be set to DENIED_INCORRECT_AUTHENTICATION_FACTOR. If the access user is allowed subsequent chances but fails to present the correct authentication factor within a certain number of attempts, then the Access_Event property shall be set to DENIED_MAX_ATTEMPTS. The maximum number of attempts the access user is allowed is a local matter.

12.31.12.1.3 External Authentication

If the authentication decision is made by an external process, such as a remote server, it may be possible that the authentication process becomes unavailable. When this occurs and when there is no secondary authentication process available, then the authentication shall fail and the Access_Event property shall be set to DENIED_AUTHENTICATION_UNAVAILABLE.

12.31.12.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Authentication_Policy_List array is increased without initial values being provided, then the new array elements for which no initial value is provided shall be initialized with an empty Policy list, Order-Enforced shall be FALSE, and Timeout shall have the value zero.

12.31.13 Authentication_Policy_Names

This property, of type BACnetARRAY of CharacterString, specifies the names of the defined authentication policies.

The size of this array shall equal the value of the Number_Of_Authentication_Policies property

12.31.14 Authorization_Mode

This property, of type BACnetAuthorizationMode, determines how authorization is performed at the Access Point. An Access Point object is not required to support all of these authorization modes but is required to support at least AUTHORIZE.

AUTHORIZE	The access rights of an active credential are evaluated, in addition to other possible authorization checks. If a credential has the value ACCESS_RIGHTS in the Authorization_Exemptions property, then access is granted unless other authorizations checks fail.
GRANT_ACTIVE	An active credential is granted access without evaluating the access rights assigned to the credential. Other authorization checks can still lead to denying access.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DENY_ALL	All credentials are denied access and the Access_Event property is set to DENIED_DENY_ALL. If a credential has the value DENY in the Authorization_Exemptions property, then access is granted unless other authorizations checks fail.
VERIFICATION_REQUIRED	The access rights of an active credential are evaluated, unless an ACCESS_RIGHTS exemption exists for this credential, in addition to other possible authorization checks. Granting access requires external verification. In this case the Access_Event property is set to VERIFICATION_REQUIRED and the access point waits for the external verification. The external verification process and the mechanism by which the verification result is provided to the access point is a local matter. If the external verification process denies access, then the Access_Event property shall be set to DENIED_VERIFICATION_FAILED. If there is no external verification result within the time specified by the Verification_Time property, then the Access_Event property shall be set to DENIED_VERIFICATION_TIMEOUT. If the credential has a VERIFICATION exemption, then the external verification step shall be omitted.
AUTHORIZATION_DELAYED	The access rights of an active credential are evaluated, unless an ACCESS_RIGHTS exemption exists for this credential, in addition to other possible authorization checks. Granting access is delayed by the time specified by the Verification_Time property. This provides an external verification process the opportunity to deny access. In this case the Access_Event property is set to AUTHORIZATION_DELAYED and the access point waits for the external verification. The external verification process and the mechanism by which the verification result is provided to the access point is a local matter. If the external verification process denies access within the time specified in the Verification_Time property, then the Access_Event property shall be set to DENIED_VERIFICATION_FAILED. If there is no external verification result within the time specified by the Verification_Time property, then this authorization check succeeded. If the credential has an AUTHORIZATION_DELAY exemption, then the authorization delay step shall be omitted.
NONE	No authorization functionality takes place at this access point and no authorization events (e.g., grant or any deny events) are generated. This may be used to implement special access point functionality, such as a guard tour or muster point, where authorization checks are not required.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary authorization modes other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.31.14.1 Authorization Decision

Authorization is the process of determining whether or not the credential that has been used to request access at an access point will be permitted access. The authorization process completes when the access decision to grant or deny has been reached. Typically there are multiple authorization criteria used to determine if access will be granted. Examples of authorization criteria are checking access rights, checking passback violations, checking threat level,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

checking occupancy limits, etc. It is a local matter as to what order the authorization criteria are checked. The authorization criteria used to determine whether access is allowed may change according to the time of day, the location and the credential used.

If all authorization criteria are successful, then the credential is granted access at the access point. In this case, the Access_Event property shall be set to GRANTED.

If even one authorization check fails, then access is denied and the Access_Event property is set to the appropriate deny event. If there is no access event, either defined or proprietary, which is specific to the actual reason why access was denied, then the Access_Event property shall be set to DENIED_OTHER.

Access may be denied if the authorization process detects an inconsistency with the access request, such as when an access user requests access to a zone but there is no record of that access user being in the building. How the inconsistency is determined is a local matter. In this case access is denied and the Access_Event property shall be set to DENIED_UNEXPECTED_LOCATION_USAGE.

12.31.15 Verification_Time

This property, of type Unsigned, shall specify the time, in seconds, to wait for external verification when the Authorization_Mode property has a value of AUTHORIZATION_DELAYED or VERIFICATION_REQUIRED.

12.31.16 Lockout

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the access controlled point this object represents is in a lockout state. When the access point is in a lockout state, any access request shall always be denied, except for an active credential for which the value LOCKOUT is contained in the Authorization_Exemptions property of the corresponding Access Credential object. For each denied access request, the Access_Event property shall be set to DENIED_LOCKOUT. An Access Point object may be set to a lockout state due to too many failed access attempts, as defined in the Max_Failed_Attempts property, or by writing TRUE to this property.

When the property Lockout becomes TRUE due to too many failed access attempts, then the Access_Event property shall be set to LOCKOUT_MAX_ATTEMPTS. If TRUE is written to this property for any other reason, the Access_Event property shall be set to LOCKOUT_OTHER. When the Lockout property becomes FALSE, the Access_Event property shall be set to LOCKOUT_RELINQUISHED.

If the Lockout property is present, then the Lockout_Relinquish_Time property shall also be present.

12.31.17 Lockout_Relinquish_Time

This property, of type Unsigned, shall specify the time, in seconds, to delay after the Lockout property has taken on the value TRUE, before automatically relinquishing the lockout state. The lockout state is relinquished by setting the Lockout property to FALSE. A value of zero indicates that the lockout state will not automatically be relinquished.

If the Lockout_Relinquish_Time is present, then the Lockout property shall also be present.

12.31.18 Failed_Attempts

This property, of type Unsigned, shall indicate the current count of successive failed access attempts. Any successive failed access attempt shall increment the value of this property.

This property shall be set to zero when a successful access attempt occurs or when the property Lockout becomes FALSE.

12.31.19 Failed_Attempt_Events

This property, of type BACnetLIST of BACnetAccessEvent, specifies those access events that are counted as a failed access attempt.

If this property is not present, it is a local matter as to which access events are considered a failed attempt.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.31.20 Max_Failed_Attempts

This property, of type Unsigned, shall specify the maximum number of successive failed access attempts before the Lockout property is set to TRUE. If the Failed_Attempts property becomes greater than or equal to the value of this property and this property is not zero, the Lockout property is set to TRUE. Zero indicates that the Lockout property is not set to TRUE as the result of failed access attempts.

If the Max_Failed_Attempts property is present, then the Failed_Attempts property shall also be present.

12.31.21 Failed_Attempts_Time

This property, of type Unsigned, shall specify the time, in seconds, to delay before setting the Failed_Attempts property to zero, after the last failed access attempt.

If the Failed_Attempts_Time is present, then the Failed_Attempts property shall also be present.

12.31.22 Threat_Level

This property, of type BACnetAccessThreatLevel, shall specify the current threat level for this Access Point. Zero is the lowest threat level, effectively disabling the threat level check, while 100 is the maximum threat level. If the threat authority of the authenticated credential is lower than the value of this property, then the authorization fails. In this case the Access_Event property shall be set to DENIED_THREAT_LEVEL.

12.31.23 Occupancy_Upper_Limit_Enforced

This property, of type BOOLEAN, indicates whether the upper occupancy limit of the access controlled zone, for which this object is an entry point, is enforced (TRUE) or not (FALSE). If enforced, authorization shall fail if the access controlled zone's occupancy is greater than or equal to its upper occupancy limit, unless the credential is exempted from this authorization check. When this authorization check fails, the Access_Event property shall be set to DENIED_UPPER_OCCUPANCY_LIMIT.

12.31.24 Occupancy_Lower_Limit_Enforced

This property, of type BOOLEAN, indicates whether the lower occupancy limit of the access controlled zone, for which this object is an exit point, is enforced (TRUE) or not (FALSE). If enforced, authorization shall fail if the access controlled zone's occupancy is lower than or equal to its lower occupancy limit, unless the credential is exempted from this authorization check. When this authorization check fails, the Access_Event property shall be set to DENIED_LOWER_OCCUPANCY_LIMIT.

12.31.25 Occupancy_Count_Adjust

This property, of type BOOLEAN, indicates whether (TRUE) this object will adjust the occupancy count of the zones for which it controls access, or not (FALSE). The occupancy count is decremented for the zone for which this Access Point is an exit point and incremented for the zone for which this Access Point is an entry point.

Occupancy count shall be adjusted if the credential holder passes through the access point. How this is determined is a local matter. The occupancy count of the zones is adjusted by writing a negative amount to the Adjust_Value property of the exit access zone and the corresponding positive amount to the Adjust_Value property of the entry access zone.

If this property is not supported, then the Access Point object behaves as if the value is FALSE.

12.31.26 Accompaniment_Time

This property, of type Unsigned, shall specify the time, in seconds, to wait for a second credential to be presented at this access point when the original credential requires accompaniment. If an accompanying credential is not presented within this time the authorization of the original credential shall fail and the Access_Event property shall be set to DENIED_NO_ACCOMPANIMENT.

12.31.27 Access_Event

This property, of type BACnetAccessEvent, indicates the last access event which occurred at this Access Point. This property is the pMonitoredValue parameter for the object's ACCESS_EVENT event algorithm for both Access Alarm Events and Access Transaction Events. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnetAccessEvent is an enumeration of authentication and authorization decisions, subsequent actions, and results of these actions. An Access Point is not required to support all values of this enumeration.

NONE	The access point did not yet determine any access event or the object is not generating access events. This is not a reported event. It is required to enable algorithmic ACCESS_EVENT reporting.
GRANTED	Access granted to the presented credential.
MUSTER	If the access point is a muster point, a muster event is generated when a credential is presented.
PASSBACK_DETECTED	A passback violation for the presented credential has been detected.
DURESS	A duress incident was detected at this access point.
TRACE	A traced credential has been presented.
LOCKOUT_MAX_ATTEMPTS	The access point is in a lockout state due to maximum failed access attempts.
LOCKOUT_OTHER	The access point is in a lockout state due to any reason other than maximum failed access attempts.
LOCKOUT_RELINQUISHED	The access point has relinquished the lockout state.
LOCKED_BY_HIGHER_PRIORITY	The controlled Access Door object is commanded at a higher priority.
OUT_OF_SERVICE	The Out_Of_Service flag of the Access Point object has been set to TRUE.
OUT_OF_SERVICE_RELINQUISHED	The Out_Of_Service flag of the Access Point object has been set to FALSE.
ACCOMPANIMENT_BY	The credential presented accompanies the previous credential.
AUTHENTICATION_FACTOR_READ	An authentication factor has been read. This event indicates a successful read of an authentication factor in single-factor or multi-factor authentication.
AUTHORIZATION_DELAYED	Authorization of a credential is delayed to allow time for an external process to deny access.
VERIFICATION_REQUIRED	Authorization of a credential requires verification from an external process.
NO_ENTRY_AFTER_GRANTED	Access was granted to the presented credential but the physical door was not opened.
DENIED_DENY_ALL	Access denied because the Authorization_Mode property of the Access Point object is set to DENY_ALL.
DENIED_UNKNOWN_CREDENTIAL	Access denied due to an unknown credential. The authentication factor presented did not match any known authentication factor.
DENIED_AUTHENTICATION_UNAVAILABLE	Access denied because the authentication and authorization decision is unavailable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DENIED_AUTHENTICATION_FACTOR_TIMEOUT	Access denied due to required authentication factor for multi-factor authentication not being presented within time.
DENIED_INCORRECT_AUTHENTICATION_FACTOR	Access denied due to the authentication factor presented for a multi-factor-authentication not being the one expected.
DENIED_POINT_NO_ACCESS_RIGHTS	Access denied due to evaluation of TRUE of a negative access rule for the access point.
DENIED_ZONE_NO_ACCESS_RIGHTS	Access denied due to evaluation of TRUE of a negative access rule for the access zone.
DENIED_NO_ACCESS_RIGHTS	Access denied due to no positive access rule being found for the access zone or access point.
DENIED_OUT_OF_TIME_RANGE	Access denied due to the presented credential not being valid at this access point or access zone at this time.
DENIED_THREAT_LEVEL	Access denied due to insufficient threat authority for the presented credential.
DENIED_PASSBACK	Access denied due to a passback violation for the presented credential.
DENIED_UNEXPECTED_LOCATION_USAGE	Access denied due to the credential being used at a location which violates local consistency rules.
DENIED_MAX_ATTEMPTS	Access denied due to too many failed multi-factor access attempts at the access point.
DENIED_LOWER_OCCUPANCY_LIMIT	Exit from a zone for which this access point is an exit access point is denied due to zone occupancy being below or at the minimum limit.
DENIED_UPPER_OCCUPANCY_LIMIT	Access to a zone for which this access point is an entry access point is denied due to zone occupancy being at or above the maximum limit.
DENIED_AUTHENTICATION_FACTOR_LOST	Access denied due to the authentication factor used being reported as lost.
DENIED_AUTHENTICATION_FACTOR_STOLEN	Access denied due to the authentication factor used being reported as stolen.
DENIED_AUTHENTICATION_FACTOR_DAMAGED	Access denied due to the authentication factor used being reported as damaged.
DENIED_AUTHENTICATION_FACTOR_DESTROYED	Access denied due to the authentication factor used being reported as destroyed.
DENIED_AUTHENTICATION_FACTOR_DISABLED	Access denied due to the authentication factor used being disabled for unspecified or unknown reasons.
DENIED_AUTHENTICATION_FACTOR_ERROR	Access denied due to the authentication factor used having a read error.
DENIED_CREDENTIAL_UNASSIGNED	Access denied due to the credential used not having yet been assigned to an access user.
DENIED_CREDENTIAL_NOT_PROVISONED	Access denied due to the credential used not yet having been provisioned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DENIED_CREDENTIAL_NOT_YET_ACTIVE	Access denied due to the credential used not yet being active.
DENIED_CREDENTIAL_EXPIRED	Access denied due to the credential used having expired.
DENIED_CREDENTIAL_MANUAL_DISABLE	Access denied due to the credential used having been manually disabled.
DENIED_CREDENTIAL_LOCKOUT	Access denied due to the credential used being locked out.
DENIED_CREDENTIAL_MAX_DAYS	Access denied due to the number of days the credential may be used having been exceeded.
DENIED_CREDENTIAL_MAX_USES	Access denied due to the number of allowed uses of the credential used has been exceeded.
DENIED_CREDENTIAL_INACTIVITY	Access denied due to the credential used being disabled after a period of inactivity.
DENIED_CREDENTIAL_DISABLED	Access denied due to the credential used being disabled for unspecified or unknown reasons.
DENIED_NO_ACCOMPANIMENT	Access denied due to the expected accompanying credential not being presented.
DENIED_INCORRECT_ACCOMPANIMENT	Access denied due to the accompanying credential presented being incorrect
DENIED_LOCKOUT	Access denied due to the access point being in lockout state.
DENIED_VERIFICATION_FAILED	Access denied due to an external process denying access when verification was required.
DENIED_VERIFICATION_TIMEOUT	Access denied due to an external process having failed to send a response, in the allotted time, when verification was required.
DENIED_OTHER	Access is denied for unspecified reasons.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to indicate Access Events other than those defined by this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.31.27.1 Operations for setting the Access_Event property

When a new event occurs at the access point, the following series of operations shall be performed atomically:

The value written to Access_Event shall be stored in the Access_Event property,

(1) If this event is the start of a new access transaction, the value of the Access_Event_Tag property shall be incremented.
(2) The current date and time shall be stored in the Access_Event_Time property.
(3) The reference to the Access Credential object that is associated with this event shall be stored in the Access_Event_Credential property. See Clause 12.31.30 for other conditions.
(4) The value of the authentication factor that is associated with this event shall be stored in the Access_Event_Authentication_Factor property. See Clause 12.31.31 for other conditions.

When a single access transaction causes multiple access events to be generated, then the granted event or the denied event shall be the final event generated. If none of the multiple events are granted or denied events, then the resulting events can be generated in any order.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.31.28 Access_Event_Tag

This property, of type Unsigned, is a numeric value which identifies the access transaction to which the current access event belongs. Multiple access events may be generated by a single access transaction.

The value of this property shall increase monotonically for each new access transaction. It may be implemented using modulo arithmetic. The initial value of this property before any access transaction has occurred shall be a local matter.

This property is the pAccessEventTag parameter for the object's ACCESS_EVENT event algorithm for both Access Alarm Events and Access Transaction Events. See Clause 13.3 for event algorithm parameter descriptions.

12.31.29 Access_Event_Time

This property, of type BACnetTimeStamp, indicates the most recent update time of the Access_Event property. This property shall update its value on each update of Access_Event. Update times of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number update times shall have the value 0 if no update has yet occurred.

This property is the pAccessEventTime parameter for the object's ACCESS_EVENT event algorithm for both Access Alarm Events and Access Transaction Events. See Clause 13.3 for event algorithm parameter descriptions.

12.31.30 Access_Event_Credential

This property, of type BACnetDeviceObjectReference, shall specify the Access Credential object that corresponds to the access event specified in the Access_Event property, if applicable.
This property shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present, under the following conditions:

(a) there is no credential recognized up to now, or
(b) there is no credential that is associated to the current access event, or
(c) the credential of the authentication factor that is associated to the current event is unknown, or
(d) the device chooses not to expose the credential.

This property is the pAccessCredential parameter for the object's ACCESS_EVENT event algorithm for both Access Alarm Events and Access Transaction Events. See Clause 13.3 for event algorithm parameter descriptions.

12.31.31 Access_Event_Authentication_Factor

This property, of type BACnetAuthenticationFactor, shall specify the authentication factor that corresponds to the access event specified in the Access_Event property, if applicable. Otherwise it shall contain a value of format type UNDEFINED.

This property shall contain a value of format type UNDEFINED under the following conditions:

(a) there was no authentication factor read up to now, or
(b) there is no authentication factor that is associated to the current access event, or
(c) the device chooses not to expose the authentication factor.

This property is the pAccessFactor parameter for the object's ACCESS_EVENT event algorithm for both Access Alarm Events and Access Transaction Events. See Clause 13.3 for event algorithm parameter descriptions.

12.31.32 Access_Doors

This property, of type BACnetARRAY of BACnetDeviceObjectReference, shall specify the references to those Access Door objects whose Present_Value properties are commanded after successful authorization. If this Access Point object does not command Access Door objects (e.g., muster point), or is used to control access to other resources or functions, or commands other objects, then this array shall be empty.

12.31.32.1 Commanding Access Doors

After successful authorization the following actions occur:

(1) The Access_Event property shall be set to GRANTED, and

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(2) The physical doors, as specified in the Access_Doors property, are commanded with either a PULSE_UNLOCK or EXTENDED_PULSE_UNLOCK door command, at the priority specified by the Priority_For_Writing property.

Commanding the doors may fail due to a higher priority command in effect in the Access Door object. In this case the Access_Event property shall be set to LOCKED_BY_HIGHER_PRIORITY.

The respective Access Doors may be monitored to verify that they are opened and access takes place. If no access takes place, the Access_Event property shall be set to NO_ENTRY_AFTER_GRANTED.

12.31.33 Priority_For_Writing

This property, of type Unsigned, defines the priority at which the referenced properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. The value 1 is considered the highest priority and 16 the lowest. See Clause 19.2.

12.31.34 Muster_Point

This property, of type BOOLEAN, indicates whether this Access Point generates muster access events (TRUE) or not (FALSE).

A muster event is generated by setting the Access_Event property to MUSTER after an access credential has been presented at the access point. It is a local matter as to whether a muster event is generated for unknown credentials.

12.31.35 Zone_To

This property, of type BACnetDeviceObjectReference, shall specify the Access Zone object for which this Access Point object is an entry access controlled point, allowing entrance to the zone. This property shall not reference the same Access Zone object as the Zone_From property. If the Access Point is not an entry point to an access controlled zone, then this property shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.

12.31.36 Zone_From

This property, of type BACnetDeviceObjectReference, shall specify the Access Zone object for which this Access Point object is an exit access controlled point, allowing exit from the zone. This property shall not reference the same Access Zone object as the Zone_To property. If the Access Point is not an exit point from an access controlled zone, then this property shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.

12.31.37 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.31.38 Transaction_Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution of Access Transaction Events. If this property is not present, then the Notification Class specified by the property Notification_Class shall be used for Access Transaction Events.

12.31.39 Access_Alarm_Events

The value of this property is used as the value of the pAccessEvents parameter for the object's ACCESS_EVENT event algorithm for Access Alarm Events.

An Access Alarm Event is reported when the following conditions are true:

(a) The Access_Event is updated and the updated value is equal to one of the values in Access_Alarm_Events, and
(b) the TO_NORMAL flag is enabled in the Event_Enable property.

The notification is sent with Notify Type as specified by the property Notify_Type.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The Notification Class object referenced by the Notification_Class property is used to report Access Alarm Events.

12.31.40 Access_Transaction_Events

The value of this property is used as the value of the pAccessEvents parameter for the object's ACCESS_EVENT event algorithm for Access Transaction Events.

An Access Transaction Event is reported when the following conditions are true:

(a) the Access_Event is updated and the updated value is equal to one of the values in Access_Transaction_Events, and
(b) the TO_NORMAL flag is set in the Event_Enable property.

The value of Notify_Type is ignored and the notification is sent with a Notify Type of EVENT. The Event_Time_Stamps TO_NORMAL element is not affected. The Acked_Transitions TO_NORMAL bit is not affected.

The Notification Class object referenced by the Transaction_Notification_Class property is used to report Access Transaction Events. If Transaction_Notification_Class is not present, the Notification Class object referenced by Notification_Class is used. The Ack_Required property of the respective Notification Class object is ignored and the value FALSE is conveyed in the AckRequired parameter of the event notification message.

12.31.41 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.31.42 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.31.43 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

Access Transaction Events generated by the object are always of type EVENT, regardless of the value of this property.

12.31.44 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.31.45 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.31.46 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.31.47 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.31.48 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.31.49 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.31.50 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.31.51 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.31.52 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.31.53 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.31.54 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.31.55 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.31.56 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.32 Access Zone Object Type

The Access Zone object type defines a standardized object whose properties represent the externally visible characteristics associated with a secured geographical zone for which authentication and authorization of a credential takes place to obtain physical access. Entrance to the zone takes place through entry access controlled points while the zone is exited through exit access controlled points. These access controlled points are represented by Access Point objects.

The Access Zone object may optionally support occupancy counting and the specification of occupancy limits. Access may be denied if limits are violated. The enforcement rules are specified at the corresponding entry and/or exit Access Point objects. The Access Zone object's Occupancy_State is the state of occupancy. This state is derived from the actual occupancy count and occupancy limits.

Intrinsic reporting of this object is based on the Occupancy_State property and uses the CHANGE_OF_STATE algorithm.

"Who's in" reporting is supported through a list of the credentials which are currently in the zone. Credentials are added on successful entrance through an access controlled entry point and removed on successful exit through an access controlled exit point. This list may also be maintained based on time conditions or other local methods.

The Access Zone object supports passback detection and allows the selection of hard, soft or no-passback enforcement. A passback violation occurs when entrance to this zone is requested at an access controlled point while the credential is assumed to be in this zone. The list of credentials in this zone may be used to detect a passback violation.

A specific access controlled zone may be represented by a single Access Zone object in a single device, or in multiple devices by one Access Zone object per device. When an access controlled zone is represented in multiple devices, the representing Access Zone objects may not have the same Object_Identifier in each device; however, they may be identified using the Global_Identifier property. It is a local matter as to how these objects are synchronized.

Table 12-37. Properties of the Access Zone Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Global_Identifier	Unsigned32	W
Occupancy_State	BACnetAccessZoneOccupancyState	R
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R[1]
Out_Of_Service	BOOLEAN	R
Occupancy_Count	Unsigned	O[1,3,4]
Occupancy_Count_Enable	BOOLEAN	O[3,4]
Adjust_Value	INTEGER	O[3,4,5]
Occupancy_Upper_Limit	Unsigned	O
Occupancy_Lower_Limit	Unsigned	O
Credentials_In_Zone	BACnetLIST of BACnetDeviceObjectReference	O
Last_Credential_Added	BACnetDeviceObjectReference	O
Last_Credential_Added_Time	BACnetDateTime	O
Last_Credential_Removed	BACnetDeviceObjectReference	O
Last_Credential_Removed_Time	BACnetDateTime	O
Passback_Mode	BACnetAccessPassbackMode	O
Passback_Timeout	Unsigned	O[2]
Entry_Points	BACnetLIST of BACnetDeviceObjectReference	R
Exit_Points	BACnetLIST of BACnetDeviceObjectReference	R
Time_Delay	Unsigned	O[3,7]
Notification_Class	Unsigned	O[3,7]
Alarm_Values	BACnetLIST of BACnetAccessZoneOccupancyState	O[3,7]
Event_Enable	BACnetEventTransitionBits	O[3,7]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Acked_Transitions	BACnetEventTransitionBits	O[3,7]
Notify_Type	BACnetNotifyType	O[3,7]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,7]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[7]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[7]
Event_Detection_Enable	BOOLEAN	O[3,7]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[7]
Event_Algorithm_Inhibit	BOOLEAN	O[7,8]
Time_Delay_Normal	Unsigned	O[7]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties, if present, shall be writable when Out_Of_Service is TRUE.
[2] If this property is present, then Passback_Mode shall be present.
[3] These properties are required if the object supports intrinsic reporting.
[4] These properties are required if, and shall be present only if, the object supports occupancy counting.
[5] The Adjust_Value property shall be writable if present.
[6] Footnote removed.
[7] These properties shall be present only if the object supports intrinsic reporting.
[8] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[9] This property shall be present only if the device supports audit reporting.

12.32.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.32.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.32.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_ZONE.

12.32.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.32.5 Global_Identifier

This property, of type Unsigned32, is a unique identifier which is used to globally identify the access controlled zone this object represents. This value may be used to identify Access Zone objects in multiple devices that represent the same access controlled zone.

If this value is assigned, it shall be unique internetwork-wide and all Access Zone objects in all devices that represent this access controlled zone shall have this value. A value of zero indicates that no global identifier is assigned.

12.32.6 Occupancy_State

This property, of type BACnetAccessZoneOccupancyState, reflects the occupancy state of the zone.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

BACnetAccessZoneOccupancyState is an enumeration of possible occupancy states.

NORMAL	This is the occupancy state when occupancy counting is enabled and no other standard or proprietary states are applicable.
BELOW_LOWER_LIMIT	If Occupancy_Lower_Limit property is present and the Occupancy_Count property is lower than this value.
AT_LOWER_LIMIT	If Occupancy_Lower_Limit property is present and the Occupancy_Count property is equal to this value.
AT_UPPER_LIMIT	If Occupancy_Upper_Limit property is present and the Occupancy_Count property is equal to this value.
ABOVE_UPPER_LIMIT	If Occupancy_Upper_Limit property is present and the Occupancy_Count property is greater than this value.
DISABLED	This is the occupancy state when occupancy counting is disabled for this object. Occupancy counting is disabled when the Occupancy_Count_Enable property is FALSE.
NOT_SUPPORTED	This is the occupancy state when occupancy counting is not supported by this object.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to indicate other states based on the Occupancy_Count property other than those defined by this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.32.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Access Zone object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.32.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.32.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Occupancy_State, Occupancy_Count and/or Credentials_In_Zone properties of this object are "reliable" as far as the BACnet device can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.32.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the object is out of service.

When Out_Of_Service is TRUE:

(a) the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the associated process this object represents;
(b) the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) if occupancy counting is supported, then the Occupancy_Count property is decoupled from the processing of occupancy counting;
(d) writing to the Adjust_Value property shall not modify the Occupancy_Count;
(e) the Occupancy_Count property, if present, shall be writable to allow simulating specific conditions or for testing purposes;
(f) other functions that depend on the state of the Occupancy_State or Reliability properties shall respond to changes made to these properties, as if those changes had occurred to the associated process this object represents.

12.32.11 Occupancy_Count

This property, of type Unsigned, is used to indicate the actual occupancy count of a zone. If the value of the Occupancy_Count_Enable property is FALSE, then this property shall have a value of zero. The value of the Occupancy_Count property may be adjusted by writing to the Adjust_Value property. The Occupancy_Count property shall be writable when Out_Of_Service is TRUE. When Out_Of_Service becomes FALSE, it is a local matter as to what value this property is set to.

12.32.12 Occupancy_Count_Enable

This property, of type BOOLEAN, indicates whether occupancy counting is enabled (TRUE) or not (FALSE).

If this property has a value of FALSE, then the Occupancy_State property shall have a value of DISABLED.

When this property changes from FALSE to TRUE it is a local matter as to what value the Occupancy_Count property is set to.

12.32.13 Adjust_Value

This property, of type INTEGER, shall adjust the Occupancy_Count property when written.

The following series of operations shall be performed atomically when this property is written and the value of the Occupancy_Count_Enable property is TRUE:

(1) The value written to Adjust_Value shall be stored in the Adjust_Value property.
(2) If the value written is non-zero, then this value shall be added to the value of the Occupancy_Count property. If the value written is negative and the resulting value of the Occupancy_Count property would be less than zero, then the Occupancy_Count property shall be set to zero. If the value written is zero, then the value of the Occupancy_Count property shall be set to zero.

When this property is written and the value of the Occupancy_Count_Enable property is FALSE, then the Adjust_Value property shall be set to zero.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If Adjust_Value has never been written or the Occupancy_Count_Enable property is FALSE, then this property shall have a value of zero.

12.32.14 Occupancy_Upper_Limit

This property, of type Unsigned, specifies the occupancy upper limit of the zone. If this property has a value of zero, then there is no upper limit. If this value is not zero, it shall be greater than the value of the Occupancy_Lower_Limit, if present.

12.32.15 Occupancy_Lower_Limit

This property, of type Unsigned, specifies the occupancy lower limit of the zone. If this property has a value of zero, then there is no lower limit.

12.32.16 Credentials_In_Zone

This property, of type BACnetLIST of BACnetDeviceObjectReference, is used to list references to those Access Credential objects that represent credentials assumed to be in this zone. This information may be used to verify whether a specific credential is already in the zone for passback detection purposes. If the zone does not support listing credentials, then this list, if present, shall be empty. It is a local matter as to how this list is updated.

12.32.17 Last_Credential_Added

This property, of type BACnetDeviceObjectReference, indicates the reference to the Access Credential object which has last been added to the Credentials_In_Zone property. If no credential has been added yet, then this reference shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present. If COV property subscriptions for this property are present, then any update, even one with the same value, is reported by a COV notification.

12.32.18 Last_Credential_Added_Time

This property, of type BACnetDateTime, indicates the date and time when a reference to an Access Credential object has last been added to the Credentials_In_Zone property. If this property is present, but no credential has yet been added, then this property shall not convey an actual time and shall contain a value of X'FF' in all octets.

12.32.19 Last_Credential_Removed

This property, of type BACnetDeviceObjectReference, indicates the reference to the Access Credential object which has last been removed from the Credentials_In_Zone property. If no credential has been removed yet, then this reference shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present. If COV property subscriptions for this property are present, then any update, even one with the same value, is reported by a COV notification.

12.32.20 Last_Credential_Removed_Time

This property, of type BACnetDateTime, indicates the date and time when a reference to an Access Credential object has last been removed from the Credentials_In_Zone property. If this property is present, but no credential has yet been removed, then this property shall not convey an actual time and shall contain a value of X'FF' in all octets.

12.32.21 Passback_Mode

This property, of type BACnetAccessPassbackMode, specifies how all Access Point objects that represent entry points to the access controlled zone this object represents shall handle passback violations. Passback modes are:

PASSBACK_OFF	Passback violations are not checked.
HARD_PASSBACK	Passback violations are checked, enforced and reported. When a passback violation is detected, the Access_Event Property of the corresponding Access Point object shall be set to DENIED_PASSBACK and the authorization for the credential shall fail.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

SOFT_PASSBACK	Passback violations are checked and reported but not enforced. When a passback violation is detected, the Access_Event Property of the corresponding Access Point object shall be set to PASSBACK_DETECTED.

12.32.22 Passback_Timeout

This property, of type Unsigned, specifies the passback timeout in minutes. The timeout is evaluated individually for every credential used to enter the zone. The timeout period for a particular credential begins at the time of successful access to the zone. After the timeout has expired for a particular credential, a passback violation of this credential will no longer be detected. A value of zero or absence of this property indicates passback violations will never time out.

If Passback_Timeout is present, Passback_Mode shall be present.

12.32.23 Entry_Points

This property, of type BACnetLIST of BACnetDeviceObjectReference, references all Access Point objects that lead into the zone.

12.32.24 Exit_Points

This property, of type BACnetLIST of BACnetDeviceObjectReference, references all Access Point objects that lead out of the zone.

12.32.25 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.32.26 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.32.27 Alarm_Values

This property, of type BACnetLIST of BACnetAccessZoneOccupancyState, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.32.28 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.32.29 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.32.30 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.32.31 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.32.32 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.32.33 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.32.34 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.32.35 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.32.36 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.32.37 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.32.38 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.32.39 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.32.40 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.32.41 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.32.42 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.32.43 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.32.44 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.33 Access User Object Type

The Access User object type defines a standardized object whose properties represent the externally visible characteristics associated with a user of a physical access control system.

The Access User object is used to represent an individual person, a group of users, or an asset. Relationships among access users are supported for representation of hierarchical organizations (e.g., companies, departments, or groups of any kind) or for representing ownership of assets.

The Access User object is not directly involved in authentication and authorization. It is used for informational purposes. It can hold a name, a reference number and a reference to an external system providing details of the access user.

The Access User object can have Access Credential objects assigned. This information can be used for administrative purposes (e.g., disabling all credentials of a person).

When a user is represented in multiple devices, the representing Access User objects may not have the same Object_Identifier in each device; however, they may be identified using the Global_Identifier property. It is a local matter as to how these objects are synchronized.

Table 12-38. Properties of the Access User Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Global_Identifier	Unsigned32	W
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
User_Type	BACnetAccessUserType	R
User_Name	CharacterString	O
User_External_Identifier	CharacterString	O
User_Information_Reference	CharacterString	O
Members	BACnetLIST of BACnetDeviceObjectReference	O
Member_Of	BACnetLIST of BACnetDeviceObjectReference	O
Credentials	BACnetLIST of BACnetDeviceObjectReference	R
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.33.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.33.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.33.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_USER.

© ISO 2022 – All rights reserved

12.33.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.33.5 Global_Identifier

This property, of type Unsigned32, is a unique identifier which is used to globally identify the access user this object represents. This value may be used to identify Access User objects in multiple devices which represent the same access user.

If this value is assigned, it shall be unique internetwork-wide and all Access User objects in all devices which represent this access user shall have this value. A value of zero indicates that no global identifier is assigned.

12.33.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	The value of this flag shall be logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

12.33.7 Reliability

This property, of type BACnetReliability, provides an indication of whether this object is "reliable" as far as the BACnet device can determine and, if not, why.

12.33.8 User_Type

This property, of type BACnetAccessUserType, specifies the access user type this object represents. The following user types are defined:

ASSET	The Access User object represents a physical item.
GROUP	The Access User object represents a group of access users.
PERSON	The Access User object represents an individual person.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary access user types other than those defined by this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.33.9 User_Name

This property, of type CharacterString, is a string of printable characters which specifies the name of the access user. The content is not restricted and can contain multiple lines.

12.33.10 User_External_Identifier

This property, of type CharacterString, specifies an external identifier associated with the access user. While the content is typically unique, its interpretation is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.33.11 User_Information_Reference

This property, of type CharacterString, specifies a reference to an external system where additional information of the user can be found. The interpretation of the content is a local matter.

12.33.12 Members

This property, of type BACnetLIST of BACnetDeviceObjectReference, references the Access User objects that represent the associated access users. Each object referenced shall be an Access User object.

12.33.13 Member_Of

This property, of type BACnetLIST of BACnetDeviceObjectReference, references the Access User objects that represent the access users to which this access user is associated. Each object referenced shall be an Access User object.

12.33.14 Credentials

This property, of type BACnetLIST of BACnetDeviceObjectReference, references all Access Credential objects that represent those credentials which are owned by this access user. Each object referenced shall be an Access Credential object.

12.33.15 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.33.16 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.33.17 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.33.18 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.33.19 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.33.20 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.33.21 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.34 Access Rights Object Type

The Access Rights object type defines a standardized object whose properties represent the externally visible characteristics associated with access rights for physical access control.

The Access Rights object is a collection of individual access rule specifications which define privileges for entering and leaving access controlled zones or for accessing other resources or functions. One or many credentials can share this collection of access rules. This object type supports role-based access control models.

The Access Rights object contains a collection of negative and positive access rules. A negative access rule specifies where and when access shall be denied. A positive access rule specifies where and when access may be granted. Negative access rules take precedence over positive access rules. All negative access rules of all Access Rights objects assigned to a credential are evaluated before any positive access rule.

Each access rule, whether positive or negative, specifies the location of access, which is an access controlled point or a zone, a condition which determines whether the rule applies at this time, and a flag which indicates whether the rule is enabled. In the most typical case the condition is determined by evaluating the Present_Value of a Schedule object that specifies time ranges. In addition, each of these access rules can be enabled or disabled individually.

The Access Rights object can specify an accompaniment requirement that defines the access user that owns the accompanying credential, the credential required to accompany, or the access rights required to be assigned to the accompanying credential.

When a specific access rights collection is represented in multiple devices, the representing Access Rights objects may not have the same Object_Identifier in each device; however, they may be identified using the Global_Identifier property. It is a local matter as to how these objects are synchronized.

Table 12-39. Properties of the Access Rights Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Global_Identifier	Unsigned32	W
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Enable	BOOLEAN	R
Negative_Access_Rules	BACnetARRAY[N] of BACnetAccessRule	R
Positive_Access_Rules	BACnetARRAY[N] of BACnetAccessRule	R
Accompaniment	BACnetDeviceObjectReference	O
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property shall be present only if the device supports audit reporting.

12.34.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.34.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.34.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_RIGHTS.

12.34.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.34.5 Global_Identifier

This property, of type Unsigned32, is a unique identifier which is used to globally identify the collection of access rights this object represents. This value may be used to identify Access Rights objects in multiple devices which represent the same collection of access rights.

If this value is assigned, it shall be unique internetwork-wide and all Access Rights objects in all devices which represent this collection of access rights shall have this value. A value of zero indicates that no global identifier is assigned.

12.34.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	The value of this flag shall be logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

12.34.7 Reliability

This property, of type BACnetReliability, provides an indication of whether this object is "reliable" as far as the BACnet device can determine and, if not, why.

12.34.8 Enable

This property, of type BOOLEAN, indicates whether this object is enabled (TRUE) or disabled (FALSE). When this object is disabled all the access rules specified in the Positive_Access_Rules and Negative_Access_Rules properties are disabled.

12.34.9 Negative_Access_Rules

This property, of type BACnetARRAY of BACnetAccessRule, specifies the negative access rules. Each element of the array is evaluated as described in Clause 12.34.9.1.

To determine how access rules are evaluated, see Clause 12.34.9.2.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.34.9.1 Access Rule Specification

An access rule specification is of type BACnetAccessRule. This is a structure with the following fields:

Time-Range-Specifier — This field is an enumeration that specifies the evaluation of the Time-Range field:

- SPECIFIED — Time-Range references a property that will be evaluated to TRUE or FALSE as defined for the Time-Range field.
- ALWAYS — The value of the Time-Range field is ignored and always evaluates to TRUE.

Time-Range — This optional field, of type BACnetDeviceObjectPropertyReference, references a property that can be evaluated to TRUE or FALSE, which defines whether the rule is valid (TRUE) or not (FALSE).

The Time-Range reference shall be considered unspecified if it contains 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.

This field shall be present if Time-Range-Specifier is SPECIFIED. If Time-Range-Specifier is ALWAYS and this field is present, then this reference shall be unspecified.

If Time-Range-Specifier is SPECIFIED and this field references a property of type other than BOOLEAN, then the following evaluations apply:

If the value of the referenced property is of type Unsigned, a value of zero shall evaluate to FALSE, while any other value shall evaluate to TRUE.

If the value of the referenced property is of type INTEGER, a value of less than or equal to zero shall evaluate to FALSE, while any value greater than zero shall evaluate to TRUE.

If the value of the referenced property is of type BACnetBinaryPV, then INACTIVE shall evaluate to FALSE, while ACTIVE evaluates to TRUE.

If the referenced property does not exist or is unspecified, or if its value cannot be retrieved or is of type NULL, the Time-Range evaluates to FALSE.

If the reference property is of any other type, then the evaluation is a local matter.

Note: This field can reference a Schedule object Present_Value property for the specification of time ranges.

Location-Specifier — This field is an enumeration that specifies how the Location field is evaluated:

- SPECIFIED — Location references a specific Access Point or Access Zone object and is evaluated as specified for the Location field.
- ALL — The value of the Location field is ignored and matches any access controlled point.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Location	This optional field, of type BACnetDeviceObjectReference, refers to the Access Point or Access Zone that this access rule is valid for. The Location reference shall be considered unspecified if it contains 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present. This field shall be present if Location-Specifier is SPECIFIED. If Location-Specifier is ALL and this field is present, then this reference shall be unspecified. If Location-Specifier is SPECIFIED, then the following evaluations apply: When Location refers to an Access Point object, this access controlled point is required to be the location where the credential used to request access has been authenticated. When Location refers to an Access Zone object, the access controlled point where the credential used to request access has been authenticated is required to be an entry point to this zone. If the referenced object does not exist, is unspecified, or cannot be retrieved, then the Location evaluates to FALSE.
Enable	This field, of type BOOLEAN, specifies whether this rule is enabled (TRUE) or not (FALSE).

An access rule evaluates to true when all of the following conditions are met:

(a) Time_Range field evaluates to TRUE, and
(b) the Location field matches the location of authentication, and
(c) the Enable field is TRUE.

12.34.9.2 Access Rules Authorization Check

The access rules authorization check is performed on the access rules assigned to a credential.

All the negative access rules of all the Access Rights objects referenced by the respective Access Credential object shall be evaluated before the positive access rules. If any enabled negative rule evaluates to true, then this authorization check fails and access shall be denied. In this case, the Access_Event property of the Access Point object shall be set to DENIED_POINT_NO_ACCESS_RIGHTS if the negative rule prohibits access through the access point, or to DENIED_ZONE_NO_ACCESS_RIGHTS if the negative rule prohibits access to the zone.

If no negative access rule evaluates to true, then the positive access rules of all the Access Rights objects referenced by the respective Access Credential object shall be evaluated. When the first enabled positive access rule is found that evaluates to true, then this authorization check succeeds. Access may subsequently be denied or granted based on other authorization checks.

If all positive access rules of all the Access Rights objects referenced by this credential evaluate to false, then this authorization check shall fail. In this case, if the credential has access through this access point or to the access zone at a different time, then the Access_Event property of the Access Point object shall be set to DENIED_OUT_OF_TIME_RANGE. Otherwise, the Access_Event property shall be set to DENIED_NO_ACCESS_RIGHTS.

If the respective Access Credential object contains the value ACCESS_RIGHTS in the Authorization_Exemptions property, then this authorization check is not performed and always considered successful.

12.34.9.3 Initializing New Array Elements When the Array Size is Increased

If the size of the Negative_Access_Rules array is increased without initial values being provided, then the new array elements, for which no initial value is provided, shall be initialized to contain SPECIFIED for the Time-Range-Specifier

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

field, an unspecified reference in the Time-Range field, SPECIFIED for the Location-Specifier field, an unspecified reference in the Location field, and FALSE for the Enable field.

12.34.10 Positive_Access_Rules

This property, of type BACnetARRAY of BACnetAccessRule, specifies the positive access rules. Each element of the array is evaluated as described in Clause 12.34.9.1.

To determine how access rules are evaluated, see Clause 12.34.9.2

12.34.10.1 Initializing New Array Elements When the Array Size is Increased

If the size of the Positive_Access_Rules array is increased without initial values being provided, then the new array elements, for which no initial value is provided, shall be initialized to contain SPECIFIED for the Time-Range-Specifier field, an unspecified reference in the Time-Range field, SPECIFIED for the Location-Specifier field, an unspecified reference in the Location field, and FALSE for the Enable field.

12.34.11 Accompaniment

This property, of type BACnetDeviceObjectReference, specifies that the access rights, which this object represents, may be evaluated successfully only if the original credential, which has this Access Rights object assigned, is accompanied by a second credential that meets the accompaniment criteria and is presented at the same access point. The accompanying credential must also have valid access rights to the Access Point where both credentials are presented. It is a local matter as to whether the accompanying credential is required to be presented by the access user before or after the original credential.

If the accompanying credential is not presented within the amount of time, specified by the Accompaniment_Time property of the Access Point object, then the authorization of the original credential will fail. If this time is not specified then the amount of time to wait for the accompanying credential is a local matter. When the expected accompaniment is not received, the Access_Event property of the Access Point object shall be set to DENIED_NO_ACCOMPANIMENT.

When an accompaniment is presented, whether valid or not, the Access_Event property of the Access Point object shall be set to ACCOMPANIMENT_BY.

The accompaniment criteria are specified as:

(a) If this property refers to an Access Rights object, then the accompanying credential is required to have that Access Rights object assigned.
(b) If this property refers to an Access Credential object, then this object is required to represent the accompanying credential.
(c) If this property refers to an Access User object, then this object is required to represent the access user which owns the accompanying credential.

If an invalid accompaniment is provided, then the Access_Event property of the Access Point object shall be set to DENIED_INCORRECT_ACCOMPANIMENT.

If no accompaniment requirement is specified then this reference shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.

12.34.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.34.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.34.14 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.34.15 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.34.16 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.34.17 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.34.18 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.35 Access Credential Object Type

The Access Credential object type defines a standardized object whose properties represent the externally visible characteristics of a credential that is used for authentication and authorization when requesting access.

The credential can be owned by an access user of any type. Access user ownership is represented by a reference to an Access User object.

The Access Credential object is a container of related authentication factors. Each authentication factor in the credential can be individually enabled or disabled. An Access Credential object can represent a single authentication factor, a group of authentication factors each having identical access rights, or multiple authentication factors required for multi-factor-authentications.

The access rights assigned to the credential are specified by referencing Access Rights objects. Each reference can be individually enabled or disabled.

The Credential_Status indicates the validity of this credential for authentication. The status is derived from other properties of this object or can be set from an external process.

The credential can be restricted in its use for authentication. It can be restricted based on activation and expiry dates, the number of days it can be used or the number of uses. It can be disabled if it is not used for a specified number of days. The credential can be exempted from authorization checks such as passback violation enforcement and occupancy enforcements. It can indicate whether an extended time is required to pass through a door.

A threat authority can be specified for the credential. If this value is lower than the threat level at the access controlled point, then access is denied.

The credential can be flagged to be traced. Any access controlled point recognizing this credential shall generate a corresponding TRACE access event.

When a credential is represented in multiple devices, the representing Access Credential objects may not have the same Object_Identifier in each device; however, they may be identified using the Global_Identifier property. It is a local matter as to how these objects are synchronized.

Table 12-40. Properties of the Access Credential Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Global_Identifier	Unsigned32	W
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Credential_Status	BACnetBinaryPV	R
Reason_For_Disable	BACnetLIST of BACnetAccessCredentialDisableReason	R
Authentication_Factors	BACnetARRAY[N] of BACnetCredentialAuthenticationFactor	R
Activation_Time	BACnetDateTime	R
Expiration_Time	BACnetDateTime	R
Credential_Disable	BACnetAccessCredentialDisable	R
Days_Remaining	INTEGER	O[1]
Uses_Remaining	INTEGER	O
Absentee_Limit	Unsigned16	O[1]
Belongs_To	BACnetDeviceObjectReference	O
Assigned_Access_Rights	BACnetARRAY[N] of BACnetAssignedAccessRights	R
Last_Access_Point	BACnetDeviceObjectReference	O
Last_Access_Event	BACnetAccessEvent	O
Last_Use_Time	BACnetDateTime	O
Trace_Flag	BOOLEAN	O

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Threat_Authority	BACnetAccessThreatLevel	O
Extended_Time_Enable	BOOLEAN	O
Authorization_Exemptions	BACnetLIST of BACnetAuthorizationExemption	O
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[2]
Auditable_Operations	BACnetAuditOperationFlags	O[2]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If this property is present, then the property Last_Use_Time shall also be present.
[2] This property shall be present only if the device supports audit reporting.

12.35.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.35.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.35.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCESS_CREDENTIAL.

12.35.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.35.5 Global_Identifier

This property, of type Unsigned32, is a unique identifier which is used to globally identify the credential this object represents. This value may be used to identify Access Credential objects in multiple devices which represent the same credential.

If this value is assigned, it shall be unique internetwork-wide and all Access Credential objects in all devices which represent this credential shall have this value. A value of zero indicates that no global identifier is assigned.

12.35.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM The value of this flag shall be logical FALSE (0).

FAULT Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN The value of this flag shall be logical FALSE (0).

OUT_OF_SERVICE The value of this flag shall be logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.7 Reliability

This property, of type BACnetReliability, provides an indication of whether this object is "reliable" as far as the BACnet device can determine and, if not, why.

12.35.8 Credential_Status

This property, of type BACnetBinaryPV, specifies whether the credential is active or inactive. Only the value ACTIVE enables the credential to be used for authentication. While the list in property Reason_For_Disable is nonempty, the status of the credential shall be INACTIVE, otherwise it shall be ACTIVE.

When an inactive credential is used, the authentication of this credential shall fail and access to the access point shall be denied. In this case, the Access_Event property of the Access Point object where the credential has attempted access shall be set to the value which corresponds to the reason this credential is disabled, as specified in the Reason_For_Disable property. See Clause 12.36.9.1.

12.35.9 Reason_For_Disable

This property, of type BACnetLIST of BACnetAccessCredentialDisableReason, contains a list of disable-reasons why the credential has been disabled. The credential can be disabled for multiple reasons at the same time. While the Credential_Status property has a value INACTIVE, this list shall not be empty. When an entry is removed from this list that results in the list becoming empty, the Credential_Status shall be set to ACTIVE.

The disable-reasons for which the credential can be disabled are as follows:

DISABLED	The credential is disabled for unspecified reasons.
DISABLED_NEEDS_PROVISIONING	The credential needs further provisioning, which may include vendor proprietary data.
DISABLED_UNASSIGNED	The credential is not currently assigned to any access user. This status is assigned only if the property Belongs_To is present and contains instance 4194303 in the object identifier.
DISABLED_NOT_YET_ACTIVE	The credential is not yet valid at this time. The current time is before the Activation_Time.
DISABLED_EXPIRED	The credential is no longer valid. The current time is after the Expiration_Time.
DISABLED_LOCKOUT	Too many retries in multi-factor authentications have been performed.
DISABLED_MAX_DAYS	The maximum number of days for which this credential is valid for has been exceeded.
DISABLED_MAX_USES	The maximum number of uses for which this credential is valid for has been exceeded.
DISABLED_INACTIVITY	The credential has exceeded the allowed period of inactivity.
DISABLED_MANUAL	The credential is commanded to be disabled by a human operator.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to indicate disable reasons other than those defined by this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.9.1 Conditions for setting the Access_Event property of the Access Point object

When access is requested using a credential that is inactive, access shall not be granted. In this case the Access Point object representing the access point where access was requested shall set its Access_Event property as defined in the following table:

Table 12-41. Credential Disable Reasons and Applicable Access Events

Credential Disable Reason	Applicable Access Event
DISABLED_NEEDS_PROVISIONING	DENIED_CREDENTIAL_NOT_PROVISONED
DISABLED_UNASSIGNED	DENIED_CREDENTIAL_UNASSIGNED
DISABLED_NOT_YET_ACTIVE	DENIED_CREDENTIAL_NOT_YET_ACTIVE
DISABLED_LOCKOUT	DENIED_CREDENTIAL_LOCKOUT
DISABLED_MAX_DAYS	DENIED_CREDENTIAL_MAX_DAYS
DISABLED_MAX_USES	DENIED_CREDENTIAL_MAX_USES
DISABLED_INACTIVITY	DENIED_CREDENTIAL_INACTIVITY
DISABLED_MANUAL	DENIED_CREDENTIAL_MANUAL_DISABLE
DISABLED	DENIED_CREDENTIAL_DISABLED

If Reason_For_Disable contains multiple values, it is a local matter as to which corresponding access event the Access_Event property is set to.

12.35.10 Authentication_Factors

This property, of type BACnetARRAY of BACnetCredentialAuthenticationFactor, specifies the authentication factors that belong to this credential. Each element of the array has two fields:

Disable — This field, of type BACnetAccessAuthenticationFactorDisable, specifies whether the corresponding authentication factor is disabled or not. Any value other than NONE indicates that the authentication factor is not valid for authentication.

The following authentication factor disable values are defined:

DISABLED	The physical authentication factor is disabled for unspecified reasons.
DISABLED_LOST	The physical authentication factor is reported to be lost.
DISABLED_STOLEN	The physical authentication factor is reported to be stolen.
DISABLED_DAMAGED	The physical authentication factor is reported to be damaged.
DISABLED_DESTROYED	The physical authentication factor is reported to be destroyed.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to specify disable values other than those defined by this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

Authentication-Factor — This field, of type BACnetAuthenticationFactor, specifies the authentication factor that belongs to this credential.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Any access attempt using an authentication factor which is disabled shall fail. In this case, the Access_Event property of the Access Point object where this authentication factor was used shall be set to the value corresponding to the reason why it was disabled. See the following table.

Table 12-42. Authentication Factor Disable and Applicable Access Events

Authentication Factor Disable	Applicable Access Event
DISABLED	DENIED_AUTHENTICATION_FACTOR_DISABLED
DISABLED_LOST	DENIED_AUTHENTICATION_FACTOR_LOST
DISABLED_STOLEN	DENIED_AUTHENTICATION_FACTOR_STOLEN
DISABLED_DAMAGED	DENIED_AUTHENTICATION_FACTOR_DAMAGED
DISABLED_DESTROYED	DENIED_AUTHENTICATION_FACTOR_DESTROYED

12.35.10.1 Initializing New Array Elements When the Array Size is Increased

If the size of the Authentication _Factors array is increased without initial values being provided, then the new array elements for which no initial value is provided shall be initialized to contain DISABLE for the Disable field and an authentication factor with format type UNDEFINED for the Authentication-Factor field.

12.35.11 Activation_Time

This property, of type BACnetDateTime, specifies the date and time at or after which the credential becomes active. If the current time is before the activation time, the credential shall be disabled and the value DISABLED_NOT_YET_ACTIVE shall be added to the Reason_For_Disable list. The value DISABLED_NOT_YET_ACTIVE shall be removed from the list when this condition no longer applies. If all of the octets of the BACnetDateTime value contain a value of X'FF', then the credential has an activation time of 'start of time'.

12.35.12 Expiration_Time

This property, of type BACnetDateTime, specifies the date and time after which the credential will expire. This defines the end of the validity period of the credential. If the current time is after the expiry time, the credential shall be disabled and the value DISABLED_EXPIRED shall be added to the Reason_For_Disable list. The value DISABLED_EXPIRED shall be removed from the list when this condition no longer applies. If all of the fields of the BACnetDateTime value contain a value of X'FF', then the credential has an expiry time of 'end-of-time'.

12.35.13 Credential_Disable

This property, of type BACnetAccessCredentialDisable, contains a value that disables a credential for reasons external to this object. If this property is writable, then it is the mechanism by which an operator or external process may disable the credential.

When this property is changed, any disable reason added to the Reason_For_Disable list as a result of a previous change of this property shall be removed from that list. When this property takes on any value other than NONE, the corresponding disable-reason value shall be added to the Reason_For_Disable list.

The following credential disable values are defined:

NONE	The credential has not been disabled by an operator or external process.
DISABLE	The credential has been disabled for unspecified reasons. The disable-reason value DISABLED shall be added to the Reason_For_Disable property.
DISABLE_MANUAL	The credential has been disabled by a human operator. The disable-reason value DISABLED_MANUAL shall be added to the Reason_For_Disable property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

DISABLE_LOCKOUT	The credential is disabled because it has been locked out by an external process. The disable-reason value DISABLED_LOCKOUT shall be added to the Reason_For_Disable property.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values for disabling a credential other than those defined by this standard. A disable-reason value shall be added to the Reason_For_Disable property. It is a local matter which disable reason is added. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.35.14 Days_Remaining

This property, of type INTEGER, specifies the number of remaining days for which the credential can be used. If this property has a value greater than zero, its value shall be decremented by one when the credential this object represents is granted access at an access controlled point, and the current date is more recent than the date indicated in the property Last_Use_Time. If this property becomes zero, the Access Credential shall be disabled and the value DISABLED_MAX_DAYS shall be added to the Reason_For_Disable property. The value DISABLED_MAX_DAYS shall be removed from the Reason_For_Disable property when this property is set to a value greater than zero.

If this property is present and the credential this object represents is not limited in the days it can be used, then the value of this property shall be -1 and DISABLED_MAX_USES shall never be added to the Reason_For_Disable property.

If Days_Remaining is present, then Last_Use_Time shall also be present.

12.35.15 Uses_Remaining

This property, of type INTEGER, specifies the number of remaining uses that the credential can be used for authentication. If this property has a value greater than zero and access is granted at an access controlled point, then the value of this property shall be decremented by one. If this property becomes zero, then the Access Credential shall be disabled and the value DISABLED_MAX_USES shall be added to the Reason_For_Disable property. The value DISABLED_MAX_USES shall be removed from the Reason_For_Disable property when this property is set to a value greater than zero.

If this property is present and the credential this object represents is not limited in the number of uses, then the value of this property shall be -1 and DISABLED_MAX_USES shall never be added to the Reason_For_Disable property.

12.35.16 Absentee_Limit

This property, of type Unsigned16, specifies the maximum number of consecutive days during which the credential this object represents is unused before it becomes disabled. The calculation of inactivity duration is based on the time of last use as indicated by the property Last_Use_Time. If Last_Use_Time does not have a valid time and date, then the absentee limit shall be considered to not be exceeded.

An Absentee_Limit of 0 specifies that the credential this object represents must be used every day to avoid being disabled due to inactivity.

When the absentee limit is exceeded, the Access Credential shall be disabled and the value DISABLED_INACTIVITY shall be added to the Reason_For_Disable list. The value DISABLED_INACTIVITY shall be removed from the list when this condition no longer applies.

If this property is present and has the value 65535, then the credential this object represents shall never be disabled due to inactivity.

If Absentee_Limit is present, Last_Use_Time shall also be present.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.17 Belongs_To

This property, of type BACnetDeviceObjectReference, references an Access User object that represents the owning access user (i.e. person, group, or asset). If this property is present and the credential is not assigned to an access user, this property shall contain an instance number of 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present. The determination of whether the credential is valid for authentication, based on the value of this property, is a local matter. If the credential has not been assigned to an access user and the policy of the site requires that it be assigned, then the credential shall be disabled and the value DISABLED_UNASSIGNED shall be added to the Reason_For_Disable list. The value DISABLED_UNASSIGNED shall be removed from the list when this condition no longer applies.

12.35.18 Assigned_Access_Rights

This property, of type BACnetARRAY of BACnetAssignedAccessRights, specifies the access rights assigned to this credential. The structure has two fields:

Assigned-Access-Rights	This field, of type BACnetDeviceObjectReference, refers to an Access Rights object that defines access rights assigned to this credential. Each object referenced in this field shall be an Access Rights object. Any entry which references a non-existent Access Rights object shall be ignored. If no access rights are specified, then this reference shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.
Enable	This field, of type BOOLEAN, specifies whether the access rights specified in the assigned-access-rights field is enabled (TRUE) or not (FALSE) for the credential this object represents.

12.35.18.1 Initializing New Array Elements When the Array Size is Increased

If the size of the Assigned_Access_Rights array is increased without initial values being provided, then the new array elements for which no initial value is provided shall be initialized to contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present, for the Assigned-Access-Rights field, and the value FALSE for the Enable field.

12.35.19 Last_Access_Point

This property, of type BACnetDeviceObjectReference, refers to the last Access Point object where one of the authentication factors of the credential has been used. If property level COV is in effect for this property, any update of this property shall cause a COV notification to be issued, regardless of whether the value of this property changes. If the credential this object represents has never been used, then this property shall contain 4194303 in the instance part of the object identifier and in the device instance part of the device identifier, if present.

12.35.20 Last_Access_Event

This property, of type BACnetAccessEvent, shall specify the last access event generated at an access controlled point upon use of this credential. If the credential this object represents has never been used, then this property shall have a value of NONE.

12.35.21 Last_Use_Time

This property, of type BACnetDateTime, specifies the date and time of the last use of the credential at an access controlled point, independent of whether access was granted or denied. If the credential this object represents has never been used, then this property shall have the value X'FF' for all date and time octets.

12.35.22 Trace_Flag

This property, of type BOOLEAN, specifies whether the credential is being traced. When a traced credential is used at an access point, the Access_Event property of the corresponding Access Point object shall be set to TRACE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.23 Threat_Authority

This property, of type BACnetAccessThreatLevel, specifies the maximum threat level for which this credential is valid. If this value is less than the Threat_Level property of the Access Point object where the access credential is used, access is denied. If this property is not present, the threat authority of this credential is assumed to be zero.

12.35.24 Extended_Time_Enable

This property, of type BOOLEAN, specifies which command of type BACnetDoorValue shall be used to command the access door when access is granted. If extended time is enabled (TRUE), EXTENDED_PULSE_UNLOCK is used, otherwise (FALSE) PULSE_UNLOCK is used.

12.35.25 Authorization_Exemptions

This property, of type BACnetLIST of BACnetAuthorizationExemption, specifies the authorization checks from which this credential is exempt. When a credential is exempt from an authorization check, the access attempt shall not be denied due to this authorization criterion.

The following authorization exemption values are defined:

PASSBACK	The credential is exempt from passback enforcement. If a passback exemption is enabled for this credential, then the credential shall not be denied access due to passback violations.
OCCUPANCY_CHECK	The credential is exempt from occupancy enforcement. If an occupancy exemption is enabled for this credential, then the occupancy count in the Access Zone object shall be updated as normal; however, the access credential shall not be denied access due to occupancy limit enforcement.
ACCESS_RIGHTS	The credential is exempt from standard access rights checks at the access point. If an access rights exemption is enabled for this credential, then the credential shall not be denied access due to having insufficient access rights.
LOCKOUT	The credential is exempt from lockout enforcement at an access controlled point. If a lockout exemption is enabled for this credential, then the credential shall not be denied access due to the access controlled point being locked out.
DENY	The credential is exempt from being denied access due to the Authorization_Mode property of the Access Point object having the value DENY_ALL.
VERIFICATION	The credential is exempt from requiring secondary verification at an access controlled point when the Authorization_Mode property has the value VERIFICATION_REQUIRED.
AUTHORIZATION_DELAY	The credential is exempt from an authorization delay at an access controlled point when the Authorization_Mode has the value AUTHORIZATION_DELAYED.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values for exempting the credential from specific proprietary authorization checks. For proprietary extensions of this enumeration, see Clause 23 of this standard.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.26 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED.

12.35.27 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.35.28 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.35.29 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.35.30 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.35.31 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.35.32 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.36 Credential Data Input Object Type

The Credential Data Input object type defines a standardized object whose properties represent the externally visible characteristics of a process that provides authentication factors read by a physical device. An authentication factor is a data element of a credential that is a unique digital identifier used to verify the identity of a credential. A credential may have multiple authentication factors.

Examples of physical devices that may be represented by this object type are card readers, keypads, biometric readers, etc.

A single physical credential reader which supports multiple authentication factor formats may be represented by multiple Credential Data Input objects when the authentication factor formats are not functionally equivalent or cannot be used interchangeably. An example of a device of this type is a credential reader that contains both a card and biometric reader. In this case two specific Credential Data Input objects are used; one for the card reader function and one for the biometric reader function respectively.

Alternatively, a single physical credential reader that supports multiple authentication factor formats may be represented by a single Credential Data Input object when the authentication factor formats are functionally equivalent and may be used interchangeably. An example of a device of this type is a credential reader that can read multiple Wiegand formats. It is recommended that a single Credential Data Input object that supports multiple authentication factor formats be associated with a single physical device.

Credential Data Input objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Credential Data Input objects that support intrinsic reporting shall apply the NONE event algorithm.

The Credential Data Input object type and its properties are summarized in Table 12-43 and described in detail in this clause.

Table 12-43. Properties of the Credential Data Input Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetAuthenticationFactor	R[1]
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R[1]
Out_Of_Service	BOOLEAN	R
Supported_Formats	BACnetARRAY[N] of BACnetAuthenticationFactorFormat	R
Supported_Format_Classes	BACnetARRAY[N] of Unsigned	O[2]
Update_Time	BACnetTimeStamp	R
Event_Detection_Enable	BOOLEAN	O[3,4]
Notification_Class	Unsigned	O[3,4]
Event_Enable	BACnetEventTransitionBits	O[3,4]
Event_State	BACnetEventState	O[3,4]
Acked_Transitions	BACnetEventTransitionBits	O[3,4]
Notify_Type	BACnetNotifyType	O[3,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[5]
Auditable_Operations	BACnetAuditOperationFlags	O[5]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[2] The size of this array shall be the same as the size of the Supported_Formats array.
[3] These properties are required if the object supports intrinsic reporting.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] This property shall be present only if the device supports audit reporting.

12.36.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.36.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.36.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be CREDENTIAL_DATA_INPUT.

12.36.4 Present_Value

This property, of type BACnetAuthenticationFactor, is a structure that encapsulates the authentication factor value. The structure has three fields, which are defined as follows:

Format-Type — This field, of type BACnetAuthenticationFactorType, specifies the format of the authentication factor value in the Value field. The value of this field shall be one of the format types specified in the Supported_Formats property. If there is no current authentication factor value read by this object, then this field shall take on the value UNDEFINED. In addition, if this field contains a value that is not specified in the Supported_Formats property, such as after a modification of the Supported_Formats property or after the Out_Of_Service property changes from TRUE to FALSE, then this field shall take on the value UNDEFINED. If an authentication factor is read that contains errors or that cannot be interpreted as one of the specified format types, then this field shall take on the value ERROR.

Format-Class — This field, of type Unsigned, shall contain the value specified in the Supported_Format_Classes array field that corresponds to the authentication format type in the Format-Type field. If the Supported_Format_Classes property is not present, this field shall always have a value of zero. If Format-Type has a value of UNDEFINED, then this field shall have a value of zero.

Value — This field, of type OCTET STRING, holds the authentication factor value data. The encoding of this value is specified in the Format-Type field and defined in Table P-1 of Annex P.

The Present_Value property shall be writable when Out_Of_Service is TRUE.

12.36.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.36.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Credential Data Input object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability is not NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.36.7 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet device or operator can determine.

The Reliability property shall be writable when Out_Of_Service is TRUE.

12.36.8 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value of the Credential Data Input object is prevented from being modified by some process local to the BACnet device in which the object resides.

When Out_Of_Service is TRUE:

(a) the Present_Value property and the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(b) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred in the input.

12.36.9 Supported_Formats

This property, of type BACnetARRAY of BACnetAuthenticationFactorFormat, is used to specify which authentication factor formats are supported by this object. The structure of an element of this array has three fields that are defined as follows:

Format-Type	This field, of type BACnetAuthenticationFactorType, specifies a supported authentication factor format type.
Vendor-ID	This optional field, of type Unsigned16, is required when Format-Type field has a value of CUSTOM. It shall contain the BACnet vendor identifier of the vendor which defined the custom format. This value may differ from the Vendor_Identifier property value in the Device object, which identifies the device manufacturer. If the Format-Type field does not have a value of CUSTOM and this field is present it shall have a value of zero.
Vendor-Format	This optional field of type Unsigned16 is required when Format-Type field has a value of CUSTOM. It shall contain a unique identifier that identifies a specific custom authentication factor format as defined by the BACnet vendor in the Vendor-ID field. If the Format-Type field does not have a value of CUSTOM and this field is present, then it shall have a value of zero.

12.36.9.1 Resizing Supported_Formats Array and Supported_Format_Classes Array, by Writing Any of these Properties

The size of the Supported Formats array and Supported_Format_Classes arrays shall be maintained so that both have the same size. If either of these arrays is present and writable and the number of elements of one is reduced, then each of the arrays shall be truncated to the new reduced size. If either of these arrays is present and writable and the number

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

of elements of one array is increased, then the other array shall be increased to the new expanded size and the new array elements initialized according to the requirements of each property. See Clauses 12.36.9.2 and 12.36.10.2.

12.36.9.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Supported_Formats array is increased without entry values being provided, then the new array entries shall be initialized with the Format-Type having a value of UNDEFINED. If Vendor-ID and Vendor-Format are present, they shall be initialized with a value of zero.

12.36.10 Supported_Format_Classes

This property, of type BACnetARRAY of Unsigned, specifies the values that the Format-Class field of the Present_Value may take on. The value of the ith element of this array shall be used when an authentication factor is read that is of the format defined in the i^{th} element of the Supported_Formats array.

This property is used to distinguish between multiple different supported authentication factor formats, used on a site, of which two or more use the same authentication factor format type and may have colliding value ranges. A value of zero is used as the default where no differentiation is required. Otherwise, the value is site specific and can be any non-zero value.

12.36.10.1 Resizing Supported_Formats Array and Supported_Format_Classes Array by Writing Any of these Properties

See Clause 12.36.9.1

12.36.10.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Supported_Format_Classes array is increased without entry values being provided, then the new array entries shall be initialized with a value of zero.

12.36.11 Update_Time

This property, of type BACnetTimeStamp, indicates the most recent update time when the Present_Value was updated. This property shall update its value on each update of the Present_Value. If no update has yet occurred, update times of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number update times shall have the value 0.

12.36.12 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.36.13 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.36.14 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.36.15 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.36.16 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.36.17 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.36.18 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.36.19 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.36.20 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.36.21 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.36.22 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.36.23 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.36.24 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.36.25 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.36.26 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.36.27 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.37 CharacterString Value Object Type

The CharacterString Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a CharacterString Value object to make any kind of character string data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

If a set of strings is known and fixed, then a Multi-state Value object is an alternative that may provide some benefit to automated processes consuming the numeric Present_Value.

CharacterString Value objects that support intrinsic reporting shall apply the CHANGE_OF_CHARACTERSTRING event algorithm.

For reliability-evaluation, the FAULT_CHARACTERSTRING fault algorithm can be applied.

Table 12-44. Properties of the CharacterString Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	CharacterString	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[3]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	CharacterString	O[2]
Time_Delay	Unsigned	O[3,5]
Notification_Class	Unsigned	O[3,5]
Alarm_Values	BACnetARRAY[N] of BACnetOptionalCharacterString	O[3,5]
Fault_Values	BACnetARRAY[N] of BACnetOptionalCharacterString	O[7]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[8,10,12]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[9,11]
Last_Command_Time	BACnetTimeStamp	O[9,11]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[11]
Audit_Level	BACnetAuditLevel	O[13]
Auditable_Operations	BACnetAuditOperationFlags	O[13]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[13]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.
[8] This property is required if the object supports the value source mechanism.
[9] These properties are required if the object supports the value source mechanism and is commandable.
[10] This property shall be present only if the object supports the value source mechanism.
[11] These properties shall be present only if the object supports the value source mechanism and is commandable.
[12] This property shall be writable as described in Clause 19.5.
[13] This property shall be present only if the device supports audit reporting.

12.37.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.37.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.37.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be CHARACTERSTRING_VALUE.

12.37.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.37.5 Present_Value

This property, of type CharacterString, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.37.9).

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If a fault algorithm is applied, then this property shall be the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.37.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a CharacterString Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.37.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.37.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the CharacterString Value object is reliably reporting its value.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.37.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.37.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.37.11 Relinquish_Default

This property, of type CharacterString, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.37.12 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.37.13 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.37.14 Alarm_Values

This property, of type BACnetARRAY of BACnetOptionalCharacterString, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.37.15 Fault_Values

This property, of type BACnetARRAY of BACnetOptionalCharacterString, is the value of the pFaultValues parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.37.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.37.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.37.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.37.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.37.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.37.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.37.22 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.37.23 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.37.24 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.37.25 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.37.26 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.37.27 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.37.28 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.37.29 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.37.30 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.37.31 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.37.32 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.37.33 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.37.34 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.37.35 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.37.36 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.37.37 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.37.38 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.38 DateTime Value Object Type

The DateTime Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a DateTime Value object to make any kind of datetime data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

A DateTime Value object is used to represent a single moment in time. In contrast, the DateTime Pattern Value object can be used to represent multiple recurring dates and times.

DateTime Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. DateTime Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-45. Properties of the DateTime Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BACnetDateTime	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	BACnetDateTime	O[2]
Is_UTC	BOOLEAN	O
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.38.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.38.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.38.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DATETIME_VALUE.

12.38.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.38.5 Present_Value

This property, of type BACnetDateTime, indicates the current value of the object. The value of this property shall contain either a fully specified date and time or it shall indicate a fully unspecified date and time by setting all octets to X'FF'. A fully specified date and time does not contain any octets that are equal to X'FF' or the special values for the 'month' or 'day of month' fields, the Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.38.9).

12.38.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a DateTime Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.38.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.38.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the DateTime Value object is reliably reporting its value.

12.38.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.38.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.38.11 Relinquish_Default

This property, of type BACnetDateTime, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.38.12 Is_UTC

This property, of type BOOLEAN, indicates whether the Present_Value property indicates a UTC date and time (when TRUE) or a local date and time (when FALSE). If this property is absent, the Present_Value shall be a local date and time.

12.38.13 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.38.14 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.38.15 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.38.16 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.38.17 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.38.18 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.38.19 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.38.20 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.38.21 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.38.22 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.38.23 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.38.24 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.38.25 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.38.26 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.38.27 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.38.28 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.38.29 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.38.30 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.38.31 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.38.32 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.38.33 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.39 Large Analog Value Object Type

The Large Analog Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Large Analog Value object to make any kind of double-precision data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Large Analog Value objects that support intrinsic reporting shall apply the DOUBLE_OUT_OF_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

Table 12-46. Properties of the Large Analog Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Double	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[4]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Units	BACnetEngineeringUnits	R
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Double	O[2]
COV_Increment	Double	O[3]
Time_Delay	Unsigned	O[4,6]
Notification_Class	Unsigned	O[4,6]
High_Limit	Double	O[4,6]
Low_Limit	Double	O[4,6]
Deadband	Double	O[4,6]
Limit_Enable	BACnetLimitEnable	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Min_Pres_Value	Double	O
Max_Pres_Value	Double	O
Resolution	Double	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Fault_High_Limit	Double	O[9]
Fault_Low_Limit	Double	O[9]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[10,12,14]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[11,13]
Last_Command_Time	BACnetTimeStamp	O[11,13]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[13]
Audit_Level	BACnetAuditLevel	O[15]
Auditable_Operations	BACnetAuditOperationFlags	O[15]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[15]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[10] This property is required if the object supports the value source mechanism.
[11] These properties are required if the object supports the value source mechanism and is commandable.
[12] This property shall be present only if the object supports the value source mechanism.
[13] These properties shall be present only if the object supports the value source mechanism and is commandable.
[14] This property shall be writable as described in Clause 19.5.
[15] This property shall be present only if the device supports audit reporting.

12.39.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.39.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.39.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LARGE_ANALOG_VALUE.

12.39.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.39.5 Present_Value

This property, of type Double, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.39.9).

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.39.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Large Analog Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.39.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Large Analog Value object is reliably reporting its value.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

12.39.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.39.10 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.39.11 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.39.12 Relinquish_Default

This property, of type Double, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.39.13 COV_Increment

This property, of type Double, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.39.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.39.16 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.17 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.18 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.19 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.39.21 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.39.22 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.39.23 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.39.24 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.39.25 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.39.26 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.39.27 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.39.28 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.39.29 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.39.30 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.39.31 Min_Pres_Value

This property, of type Double, indicates the lowest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.39.32 Max_Pres_Value

This property, of type Double, indicates the highest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.39.33 Resolution

This read-only property, of type Double, indicates the smallest recognizable change in Present_Value in engineering units.

12.39.34 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.39.35 Fault_High_Limit

This property, of type Double, shall specify a limit that the Present_Value must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.39.36 Fault_Low_Limit

This property, of type Double, shall specify a limit that the Present_Value must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.39.37 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.39.38 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.39.39 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.39.40 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.39.41 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.39.42 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.39.43 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.39.44 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.39.45 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.39.46 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.39.47 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.40 BitString Value Object Type

The BitString Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a BitString Value object to make any kind of bitstring data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

BitString Value objects that support intrinsic reporting shall apply the CHANGE_OF_BITSTRING event algorithm.

Table 12-47. Properties of the BitString Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BIT STRING	R[1]
Bit_Text	BACnetARRAY[N] of CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[3]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	BIT STRING	O[2]
Time_Delay	Unsigned	O[3,5]
Notification_Class	Unsigned	O[3,5]
Alarm_Values	BACnetARRAY[N] of BIT STRING	O[3,5]
Bit_Mask	BIT STRING	O[3,5]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[3,5]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[5]
Event_Algorithm_Inhibit	BOOLEAN	O[5,6]
Time_Delay_Normal	Unsigned	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O
Last_Command_Time	BACnetTimeStamp	O
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O
Audit_Level	BACnetAuditLevel	O[13]
Auditable_Operations	BACnetAuditOperationFlags	O[13]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[13]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] These properties are required if the object supports intrinsic reporting.
[4] Footnote removed.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[7] If this property is present, then the Reliability property shall be present.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[8] This property is required if the object supports the value source mechanism.
[9] These properties are required if the object supports the value source mechanism and is commandable.
[10] This property shall be present only if the object supports the value source mechanism.
[11] These properties shall be present only if the object supports the value source mechanism and is commandable.
[12] This property shall be writable as described in Clause 19.5.
[13] This property shall be present only if the device supports audit reporting.

12.40.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.40.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.40.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be BITSTRING_VALUE.

12.40.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.40.5 Present_Value

This property, of type BIT STRING, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.40.10).

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.40.6 Bit_Text

This property, of type BACnetARRAY of CharacterString, contains descriptions of all possible bits of the Present_Value. The number of descriptions matches the number of bits in the Present_Value property. The bits in Present_Value have a one-to-one correspondence with one-based indices in the array, where bit (0) corresponds to array index one.

12.40.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a BitString Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.40.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.40.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the BitString Value object is reliably reporting its value.

12.40.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.40.11 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.40.12 Relinquish_Default

This property, of type BIT STRING, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.40.13 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.40.14 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.40.15 Alarm_Values

This property, of type BACnetARRAY of BIT STRING, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.40.16 Bit_Mask

This property is the pBitMask parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.40.17 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.40.18 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.40.19 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.40.20 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.40.21 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.40.22 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.40.23 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.40.24 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.40.25 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.40.26 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.40.27 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.40.28 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.40.29 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.40.30 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.40.31 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.40.32 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.40.33 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.40.34 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.40.35 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.40.36 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.40.37 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.40.38 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.40.39 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.41 OctetString Value Object Type

The OctetString Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use an OctetString Value object to make any kind of OCTET STRING data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Table 12-48. Properties of the OctetString Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	OCTET STRING	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	OCTET STRING	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[4,6,8]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[5,7]
Last_Command_Time	BACnetTimeStamp	O[5,7]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[7]
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] This property is required if the object supports the value source mechanism.
[5] These properties are required if the object supports the value source mechanism and is commandable.
[6] This property shall be present only if the object supports the value source mechanism.
[7] These properties shall be present only if the object supports the value source mechanism and is commandable.
[8] This property shall be writable as described in Clause 19.5.
[9] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.41.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.41.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.41.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be OCTETSTRING_VALUE.

© ISO 2022 – All rights reserved

12.41.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.41.5 Present_Value

This property, of type OCTET STRING, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.41.9).

12.41.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an OctetString Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.41.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.41.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the OctetString Value object is reliably reporting its value.

12.41.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.41.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.41.11 Relinquish_Default

This property, of type OCTET STRING, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.41.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.41.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.41.14 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.41.15 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.41.16 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.41.17 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.41.18 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.41.19 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.41.20 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.41.21 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.41.22 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.41.23 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.41.24 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.42 Time Value Object Type

The Time Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Time Value object to make any kind of time data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

A Time Value object is used to represent a single moment in time. In contrast, the Time Pattern Value object can be used to represent multiple recurring times.

Time Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Time Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-49. Properties of the Time Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Time	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Time	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.42.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.42.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.42.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TIME_VALUE.

12.42.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.42.5 Present_Value

This property, of type Time, indicates the current value of the object. The value of this property shall contain either a fully specified time of day or it shall indicate a fully unspecified time by setting all octets to X'FF'. A fully specified time shall contain no octets that are equal to X'FF'. This property shall always be used to indicate a time of day, not a duration or relative time value. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.42.9).

12.42.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Time Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.42.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.42.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Time Value object is reliably reporting its value.

12.42.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.42.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.42.11 Relinquish_Default

This property, of type Time, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.42.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.42.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.42.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.42.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.42.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.42.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.42.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.42.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.42.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.42.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.42.22 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.42.23 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.42.24 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.42.25 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.42.26 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.42.27 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.42.28 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.42.29 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.42.30 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.42.31 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.42.32 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43 Integer Value Object Type

The Integer Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use an Integer Value object to make any kind of signed integer data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Integer Value objects that support intrinsic reporting shall apply the SIGNED_OUT_OF_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

Table 12-50. Properties of the Integer Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	INTEGER	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[4]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Units	BACnetEngineeringUnits	R
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	INTEGER	O[2]
COV_Increment	Unsigned	O[3]
Time_Delay	Unsigned	O[4,6]
Notification_Class	Unsigned	O[4,6]
High_Limit	INTEGER	O[4,6]
Low_Limit	INTEGER	O[4,6]
Deadband	Unsigned	O[4,6]
Limit_Enable	BACnetLimitEnable	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Min_Pres_Value	INTEGER	O
Max_Pres_Value	INTEGER	O
Resolution	INTEGER	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Fault_High_Limit	INTEGER	O[9]
Fault_Low_Limit	INTEGER	O[9]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[10,12,14]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[11,13]
Last_Command_Time	BACnetTimeStamp	O[11,13]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[13]
Audit_Level	BACnetAuditLevel	O[15]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Auditable_Operations	BACnetAuditOperationFlags	O[15]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[15]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[10] This property is required if the object supports the value source mechanism.
[11] These properties are required if the object supports the value source mechanism and is commandable.
[12] This property shall be present only if the object supports the value source mechanism.
[13] These properties shall be present only if the object supports the value source mechanism and is commandable.
[14] This property shall be writable as described in Clause 19.5.
[15] This property shall be present only if the device supports audit reporting.

12.43.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.43.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.43.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be INTEGER_VALUE.

12.43.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.43.5 Present_Value

This property, of type INTEGER, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.43.9).

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.43.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Integer Value object. Three of the flags are associated with the values of another property of this object. A more

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.43.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Integer Value object is reliably reporting its value.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

12.43.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.43.10 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43.11 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.43.12 Relinquish_Default

This property, of type INTEGER, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.43.13 COV_Increment

This property, of type Unsigned, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.43.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.43.16 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.17 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.18 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.19 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.43.21 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.43.22 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43.23 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.43.24 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.43.25 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.43.26 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.43.27 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.43.28 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.43.29 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.43.30 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43.31 Min_Pres_Value

This property, of type INTEGER, indicates the lowest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.43.32 Max_Pres_Value

This property, of type INTEGER, indicates the highest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.43.33 Resolution

This read-only property, of type Integer, indicates the smallest recognizable change in Present_Value in engineering units.

12.43.34 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.43.35 Fault_High_Limit

This property, of type INTEGER, shall specify a limit that the Present_Value must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.43.36 Fault_Low_Limit

This property, of type INTEGER, shall specify a limit that the Present_Value must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.43.37 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.43.38 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.43.39 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.43.40 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43.41 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.43.42 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.43.43 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.43.44 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.43.45 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.43.46 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.43.47 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44 Positive Integer Value Object Type

The Positive Integer Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Positive Integer Value object to make any kind of unsigned data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Positive Integer Value objects that support intrinsic reporting shall apply the UNSIGNED_OUT_OF_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

Table 12-51. Properties of the Positive Integer Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Unsigned	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[4]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Units	BACnetEngineeringUnits	R
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Unsigned	O[2]
COV_Increment	Unsigned	O[3]
Time_Delay	Unsigned	O[4,6]
Notification_Class	Unsigned	O[4,6]
High_Limit	Unsigned	O[4,6]
Low_Limit	Unsigned	O[4,6]
Deadband	Unsigned	O[4,6]
Limit_Enable	BACnetLimitEnable	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]
Min_Pres_Value	Unsigned	O
Max_Pres_Value	Unsigned	O
Resolution	Unsigned	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Fault_High_Limit	Unsigned	O[9]
Fault_Low_Limit	Unsigned	O[9]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[10,12,14]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[11,13]
Last_Command_Time	BACnetTimeStamp	O[11,13]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[13]
Audit_Level	BACnetAuditLevel	O[15]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Auditable_Operations	BACnetAuditOperationFlags	O[15]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[15]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[10] This property is required if the object supports the value source mechanism.
[11] These properties are required if the object supports the value source mechanism and is commandable.
[12] This property shall be present only if the object supports the value source mechanism.
[13] These properties shall be present only if the object supports the value source mechanism and is commandable.
[14] This property shall be writable as described in Clause 19.5.
[15] This property shall be present only if the device supports audit reporting.

12.44.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.44.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.44.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be POSITIVE_INTEGER_VALUE.

12.44.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.44.5 Present_Value

This property, of type Unsigned, indicates the current value of the object. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.44.9).

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Positive Integer Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.44.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Positive Integer Value object is reliably reporting its value.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

12.44.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.10 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.44.11 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.44.12 Relinquish_Default

This property, of type Unsigned, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.44.13 COV_Increment

This property, of type Unsigned, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.44.14 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.44.16 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.17 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.18 Deadband

This property is the pDeadband parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.19 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.44.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.44.21 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.22 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.44.23 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.44.24 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.44.25 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.44.26 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.44.27 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.44.28 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.44.29 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.30 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.44.31 Min_Pres_Value

This property, of type Unsigned, indicates the lowest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.44.32 Max_Pres_Value

This property, of type Unsigned, indicates the highest number in engineering units that can be reliably obtained or used for the Present_Value property of this object.

12.44.33 Resolution

This read-only property, of type Unsigned, indicates the smallest recognizable change in Present_Value in engineering units.

12.44.34 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.44.35 Fault_High_Limit

This property, of type Unsigned, shall specify a limit that the Present_Value must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.44.36 Fault_Low_Limit

This property, of type Unsigned, shall specify a limit that the Present_Value must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.44.37 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.44.38 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.44.39 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.40 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.44.41 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.44.42 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.44.43 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.44.44 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.44.45 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.44.46 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.44.47 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.45 Date Value Object Type

The Date Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Date Value object to make any kind of date data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

A Date Value object is used to represent a single day. In contrast, the Date Pattern Value object can be used to represent multiple recurring dates.

Date Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Date Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-52. Properties of the Date Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Date	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Date	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.45.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.45.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.45.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DATE_VALUE.

12.45.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.45.5 Present_Value

This property, of type Date, indicates the current value of the object. The value of this property shall contain either a fully specified date or it shall indicate a fully unspecified date by setting all octets to X'FF'. A fully specified date shall not contain octets that are equal to X'FF' or contain special values for the 'month' or 'day of month' fields. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.45.9).

12.45.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Date Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.45.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.45.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Date Value object is reliably reporting its value.

12.45.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.45.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.45.11 Relinquish_Default

This property, of type Date, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.45.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.45.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.45.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.45.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.45.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.45.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.45.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.45.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.45.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.45.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.45.22 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.45.23 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.45.24 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.45.25 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.45.26 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.45.27 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.45.28 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.45.29 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.45.30 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.45.31 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.45.32 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.46 DateTime Pattern Value Object Type

The DateTime Pattern Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a DateTime Pattern Value object to make any kind of datetime data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

DateTime Pattern objects can be used to represent multiple recurring dates and times based on rules defined by the pattern of individual fields of the date and time, some of which can be special values like "even months", or "don't care", which matches any value in that field. Examples of possibilities would be: "11:00 every Thursday in any June", or "every day in May 2009".

DateTime Pattern Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. DateTime Pattern Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-53. Properties of the DateTime Pattern Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BACnetDateTime	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Is_UTC	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	BACnetDateTime	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.
[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.46.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.46.2 Object_Name

This read-only property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.46.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DATETIMEPATTERN_VALUE.

12.46.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.46.5 Present_Value

This property, of type BACnetDateTime, indicates the current value of the object. The value of this property may indicate a fully specified date and time or a partially specified datetime pattern by containing one or more unspecified octets that are equal to X'FF' or the special values for the 'month' or 'day of month' fields. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.46.9).

12.46.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a DateTime Pattern Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.46.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.46.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the DateTime Pattern Value object is reliably reporting its value.

12.46.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.46.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.46.11 Relinquish_Default

This property, of type BACnetDateTime, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.46.12 Is_UTC

This property, of type BOOLEAN, indicates whether the Present_Value property indicates a UTC date and time (when TRUE) or a local date and time (when FALSE). If this property is absent, the Present_Value shall be a local date and time.

12.46.13 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.46.14 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.46.15 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.46.16 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.46.17 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.46.18 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.46.19 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.46.20 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.46.21 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.46.22 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.46.23 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.46.24 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.46.25 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.46.26 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.46.27 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.46.28 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.46.29 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.46.30 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.46.31 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.46.32 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.46.33 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.47 Time Pattern Value Object Type

The Time Pattern Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Time Pattern Value object to make any kind of time data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Time Pattern objects can be used to represent multiple recurring times based on rules defined by the pattern of individual fields of the time, some of which may be "don't care", which matches any value in that field. Examples of possibilities would be: "every minute of the 11 o'clock hour of the day", or "the thirteenth minute of any hour".

Time Pattern Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Time Pattern Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-54. Properties of the Time Pattern Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Time	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Time	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[8] This property shall be present only if the object supports the value source mechanism.
[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.47.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.47.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.47.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TIMEPATTERN_VALUE.

12.47.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.47.5 Present_Value

This property, of type Time, indicates the current value of the object. The value of this property may indicate a fully specified time or a partially specified time pattern by containing one or more "unspecified" octets that are equal to X'FF'. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.47.9).

12.47.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Time Pattern Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.47.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.47.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Time Pattern Value object is reliably reporting its value.

12.47.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.47.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.47.11 Relinquish_Default

This property, of type Time, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.47.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.47.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.47.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.47.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.47.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.47.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.47.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.47.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.47.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.47.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.47.22 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.47.23 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.47.24 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.47.25 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.47.26 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.47.27 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.47.28 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.47.29 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.47.30 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.47.31 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.47.32 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.48 Date Pattern Value Object Type

The Date Pattern Value object type defines a standardized object whose properties represent the externally visible characteristics of a named data value in a BACnet device. A BACnet device can use a Date Pattern Value object to make any kind of date data value accessible to other BACnet devices. The mechanisms by which the value is derived are not visible to the BACnet client.

Date Pattern objects can be used to represent multiple recurring dates based on rules defined by the pattern of individual fields of the date, some of which can be special values like "even months", or "don't care", which matches any value in that field. Examples of possibilities would be: "every Thursday in May of any year", or "every day in May 2009".

Date Pattern Value objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Date Pattern Value objects that support intrinsic reporting shall apply the NONE event algorithm.

Table 12-55. Properties of the Date Pattern Value Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Date	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	O
Priority_Array	BACnetPriorityArray	O[2]
Relinquish_Default	Date	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[3]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Event_Detection_Enable	BOOLEAN	O[4,5]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Current_Command_Priority	BACnetOptionalUnsigned	O[2]
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if, and shall be present only if, Present_Value is commandable.
[3] If this property is present, then the Reliability property shall be present.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

12.48.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.48.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.48.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be DATEPATTERN_VALUE.

12.48.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.48.5 Present_Value

This property, of type Date, indicates the current value of the object. The value of this property may indicate a fully specified date or a partially specified date pattern by containing one or more "unspecified" octets that are equal to X'FF' or the special values for the 'month' or 'day of month' fields. The Present_Value property shall be writable when Out_Of_Service is TRUE (see Clause 12.48.9).

12.48.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Date Value object. Three of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property is present and has a value of TRUE, otherwise logical FALSE (0).

12.48.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.48.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Date Value object is reliably reporting its value.

12.48.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) the Present_Value of the object is prevented from being changed by software local to the BACnet device in which the object resides;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value, or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;
(d) if the Priority_Array and Relinquish_Default properties are present, the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

Restrictions on changing the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

12.48.10 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.48.11 Relinquish_Default

This property, of type Date, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.48.12 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.48.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.48.14 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.48.15 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.48.16 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.48.17 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.48.18 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.48.19 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.48.20 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.48.21 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.48.22 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.48.23 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.48.24 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.48.25 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.48.26 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.48.27 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.48.28 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.48.29 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.48.30 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.48.31 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.48.32 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.49 Deleted Clause

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.50 Global Group Object Type

The Global Group object type defines a standardized object whose properties represent a collection of other objects and one or more of their properties. A Global Group object is used to simplify the exchange of information between BACnet devices by providing a shorthand way to specify all members of the group at once.

A Global Group object differs from a Group object in that its members can be from anywhere in the BACnet internetwork, it supports intrinsic event reporting, and it exposes a method for sending periodic COV notifications. The Global Group object is able to monitor all referenced Status_Flags properties to detect changes to non-normal states and can initiate an event notification message conveying the values of all of the members of the group. This provides a mechanism to define a large set of property values that are made available when an event occurs.

Global Group objects that support intrinsic reporting shall apply the CHANGE_OF_STATUS_FLAGS event algorithm.

For reliability-evaluation, the FAULT_STATUS_FLAGS fault algorithm shall be applied.

Table 12-57. Properties of the Global Group Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Group_Members	BACnetARRAY[N] of BACnetDeviceObjectPropertyReference	R
Group_Member_Names	BACnetARRAY[N] of CharacterString	O
Present_Value	BACnetARRAY[N] of BACnetPropertyAccessResult	R
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Member_Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O
Out_of_Service	BOOLEAN	R
Update_Interval	Unsigned	O
Requested_Update_Interval	Unsigned	O
COV_Resubscription_Interval	Unsigned	O
Client_COV_Increment	BACnetClientCOV	O
Time_Delay	Unsigned	O[1,4]
Notification_Class	Unsigned	O[1,4]
Event_Enable	BACnetEventTransitionBits	O[1,4]
Acked_Transitions	BACnetEventTransitionBits	O[1,4]
Notify_Type	BACnetNotifyType	O[1,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Event_Detection_Enable	BOOLEAN	O[1,4]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[4]
Event_Algorithm_Inhibit	BOOLEAN	O[4,5]
Time_Delay_Normal	Unsigned	O[4]
COVU_Period	Unsigned	O[2]
COVU_Recipients	BACnetLIST of BACnetRecipient	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[6]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[7]
Auditable_Operations	BACnetAuditOperationFlags	O[7]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

[2] These properties are required if the object sends periodic unsubscribed COV notifications for Present_Value. These properties are required to be writable if present.
[3] Footnote removed.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[6] If this property is present, then the Reliability property shall be present.
[7] This property shall be present only if the device supports audit reporting.

12.50.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.50.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.50.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be GLOBAL_GROUP.

12.50.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.50.5 Group_Members

This property, of type BACnetARRAY of BACnetDeviceObjectPropertyReference, defines the members of the group. If the optional device identifier is not present for a particular group member, then that object shall reside in the same device that maintains the Global Group object. If the Group_Members property is writable using WriteProperty services, then the object shall support group members that are outside the device that maintains the Global Group object.

Nesting of group objects is not permitted; that is, Group_Members shall not refer to the Present_Value property of a Group object or a Global Group object.

12.50.5.1 Resizing Group_Members and Group_Member_Names by Writing Either Property

The size of the Group_Members and Group_Member_Names properties shall be maintained so that both have the same size. If either of these arrays is writable and the size of one array is reduced, the size of the other array and the Present_Value array shall also be truncated to the new reduced size. If the size of either array is increased, the other array and the Present_Value array shall all be increased to the new expanded size and the new array elements initialized according to the requirements of each property. See Clauses 12.50.5.2, 12.50.6.2, and 12.50.7.1.

12.50.5.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Group_Members array is increased by writing to the size of either the Group_Members or Group_Member_Names property, the new array entries shall be initialized by setting the object or device instance numbers of the BACnetDeviceObjectPropertyReference equal to 4194303, indicating that the value is not initialized. The initial value of the other parameters is a local matter except that they shall be of the correct datatype.

12.50.6 Group_Member_Names

This property, of type BACnetARRAY of CharacterString, represents a descriptive name for the members of the Global Group. The number of names matches the number of members defined in Group_Members. The array index of the name shall match the array index of the corresponding group member.

12.50.6.1 Resizing Group_Members and Group_Member_Names by Writing Either Property

See Clause 12.50.5.1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.50.6.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Group_Member_Names array is increased by writing to the size of either the Group_Members or Group_Member_Names property, the new array entries shall be initialized with empty strings.

12.50.7 Present_Value

This read-only property, of type BACnetARRAY of BACnetPropertyAccessResult, contains the values of all the properties specified in the Group_Members property. The array index of the Present_Value shall match the corresponding array index in Group_Members. This is a "read-only" property; it cannot be used to write a set of values to the members of the group.

The Present_Value data shall be stored locally. The Present_Value may be updated based on COV notifications, polling, or a combination of the two. The method of acquisition used for any particular member is a local matter. If the Present_Value, or a portion of the Present_Value, is acquired periodically and the Requested_Update_Interval property is present, then an attempt shall be made to update the Present_Value within this time interval. If the Present_Value, or a portion of the Present_Value, is acquired periodically and the Requested_Update_Interval is not present, then the update interval is a local matter. When updating the Present_Value, if a group member's property value cannot be acquired, a property access error shall be stored in the access result for that member of the group. If a property access error was returned when attempting to update the group member's property value, then that access error shall be the one stored in the access result. Otherwise, the choice of property access error to store shall be a local matter.

The Present_Value array shall be maintained at the same size as the Group_Members array. If the Group_Members property is writable and the size of the array is reduced, the Present_Value array shall be truncated to match. If the Group_Members property is writable and the size of the array is increased, the Present_Value array shall be increased in size to match with the value of the new array elements being determined through the same mechanism that is used to update the values. If a specific element in the Group_Members property changes, then the corresponding element in the Present_Value array shall be updated through the same mechanism that is used to update the values. Note that the size of the Group_Members property can also be affected by changing the size of the Group_Member_Names property.

The value of the Present_Value property shall continue to be updated regardless of the value of the Reliability property.

12.50.7.1 Initializing New Array Elements When the Array Size is Increased

If the size of the Present_Value array is increased by writing to the size of either the Group_Members or Group_Member_Names property, the new array entries shall be initialized with the Access Result parameter having a value of type PropertyAccessError, with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_NOT_INITIALIZED. The other parameters shall have values consistent with the corresponding entry in the Group_Members array.

12.50.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Global Group object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the global group has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value is no longer tracking the group members' values, the Event_State property is no longer tracking changes to the Event_State of group member objects, and the Reliability property is no longer a reflection of the result of any internal algorithm for determining the reliability of the Global Group object. Otherwise, the value is logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OUT_OF_SERVICE Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.50.9 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.50.10 Member_Status_Flags

The Member_Status_Flags property is a logical combination of all the Status_Flags properties contained in the Present_Value. The logical combination means that each of the flags in this property (IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE) is TRUE if and only if the corresponding flag is set in any of the Status_Flags property values in the Present_Value property. This property shall be updated whenever new Status_Flags property values are updated in the Present_Value.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm and the pSelectedFlags parameter shall have the IN_ALARM and FAULT bits set and the others cleared. See Clause 13.3 for event algorithm parameter descriptions.

This property is the pMonitoredValue fault algorithm parameter. See Clause 13.4 for fault algorithm parameter descriptions.

12.50.11 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet device or operator can determine. If one or more group member values cannot be updated because of a communication failure, the value of this property shall be COMMUNICATION_FAILURE.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.50.12 Out_Of_Service

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) the object is out of service.

When Out_Of_Service is TRUE:

(a) the Present_Value property is decoupled and is not updated to track the values of the group members;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the state of the object ;
(c) the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Reliability property shall respond to changes made to this property as if those changes had occurred by normal operation.

12.50.13 Update_Interval

This property, of type Unsigned, provides an indication of the actual period of time between updates to Present_Value, measured in hundredths of a second. The method used to calculate Update_Interval is a local matter.

12.50.14 Requested_Update_Interval

This property, of type Unsigned, indicates the requested period of time between updates to Present_Value, measured in hundredths of a second when the object is not out-of-service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.50.15 COV_Resubscription_Interval

If the Global Group is acquiring data from a remote device by COV subscription, this property, of type Unsigned, specifies the number of seconds between COV resubscriptions, provided that COV subscription is in effect. SubscribeCOV requests shall specify twice this lifetime for the subscription and shall specify the issuance of confirmed notifications. If COV subscriptions are in effect, the first COV subscription is issued when the Global Group object begins operation. If present, the value of this property shall be non-zero. If this property is not present, then COV subscription shall not be attempted.

12.50.16 Client_COV_Increment

If the Global Group is acquiring COV data, this property, of type BACnetClientCOV, specifies the increment to be used in determining that a change of value has occurred. If all the referenced objects and properties support COV reporting according to Clause 13.1, this property may have the value NULL; in this case change of value is determined by the criteria of Clause 13.1.

12.50.17 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.50.18 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.50.19 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.50.20 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.50.21 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.50.22 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.50.23 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.50.24 COVU_Period

The optional COVU_Period property, of type Unsigned, shall indicate the amount of time in seconds between the periodic unsubscribed COV notifications performed by this object. These COV notifications convey the value of the Present_Value and Member_Status_Flags properties. If the value of COVU_Period is zero, then periodic unsubscribed COV notification messages shall not be transmitted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.50.25 COVU_Recipients

This property, of type BACnetLIST of BACnetRecipient, is used to control the restrictions on which devices, if any, are to receive periodic unsubscribed COV notifications for the Present_Value and Member_Status_Flags properties. This property is required if the object sends such notifications. The value of this property shall be a list of zero or more BACnetRecipients. If the list is of length zero, a device is prohibited from sending such notifications. If the list is of length one or more, the device shall send the notifications, but only to the devices or addresses listed.

12.50.26 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.50.27 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.50.28 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.50.29 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.50.30 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.50.31 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.50.32 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.50.33 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.50.34 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.50.35 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.50.36 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.50.37 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.51 Notification Forwarder Object Type

The Notification Forwarder object type defines a standardized object whose properties represent the externally visible characteristics required for the re-distribution of event notifications to zero or more destinations. It differs from a Notification Class in that the Notification Forwarder object is not used for originating event notifications, but rather is used to forward event notifications to a different and potentially larger number of recipients.

The Notification Forwarder allows devices that can distribute notifications to a small number of destinations to have their notifications received by many destinations. It also allows for a reduction in the number of objects that need to be modified in order to change the set of event destinations for a large number of devices. Notification Forwarder objects can also be restricted to forward only locally generated notifications as indicated by the Local_Forwarding_Only property. In doing so, the Notification Forwarder object allows for the use of recipient subscriptions and the centralization of recipients for multiple Notification Classes.

A Notification Forwarder object's Process_Identifier_Filter value is used in the selection of event notification service requests that are to be forwarded by the object. If multiple Notification Forwarder objects exist within the device, any received event notification shall be passed to each Notification Forwarder object internally by the device if the Process_Identifier_Filter and other restrictions allow acceptance by the object. Event notifications generated by local event-initiating objects are only passed to local Notification Forwarder objects if the Notification Class object has the local device as a recipient.

The following restrictions are intended to reduce the likelihood of an accidental endless cycle of event forwarding for the same notification or of accidental duplicated notifications to the same device.

(a) Any event notification received on a particular port of the device containing Notification Forwarder objects shall not be forwarded as a local broadcast to the BACnet network directly attached to that port.
(b) Any event notification received as a global broadcast shall be ignored by Notification Forwarder objects in receiving devices.
(c) The Notification Forwarder object shall not send any forwarded notification using a global broadcast.
(d) Any event notification received as a broadcast to a particular BACnet network shall not be forwarded to any device resident on that same network.
(e) Any event notification received on a particular port of the device containing Notification Forwarder objects shall be ignored by any Notification Forwarder object within the device that does not have the receiving port enabled within its Port_Filter property. In order to stop multiple notifications from being forwarded to the notification-clients, there should be at most one Notification Forwarder object on a network that will forward broadcast notifications. The Port_Filter allows a site to be configured this way when the Notification Forwarder objects are located in BACnet routers.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

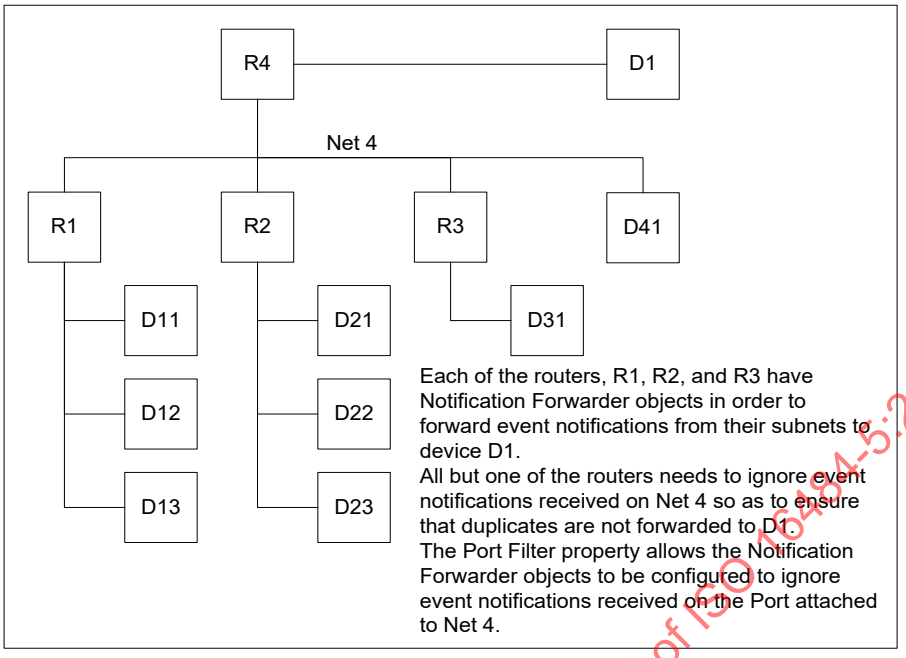


Figure 12-6. General Use Case for the Port_Filter property.

Notification Forwarder objects can be logically chained together one after another in the same or different devices.

An event notification is sent through a Notification Forwarder object in the same device by specifying the local Device object in the Notification Class object's Recipient_List along with the Process_Identifier_Filter matching the Notification Forwarding object.

Like Notification Class objects, the Notification Forwarder object allows for date, time and transition filters for destinations in the Recipient_List property. The filtering works in the same manner as the Notification Class object. In order to allow central configuration of the date, time and transition filters, Recipient_List entries that direct event notifications to Notification Forwarder objects are expected to have the filtering parameters set to send all transitions, on all days, at all times so that filtering is performed only by the Notification Forwarder object.

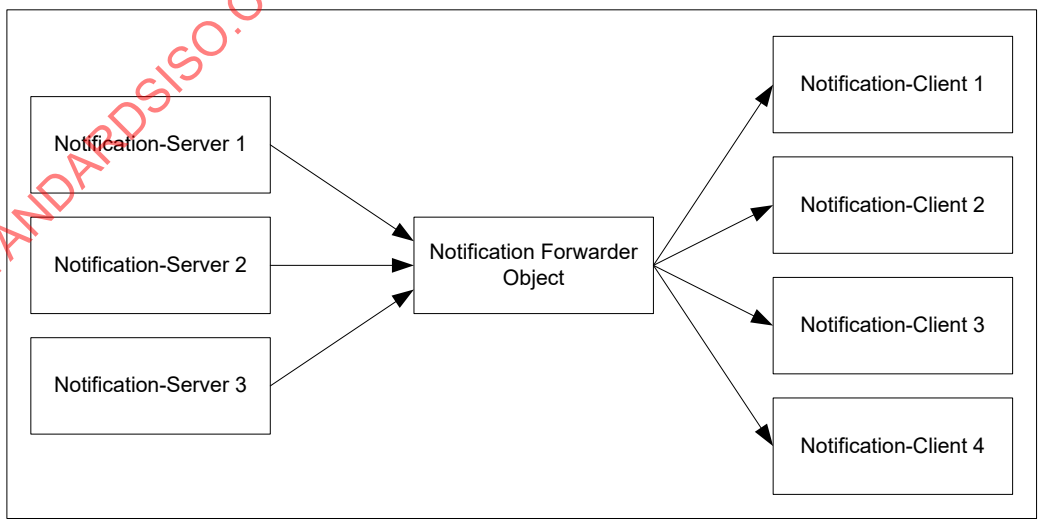


Figure 12-7. Example of Multiple Notification-Servers Routing to Multiple Notification-Clients.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

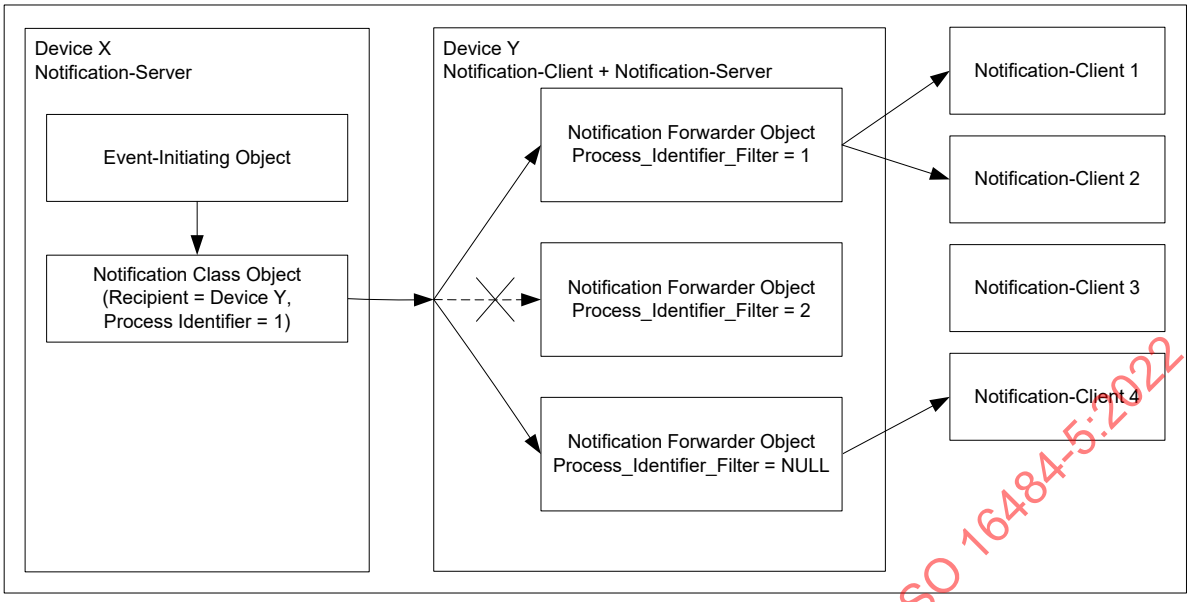


Figure 12-8. Example of Forwarding a Single-Event Notification.

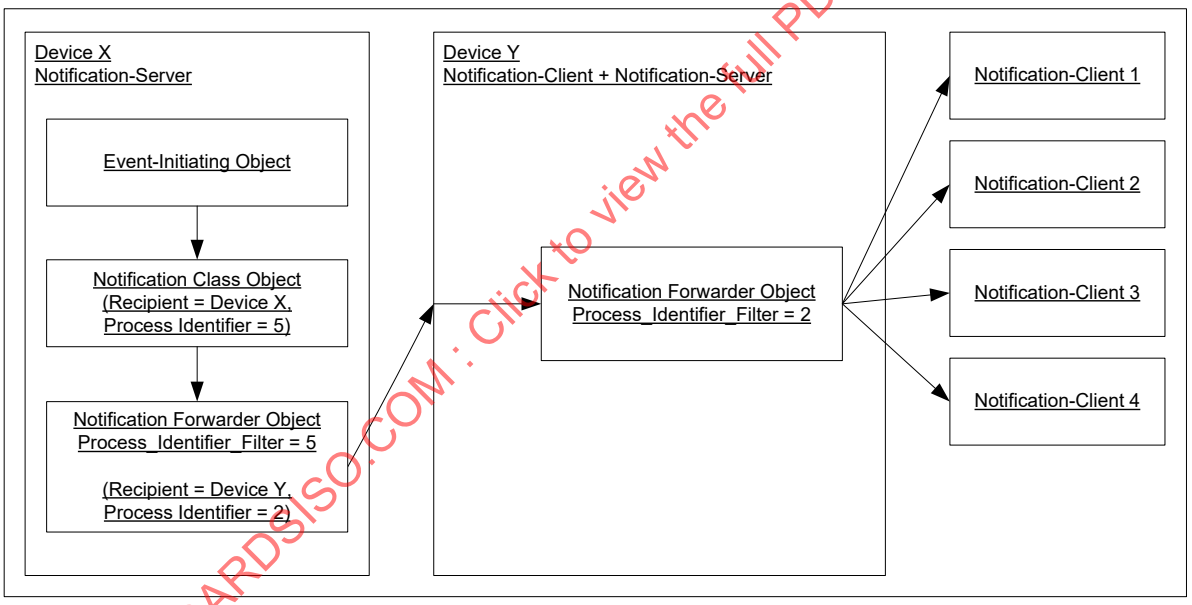


Figure 12-9. Example of Local Forwarding and Multiple-Step Forwarding.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Notification Forwarders that are forwarding for other devices shall be capable of forwarding both ConfirmedEventNotification and UnconfirmedEventNotification services, and shall be capable of forwarding them using either service regardless of which is received.

Notification Forwarder objects shall forward event notifications regardless of the character set of any text in the event notification.

To forward an event notification to a BACnetDestination, the source notification shall have the Process Identifier parameter changed to match that of the BACnetDestination, and the notification shall be sent confirmed or unconfirmed as specified by the BACnetDestination. The notification shall then be sent to the recipient indicated by the BACnetDestination.

© ISO 2022 – All rights reserved

When acknowledging an event notification that has been forwarded by a Notification Forwarder, the acknowledging device shall send the AcknowledgeAlarm service request directly to the device indicated by the Initiating Device Identifier parameter of the event notification.

Table 12-58. Properties of the Notification Forwarder Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Recipient_List	BACnetLIST of BACnetDestination	R
Subscribed_Recipients	BACnetLIST of BACnetEventNotificationSubscription	W
Process_Identifier_Filter	BACnetProcessIdSelection	R
Port_Filter	BACnetARRAY[N] of BACnetPortPermission	O[1]
Local_Forwarding_Only	BOOLEAN	R
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[2]
Auditable_Operations	BACnetAuditOperationFlags	O[2]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required if the device includes BACnet router functionality.
[2] This property shall be present only if the device supports audit reporting.

12.51.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.51.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.51.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be NOTIFICATION_FORWARDER.

12.51.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.51.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Notification Forwarder object. The OUT_OF_SERVICE flag is associated with the value of another property of this object. The relationship between individual flags is not defined by the protocol. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

where:

IN_ALARM	The value of this flag shall be Logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be Logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.51.6 Reliability

This property, of type BACnetReliability, provides an indication of whether the object is configured and operating. The value of Reliability shall not indicate that a subset of the possible recipients (Recipient_List, and Subscribed_Recipients) are not reachable, but may indicate that all possible recipients are unreachable or are undefined.

12.51.7 Out_Of_Service

This property, of type BOOLEAN is an indication whether (TRUE) or not (FALSE) the object has been prevented from forwarding event notifications.

When Out_Of_Service is TRUE:

(a) the Notification Forwarder object object is disabled and does not forward event notifications;
(b) the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Reliability property shall respond to changes made to this property as if those changes had occurred by normal operation.

12.51.8 Recipient_List

This property, of type BACnetLIST of BACnetDestination, shall convey a list of zero or more recipient destinations to which notifications shall be sent when events are processed by the Notification Forwarder object. These recipient destinations are intended to be relatively permanent, do not expire and shall be maintained through a power failure or device "restart". If not writable, the Recipient_List shall be configurable by some other means. The destinations themselves define a structure of parameters that is summarized in Table 12-25.

12.51.9 Subscribed_Recipients

This property, of type BACnetLIST of BACnetEventNotificationSubscription, conveys a list of recipient destinations to which event notifications are sent when events are forwarded by the Notification Forwarder object. These recipient destinations are intended to be temporary, and will expire if not renewed.

To add, remove, renew or modify a subscription, the AddListElement or RemoveListElement services are used. When comparing entries to those provided by a list service, an entry in this property shall be considered a match when the Recipient fields are equal and Process Identifier fields are equal. Each entry in the list is a structure of parameters that is described in Table 12-59.

Table 12-59. Components of a BACnetEventNotificationSubscription

Parameter	Type	Description
Recipient	BACnetRecipient	The destination device(s) to receive notifications.
Process Identifier	Unsigned32	The handle of a process within the recipient device that is to receive the event notification.
Issue Confirmed Notifications	Boolean	(TRUE) if confirmed notifications are to be sent and (FALSE) if unconfirmed notifications are to be sent.
Time Remaining	Unsigned	Actual time the entry will remain in the Subscribed_Recipients in minutes.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The Time Remaining field of a BACnetEventNotificationSubscription entry, of type Unsigned, indicates the remaining time of the subscription in minutes. An entry shall be removed from the list when the remaining time reaches zero, and therefore no entries in the property shall have a Time Remaining value of zero. Notification Forwarder objects shall accept subscriptions with Time Remaining values in the range of 1 through 1440 (24 hours). It is a local matter whether or not a Notification Forwarder accepts larger Time Remaining values.

When renewing an existing subscription, the values of all fields provided in the AddListElement service shall replace the values of all fields of the existing subscription.

The Subscribed_Recipients shall be maintained through a power failure or device "restart." After the restart, the Time Remaining may be any value between the value before the restart and the value provided in the entry's last subscription operation.

12.51.10 Process_Identifier_Filter

This property, of type BACnetProcessIdSelection, is used in the selection of event notification service requests that are to be forwarded by the object. When the Process Identifier parameter of a received event notification is the same as the value of the Process_Identifier_Filter property, or if the Process_Identifier_Filter property contains a NULL, then the notification will be accepted for forwarding by the Notification Forwarder object subject to the port, network and broadcast restrictions.

12.51.11 Port_Filter

This property, of type BACnetARRAY of BACnetPortPermission, enables or disables the forwarding of event notifications received on a particular network port. When an event notification is received on a port that is marked as disabled by this property, the Notification Forwarder object shall ignore that event notification.

If present, this property shall be writable. If not present, then the device is not a router and its only configured port shall be enabled for event notification forwarding.

Neither the size of the array nor the Port_ID portion of the BACnetPortPermission entries shall be modifiable via writes to this property.

The number of entries in the array shall match the number of BACnet ports currently defined in the device.

The BACnetPortPermission entries themselves define a structure of parameters that is summarized in Table 12-60.

Table 12-60. Components of a BACnetPortPermission

Parameter	Type	Description
Port_ID	Unsigned8	The Port_ID parameter shall correspond to the Port ID of the associated network as described in Clause 6. For non-routing nodes, this value shall be 0.
Enabled	Boolean	Indicates whether forwarding is enabled (TRUE) or not (FALSE) for event notifications received through the corresponding port.

12.51.12 Local_Forwarding_Only

This property, of type BOOLEAN is an indication whether (TRUE) or not (FALSE) the object is limited to forwarding notifications initiated from within the same device. If Local_Forwarding_Only has a value of FALSE, then the Notification Forwarder is capable of forwarding notifications for other devices.

12.51.13 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.51.14 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.51.15 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.51.16 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.51.17 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.51.18 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.51.19 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.52 Alert Enrollment Object Type

The Alert Enrollment object type defines a standardized object that represents and contains the information required for managing information alerts from a BACnet device. "Information alerts" are interesting notifications that are not related to algorithmic or intrinsic reporting of an object. The Alert Enrollment object allows these alerts to be generated without impacting the Event_State of the object to which the alerts are related.

Alerts are always distributed using ConfirmedEventNotification or UnconfirmedEventNotification services with 'To State' and 'From State' set to NORMAL and an 'Event Type' of EXTENDED. The values used in the 'Vendor Id' and 'Extended Event Type' parameters allow for classification of alerts. The choice of values used in the 'Vendor Id' and 'Extended Event Type' parameters is a local matter. The definition of the parameters used with any particular pair of 'Vendor Id' and 'Extended Event Type' is controlled by the registered owner of the 'Vendor Id' value. The extended notification parameters allow for a vendor-specified set of values to be provided with the notification. For the notification of alerts, the first extended notification parameter shall be a BACnetObjectIdentifier that identifies the source of the alert (not the Alert Enrollment object, but the object that provided the alert to the Alert Enrollment object). If an alert is not logically associated with a specific object, the local Device object shall be referenced as the source of the alert.

When there are multiple alert enrollment objects in a device, the method used to associate an Alert Enrollment object to any particular alert generated by an object is a local matter.

The Alert Enrollment object and its properties are summarized in Table 12-61 and described in detail in this clause.

Table 12-61. Properties of the Alert Enrollment Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BACnetObjectIdentifier	R
Event_State	BACnetEventState	R
Event_Detection_Enable	BOOLEAN	R
Notification_Class	Unsigned	R
Event_Enable	BACnetEventTransitionBits	R
Acked_Transitions	BACnetEventTransitionBits	R
Notify_Type	BACnetNotifyType	R
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	R
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O
Event_Algorithm_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[1]
Auditable_Operations	BACnetAuditOperationFlags	O[1]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property shall be present only if the device supports audit reporting.

12.52.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.52.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.52.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ALERT_ENROLLMENT.

12.52.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.52.5 Present_Value

This read-only property, of type BACnetObjectIdentifier, indicates the object that last provided an alert to this object for notification.

12.52.6 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.52.7 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.52.8 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

The TO_NORMAL transition in the Ack_Required property of the referenced Notification Class object is ignored and the value FALSE is conveyed in the 'AckRequired' parameter of the ConfirmedEventNotification or UnconfirmedEventNotification message.

12.52.9 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.52.10 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.52.11 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.52.12 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.52.13 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.52.14 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.52.15 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.52.16 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.52.17 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.52.18 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.52.19 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.52.20 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.52.21 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.52.22 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.53 Channel Object Type

The Channel object type defines a standardized object used to forward a single received value to a collection of object properties. The collection of object properties may include any combination of object types, as well as properties of different data types. The coercion of the datatype from the value written to the Channel object Present_Value to the datatypes required by the object properties is controlled by coercion rules defined in Clause 12.53.5.1.

Each Channel object is associated with a single logical "channel" in the range 0..65535. Multiple Channel object instances may be associated with a given channel number.

Each Channel object may be a member of zero or more "control groups" to facilitate writing to Channel objects with the WriteGroup service.

The Channel object is intended for value distribution and does not maintain a state. Therefore, it does not act on its own and does not contain a priority array. When the Present_Value property of this object is written by the WriteProperty, WritePropertyMultiple, or WriteGroup services, and a 'Priority' is provided in the write, this object shall use this same priority to command the referenced properties. Figure 12-10 illustrates the behavior of the Channel object.

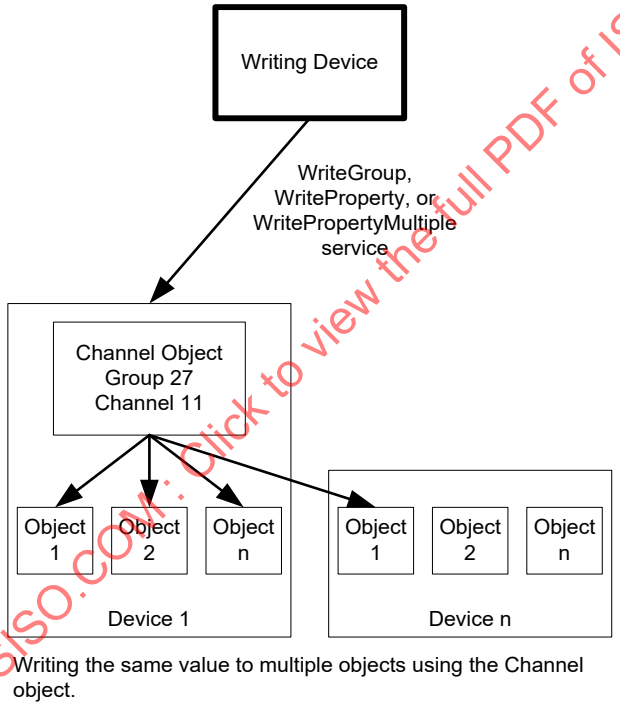


Figure 12-10. Channel Object Behavior

When the WriteGroup service is used, potentially many devices may be affected because WriteGroup is usually broadcast. As a result, WriteGroup includes a group number parameter that restricts the effect to only those receiving devices that are members of that group. The WriteGroup further restricts the targets for writing to those Channel objects within those devices that are associated with the specified channel number(s).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Devices that contain Channel objects shall also support the WriteGroup service.

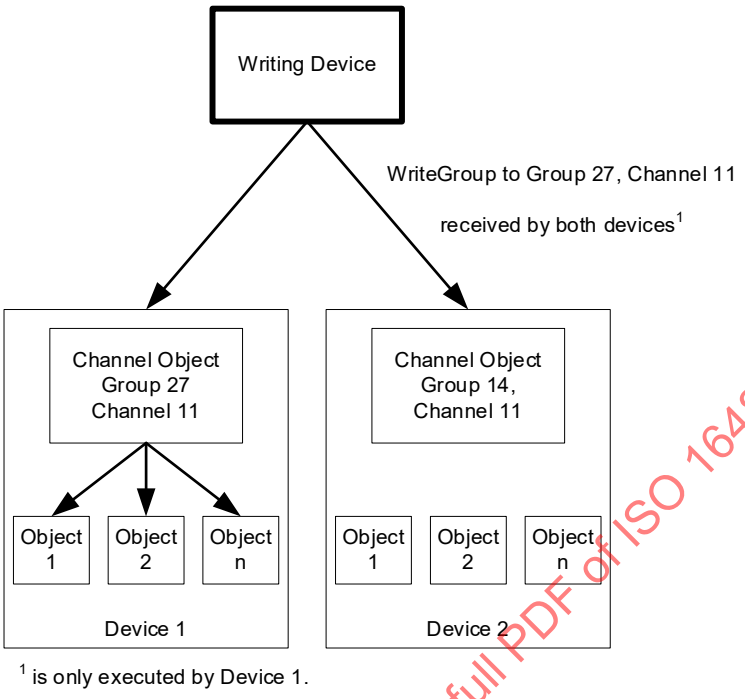


Figure 12-11. Control Groups Limit WriteGroup Effect to Specific Channel Objects Across Many Devices

Channel objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Channel objects that support intrinsic reporting shall apply the NONE event algorithm.

The object and its properties are summarized in Table 12-62 and described in detail in this clause.

Table 12-62. Properties of the Channel Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	BACnetChannelValue	W
Last_Priority	Unsigned	R
Write_Status	BACnetWriteStatus	R
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
List_Of_Object_Property_References	BACnetARRAY[N] of BACnetDeviceObjectPropertyReference	W[1]
Execution_Delay	BACnetARRAY[N] of Unsigned	O[1]
Allow_Group_Delay_Inhibit	BOOLEAN	O
Channel_Number	Unsigned16	W
Control_Groups	BACnetARRAY[N] of Unsigned32	W
Event_Detection_Enable	BOOLEAN	O[2,3]
Notification_Class	Unsigned	O[2,3]
Event_Enable	BACnetEventTransitionBits	O[2,3]
Event_State	BACnetEventState	O[2,3]
Acked_Transitions	BACnetEventTransitionBits	O[2,3]
Notify_Type	BACnetNotifyType	O[2,3]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Property Identifier	Property Datatype	Conformance Code
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,3]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[3]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[3]
Reliability_Evaluation_Inhibit	BOOLEAN	O[4]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[5,6,7]
Audit_Level	BACnetAuditLevel	O[8]
Auditable_Operations	BACnetAuditOperationFlags	O[8]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[8]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These array properties shall be the same size.
[2] These properties are required if the object supports intrinsic reporting.
[3] These properties shall be present only if the object supports intrinsic reporting.
[4] If this property is present, then the Reliability property shall be present.
[5] This property is required if the object supports the value source mechanism.
[6] This property shall be present only if the object supports the value source mechanism.
[7] This property shall be writable as described in Clause 19.5.
[8] This property shall be present only if the device supports audit reporting.

12.53.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.53.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.53.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be CHANNEL.

12.53.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.53.5 Present_Value (Commandable)

This property, of type BACnetChannelValue, shall indicate the value most recently written to the Present_Value.

When Present_Value is written, the Channel object shall propagate that value to each of the members in the List_Of_Object_Property_References except those members containing an empty reference. During the writing of values to members, Write_Status shall be IN_PROGRESS. At the end of writing all values, Write_Status shall change to SUCCESSFUL or FAILED based on the results of these writes. If Write_Status is SUCCESSFUL, then the Reliability property shall be reevaluated as described in Clause 12.53.10

When Present_Value is written with a 'Priority' parameter, the resulting writes to the members of the List_Of_Object_Property_References shall also use that 'Priority' parameter. See Clause 19.2.1.6. If the Channel object supports device-object-property references, then it may elect to use individual WriteProperty or WritePropertyMultiple, or a combination of both, to achieve the writing, as a local matter.

The initial value of the Present_Value property shall be NULL. This initial value shall not be automatically written to the properties listed in List_Of_Object_Property_References.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Attempts to write to Present_Value using WriteProperty service when Write_Status is IN_PROGRESS shall cause a Result(-) to be returned with an 'Error Class' of OBJECT and an 'Error Code' of BUSY.

```
Example                 List_Of_Object_Property_References
[1]=(101,AV27;Present_Value)
                        List_Of_Object_Property_References
[2]=(102,AO14;Present_Value)
                        List_Of_Object_Property_References
[3]=(103,AO5;Present_Value)
                        List_Of_Object_Property_References
[4]=(104,AV123;Present_Value)
                        Execution_Delay[1]=0
                        Execution_Delay[2]=100
                        Execution_Delay[3]=0
                        Execution_Delay[4]=200

t1. Present_Value written with value X
t2. If write was WriteProperty or WritePropertyMultiple, then Channel object returns
    Result(+) or Result(-)
t3. Write_Status = IN_PROGRESS
IF write was WriteGroup AND
   WriteGroup has 'Inhibit Delay'=TRUE AND
   Allow_Group_Delay_Inhibit=TRUE,    THEN
{
      t4. WriteProperty(101,AV27,Present_Value,X)
      t5. WriteProperty(102,AO14,Present_Value,X)
      t6. WriteProperty(103,AO5,Present_Value,X)
      t7. WriteProperty(104,AV123,Present_Value,X)
      t8. Write_Status = SUCCESSFUL
      t9. Reliability = NO_FAULT_DETECTED
}
ELSE
{
      t4. WriteProperty(101,AV27,Present_Value,X)
      t5. WriteProperty(103,AO5,Present_Value,X)
      t3+100ms. WriteProperty(102,AO14,Present_Value,X)
      t3+200ms. WriteProperty(104,AV123,Present_Value,X)
      t3+200ms+y  Write_Status = SUCCESSFUL
      t3+200ms+y  Reliability = NO_FAULT_DETECTED
}
```

Figure 12-12. Channel Object Execution Timeline

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.53.5.1 Datatype Coercion of Present_Value

Since List_Of_Object_Property_References can include object properties of different data types, the value written to Present_Value may require coercion to another datatype. The rules governing how these coercions occur are summarized in Table 12-63. Those cases where Invalid Datatype (ID) is indicated in Table 12-63, and those cases where coercion of values exceeds a range specified by an indicated coercion rule, shall be considered as coercion failures and the write shall not occur. In those cases where No Coercion (NC) is indicated in Table 12-63, the coercion shall be considered as successful. If any of the writes to the List_Of_Object_Property_References produces a failure then Write_Status shall indicate FAILED.

© ISO 2022 – All rights reserved

Table 12-63 - Datatype Coercion Rules

Datatype in Present_Value write	Datatype of referenced property													
	unknown	BOOLEAN	Unsigned	INTEGER	REAL	Double	OCTET STRING	CharacterString	BIT STRING	ENUMERATED	Date	Time	BACnetObjectIdentifier	BACnetLightingCommand
NULL	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	ID
BOOLEAN	NC	NC	2	2	2	2	ID	ID	ID	2	ID	ID	ID	ID
Unsigned	NC	1	NC	3	3	3	ID	ID	ID	NC	ID	ID	NC	ID
INTEGER	NC	1	4	NC	4	4	ID	ID	ID	4	ID	ID	ID	ID
REAL	NC	1	5	5	NC	5	ID	ID	ID	5	ID	ID	ID	ID
Double	NC	1	6	6	6	NC	ID	ID	ID	6	ID	ID	ID	ID
OCTET STRING	NC	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID	ID
CharacterString	NC	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID
BIT STRING	NC	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID
ENUMERATED	NC	1	NC	3	3	3	ID	ID	ID	NC	ID	ID	ID	ID
Date	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID
Time	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID
BACnetObjectIdentifier	NC	ID	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID
BACnetLightingCommand	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC

NC=No Coercion ID=Invalid Datatype

12.53.5.2 Coercion Rule 1 - Numeric to BOOLEAN

The numeric value 0 maps to FALSE and anything else is TRUE.

12.53.5.3 Coercion Rule 2 - BOOLEAN to Numeric

The BOOLEAN value FALSE is mapped to 0 and TRUE is mapped to 1.

12.53.5.4 Coercion Rule 3 - Unsigned to Numeric

The Unsigned value is mapped directly to the target datatype. The Unsigned value shall be limited to 2147483647. The REAL value shall be limited in precision to seven significant digits. Values outside this limit shall cause Write_Status to indicate FAILED when the List_Of_Object_Property_References has been completely processed.

12.53.5.5 Coercion Rule 4 - INTEGER to Numeric

The INTEGER value is mapped directly to the target datatype. The Unsigned value shall be limited to 0 to 2147483647. The REAL value shall be limited in precision to seven significant digits. Values outside these limits shall cause Write_Status to indicate FAILED when the List_Of_Object_Property_References has been completely processed.

12.53.5.6 Coercion Rule 5 - REAL to Numeric

The REAL value is mapped directly to the target datatype. The Unsigned value shall be limited to 0 to 2147483000. The INTEGER value shall be limited to -2147483000 to 214783000. Values outside these limits shall cause Write_Status to indicate FAILED when the List_Of_Object_Property_References has been completely processed.

12.53.5.7 Coercion Rule 6 - Double to Numeric

The Double value is mapped directly to the target datatype. The Unsigned value shall be limited to 0 to 2147483000. The INTEGER value shall be limited to -2147483000 to 214783000. The REAL value shall be limited to $3.4*10^{\pm 38}$. Values

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

outside these limits shall cause Write_Status to indicate FAILED when the List_Of_Object_Property_References has been completely processed.

12.53.5.8 Handling of Coercion Failures

In any case of coercion failure the Write_Status shall indicate FAILED and the write shall not occur. The List_Of_Object_Property_References shall be processed in its entirety even if one or more coercion failures occur.

12.53.6 Last_Priority

This read-only property, of type Unsigned, shall convey the priority at which the Present_Value was most recently written (1..16). If an attempt was made to write to the Present_Value without the 'Priority' parameter, a default priority of 16 (the lowest priority) shall be assumed. The initial value of Last_Priority shall be 16.

12.53.7 Write_Status

This property, of type BACnetWriteStatus, shall be set to IDLE initially. This property shall be set to IN_PROGRESS when a value is written to the Present_Value property indicating that the Channel object has begun processing the List_Of_Object_Property_References.

Once all of the writes have been attempted by the Channel object, the Write_Status property shall be set to either SUCCESSFUL or FAILED. The SUCCESSFUL value indicates that the Channel object has processed all of the properties in List_Of_Object_Property_References and did not have any coercion errors, and did not receive any errors, rejects, or aborts. The FAILED value indicates that the Channel object has processed all of the properties in List_Of_Object_Property_References and encountered a coercion failure, or received an error, reject, or abort for at least one of the writes. A special exception shall be the writing of a NULL value. If a NULL value is written and WriteProperty or WritePropertyMultiple services subsequently receive an ERROR INVALID_DATATYPE or REJECT INVALID_PARAMETER_DATA_TYPE, it shall not be treated as a FAILED value. This is specifically to allow Channel objects to point to both commandable and non-commandable properties with the same channel.

If List_Of_Object_Property_References is empty, this property shall remain set to IDLE.

12.53.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Channel object. Two of the flags are associated with the values of another property of this object. A more detailed status could be determined by reading the property that is linked to this flag. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	The value of this flag shall be logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context, "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.53.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the object is "reliable" as far as the BACnet device or operator can determine. If the Write_Status property indicates FAILED, the value of the Reliability property shall provide an indication of the type of failure that occurred. If one or more member values cannot be written because

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

of a communication failure, the value of the Reliability property shall be COMMUNICATION_FAILURE. If one or more member values cannot be written because of invalid or inconsistent configuration, the value of the Reliability property shall be CONFIGURATION_ERROR. Other errors that may occur during the processing of writes to Present_Value shall be PROCESS_ERROR conditions. If the conditions for a PROCESS_ERROR, CONFIGURATION_ERROR, or COMMUNICATION_FAILURE are present at the same time, or some other error condition occurs, the selection of which value to use shall be a local matter.

12.53.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the forwarding mechanism that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the forwarding mechanism;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the forwarding mechanism;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred and been passed on to the forwarding mechanism.

12.53.11 List_Of_Object_Property_References

This property, of type BACnetARRAY of BACnetDeviceObjectPropertyReference, specifies the Device Identifiers, Object Identifiers, and Property Identifiers of the properties to be written with the same value that is written to Present_Value.

This property may be restricted to only support references to objects inside of the device containing the Channel object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property using WriteProperty service shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property is set to reference an object outside the device containing the Channel object, the method used for writing to the referenced property value for the purpose of controlling the property is a local matter. If an implementation chooses to use WritePropertyMultiple as the preferred method of writing to the referenced property, then the device containing the Channel object shall be capable of using WriteProperty to complete writes to devices that do not support WritePropertyMultiple, or that fail before completing all required writes. If WritePropertyMultiple fails for one element, the remaining elements shall be retried as WritePropertyMultiple or WriteProperty as a local matter.

12.53.11.1 Empty References

Elements of the List_Of_Object_Property_References array containing object or device instance numbers equal to 4194303 are considered to be 'empty' or 'uninitialized'.

12.53.11.2 Initializing New Array Elements When the Array Size is Increased

If the size of this array is increased by writing to array index zero, each new array element shall contain an empty reference. The size of Execution_Delay shall be automatically increased to be the same.

12.53.12 Execution_Delay

This property, of type BACnetARRAY of Unsigned, shall indicate an execution delay in milliseconds for each value to be written in the List_Of_Object_Property_References when the Channel object's Present_Value is written. A value of zero indicates no delay. A non-zero execution delay value shall cause a delay, by that many milliseconds, in the writing to the corresponding referenced value. The resolution of Execution_Delay shall be a local matter. If present, the Execution_Delay property shall be writable. All delay periods shall "start" at the same time. So, a write of A, B(delay 100), C, D(delay 200) shall immediately write A and C, but delay the writing of B by 100 milliseconds and D by 200 milliseconds. Multiple delayed values shall execute their corresponding delays in parallel (see Figure 12-12).

12.53.12.1 Initializing New Array Elements When the Array Size is Increased

If the size of this array is increased by writing to array index zero, each new array element shall contain zero. The size of List_Of_Object_Property_References shall be automatically increased to be the same.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.53.13 Allow_Group_Delay_Inhibit

This property, of type BOOLEAN, shall indicate whether WriteGroup service writes to this object, that specify 'Inhibit Delay'=TRUE, may override any execution delay specified in this object. Execution_Delay shall always occur as the result of WriteProperty or WritePropertyMultiple. In the case of WriteGroup, Execution_Delay shall always occur unless the WriteGroup service parameter 'Inhibit Delay' is TRUE, and the Channel object property Allow_Group_Delay_Inhibit is present and has the value TRUE.

12.53.14 Channel_Number

This property, of type Unsigned16, shall indicate the logical channel number that this Channel object is associated with when the Channel object Present_Value is written to using the WriteGroup service.

12.53.15 Control_Groups

This property, of type BACnetARRAY of Unsigned32, shall indicate those logical control groups of which this Channel object is a member. This array shall contain at least one entry. Unused array slots shall contain the value zero, and control group zero shall mean "no assignment." Control_Groups is required to be writable, and it shall be permitted to configure the membership of the Channel object in arbitrary groups by writing the control group numbers into this array in any order, up to the maximum number of simultaneous groups supported by the Channel object. Duplicate entries specifying the same group number shall be permitted. The maximum size of the Control_Groups array shall be a local matter.

12.53.16 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.53.17 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.53.18 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.53.19 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.53.20 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.53.21 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.53.22 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.53.23 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.53.24 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.53.25 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.53.26 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.53.27 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.53.28 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.53.29 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.53.30 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.53.31 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.53.32 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.53.33 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.54 Lighting Output Object Type

The Lighting Output object type defines a standardized object whose properties represent the externally visible characteristics of a lighting output and includes dedicated functionality specific to lighting control that would otherwise require explicit programming. The lighting output is analog in nature.

The physical output level, or non-normalized range, is specified as the linearized percentage (0..100%) of the possible light output range with 0.0% being off, 1.0% being dimmest, and 100.0% being brightest. The actual range represents the subset of physical output levels defined by Min_Actual_Value and Max_Actual_Value (or 1.0 to 100.0% if these properties are not present). The normalized range is always 0.0 to 100.0% where 1.0% = bottom of the actual range and 100.0% = top of the actual range. All 0.0% to 100.0% properties of the Lighting Output object shall use the normalized range except for Min_Actual_Value and Max_Actual_Value. If Min_Actual_Value and Max_Actual_Value are not present, then the normalized and non-normalized ranges shall be the same.

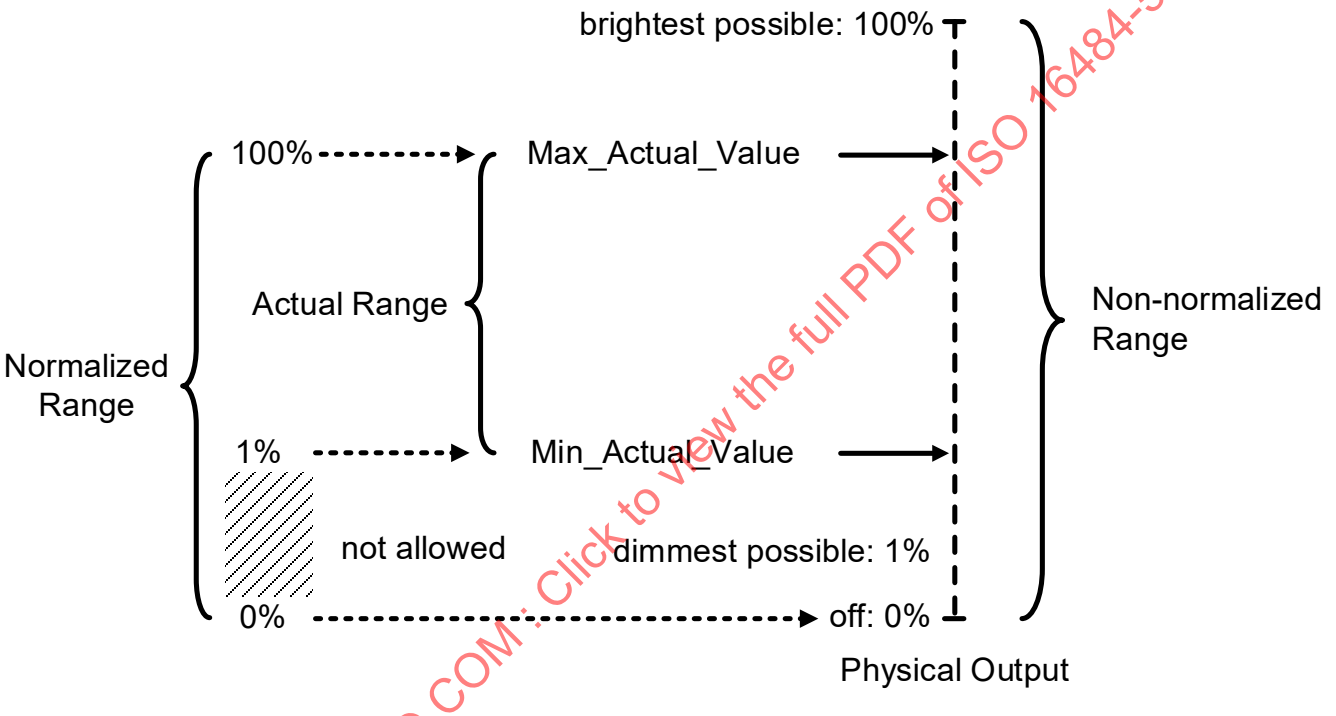


Figure 12-13. Normalized Range of the Lighting Output

The level of the lights can be changed directly to an absolute level by writing to the Present_Value. This property is commandable and uses a priority array to arbitrate between multiple writers to the lighting output.

The level of the lights may also be changed by writing to the Lighting_Command property. The lighting command provides additional lighting functionality with special lighting-specific functions such as ramping, stepping, and fading.

The Lighting_Command also provides a blink-warn mechanism to notify room occupants that the lights are about to turn off. The blink-warning mechanism is internal to the lighting output and may cause the physical lights to blink on and off or issue a notification in some other manner. The blink-warn commands come in three different variants. The WARN command causes a blink-warn notification but then leaves the level of the lights unaffected. The WARN_RELINQUISH command executes the blink-warn notification, then keeps the light at the current level for a predetermined amount of time (egress time), and then relinquishes the value at the given priority. The WARN_OFF command executes the blink-warn notification, then keeps the light at the current level for the egress time, and then writes 0.0% (off) at the given priority. See Table 12-67.

The following example illustrates how a Lighting Output object may be used in a typical office scenario. Prior to 7:00 AM the lights are off as the Lighting Output object is being controlled at the relinquish default value (0.0%). At 7:00 AM a scheduler (e.g., a BACnet Schedule object or other automated process) turns the physical lights on by writing 100.0% to the Present_Value property at priority 9. At 6:00 PM a WARN_RELINQUISH lighting command is executed at priority 9, which causes an immediate blink-warn notification to occur but leaves the lights on until the egress timer has expired.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Assuming a 10 minute (600 seconds) egress time is specified, the value at priority 9 is relinquished at 6:10 PM. This scenario is shown in Figure 12-14.

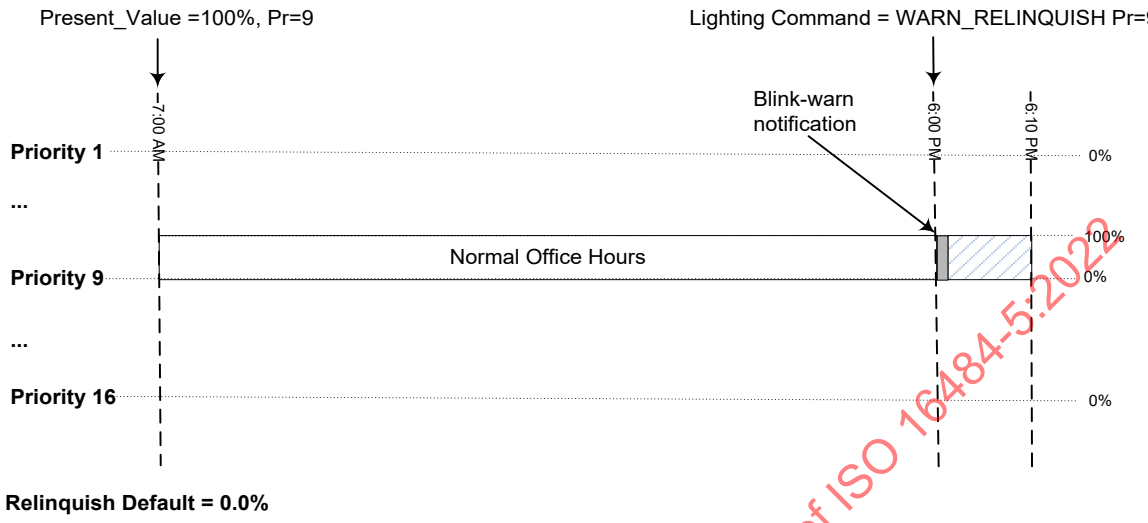


Figure 12-14. Daily Schedule with Blink-Warn Example

Lighting Output objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Lighting Output objects that support intrinsic reporting shall apply the NONE event algorithm.

The object and its properties are summarized in Table 12-64 and described in detail in this clause.

Table 12-64. Properties of the Lighting Output Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	W
Tracking_Value	REAL	R
Lighting_Command	BACnetLightingCommand	W
In_Progress	BACnetLightingInProgress	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[1]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Blink_Warn_Enable	BOOLEAN	R
Egress_Time	Unsigned	R
Egress_Active	BOOLEAN	R
Default_Fade_Time	Unsigned	R
Default_Ramp_Rate	REAL	R
Default_Step_Increment	REAL	R
Transition	BACnetLightingTransition	O
Feedback_Value	REAL	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	REAL	R
Power	REAL	O
Instantaneous_Power	REAL	O
Min_Actual_Value	REAL	O[2]
Max_Actual_Value	REAL	O[2]
Lighting_Command_Default_Priority	Unsigned	R

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
COV_Increment	REAL	O[3]
Event_Detection_Enable	BOOLEAN	O[1,4]
Notification_Class	BACnetARRAY[3] of BACnetTimeStamp	O[1,4]
Event_Enable	BACnetEventTransitionBits	O[1,4]
Acked_Transitions	BACnetEventTransitionBits	O[1,4]
Notify_Type	BOOLEAN	O[1,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[5]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.
[2] If either of these properties is present, they shall both be present, and they shall be writable.
[3] This property is required if, and shall be present only if, the object supports COV reporting for the Present_Value property.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] If this property is present, then the Reliability property shall be present.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.
[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.54.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.54.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.54.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LIGHTING_OUTPUT.

12.54.4 Present_Value (Commandable)

This property, of type REAL, specifies the target value, in percent, for the lighting output within the normalized range. The valid range of values for the Present_Value is 0.0% to 100.0%. Writes to Present_Value at a value greater than 0.0% but less than 1.0% shall be clamped to 1.0%.

© ISO 2022 – All rights reserved

Present_Value may also be affected by writes to the Lighting_Command property that initiate lighting commands. These commands may asynchronously affect the lighting output by establishing a new target for Present_Value and carrying out the requested operation. When a ramp or fade lighting command is in progress, the Present_Value shall indicate the target level of the operation and not the current value. The current value is always indicated in the Tracking_Value property. If a lighting command is currently in progress and the Present_Value is written at a higher or equal priority, the lighting command shall be halted (see Clause 12.54.6.1 Halting a Lighting Command in Progress).

The Present_Value supports special values outside of the normal range of values to provide blink-warn functionality from objects and devices that are unable to write the complex datatypes used in the Lighting Command property (e.g., the BACnet Schedule object type). The special values of the Present_Value are summarized in Table 12-65.

Table 12-65. Special Values of the Present_Value Property

Value	Description
-1.0	Provides the same functionality as the WARN lighting command.
-2.0	Provides the same functionality as the WARN_RELINQUISH lighting command.
-3.0	Provides the same functionality as the WARN_OFF lighting command.

Writing a special value has the same effect as writing the corresponding lighting command and is subject to the same restrictions. See Table 12-67. The special value itself is not written to the priority array.

The physical lighting output shall be updated whenever the Present_Value is commanded or changed as a result of executing a lighting command. However, when the device starts up or is reset, it is a local matter as to whether the physical lighting output is updated with the current value of Present_Value or whether the value of the physical output before startup or reset is retained. When the physical output is not updated at startup or reset, the property In_Progress shall be set to NOT_CONTROLLED until the physical output is updated with the current value of Present_Value. Writes to Present_Value of values outside of the valid range of values shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.5 Tracking_Value

This property, of type REAL, indicates the value at which the physical lighting output is being controlled within the normalized range at all times.

When the value of In_Progress is IDLE, Tracking_Value shall be equal to Present_Value.

When the value of In_Progress is RAMP_ACTIVE or FADE_ACTIVE, Tracking_Value shall indicate the current calculated value of the ramp or fade algorithm. The manner by which the Tracking_Value is calculated in this situation shall be a local matter.

When the value of In_Progress is NOT_CONTROLLED or OTHER, the value of Tracking_Value shall be a local matter.

12.54.6 Lighting_Command

This property, of type BACnetLightingCommand, is used to request special lighting commands with specific behaviors. Lighting_Command is written with compound values that specify particular lighting operations. Devices containing Lighting Output objects shall support all BACnetLightingOperations shown in Table 12-67.

When a lighting operation is written to the Lighting_Command property, the effect of that operation is written to the Present_Value at the priority level specified by the priority field. If the priority field is not included with the command, the priority specified in Lighting_Command_Default_Priority shall be used.

Some lighting operations require additional parameters. These are provided by optional fields of the BACnetLightingCommand value.

The fields of the BACnetLightingCommand are summarized in Table 12-66.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 12-66. BACnetLightingCommand Fields

Field	Description
operation	This field is an enumeration of type BACnetLightingOperation that defines the lighting operation desired.
target-level	This field, of type REAL, represents the target lighting output level in the normalized range (0.0%...100.0%).
ramp-rate	This field, of type REAL, represents the rate of change in percent-per-second for ramp operations. The range of allowable ramp-rate values is 0.1 to 100.0 inclusive. If this field is not specified, then the value of Default_Ramp_Rate specifies the ramp rate to be used
step-increment	This field, of type REAL, represents the percent amount to be added to Present_Value when stepping. The range of allowable values is 0.1% to 100.0% inclusive. If this field is not specified, then the value of Default_Step_Increment specifies the step increment to be used.
fade-time	This field, of type Unsigned, represents the time in milliseconds over which fade operations take place. The range of allowable fade-time values is 100 ms to 86400000 ms (1 day) inclusive. If this field is not specified, then the value of Default_Fade_Time specifies the fade time to be used.
priority	This field, of type Unsigned (1..16), represents the priority values 1 (highest priority) through 16 (lowest priority). If this field is not specified, then the value of Lighting_Command_Default_Priority specifies the priority to be used.

The lighting commands are described in Table 12-67. The notation to specify the syntax of the lighting commands is as follows:

<field in angle brackets>	required field of the BACnetLightingCommand
<field in angle brackets = value>	required field of the BACnetLightingCommand with a specified value
[fields in square brackets]	optional fields of the BACnetLightingCommand.

Table 12-67. Lighting Commands

Operation	Description
NONE	This operation is used to indicate that no lighting command has been written to the Lighting_Command property. This operation shall not be written to the Lighting_Command property. Attempts to write this operation to the Lighting_Command property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE
FADE_TO	Commands Present_Value to fade from the current Tracking_Value to the target-level specified in the command at the specified priority. The fade operation is implemented by first writing target-level to the specified slot in the priority array. If the fade command is the highest priority when written, then the fade operation continues by changing the physical lighting output proportionally from its current value to target-level, over a period of time defined by fade-time. While the fade operation is executing, In_Progress shall be set to FADE_ACTIVE, and Tracking_Value shall be updated to reflect the current progress of the fade. If the fade command is not the highest priority when written, then this fade operation is not executed. syntax: <operation = FADE_TO> <target-level> [priority] [fade-time]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Operation	Description
RAMP_TO	Commands Present_Value to ramp from the current Tracking_Value to target-level specified in the command at the specified priority. The ramp operation is implemented by first writing target-level to the specified slot in the priority array. If the ramp command is the highest priority when written, then the ramp operation continues by changing the physical lighting output proportionally from its current value to target-level, with a fixed rate of change defined by ramp-rate. While the ramp operation is executing, In_Progress shall be set to RAMP_ACTIVE, and Tracking_Value shall be updated to reflect the current progress of the fade. If the ramp command is not the highest priority when written, then this ramp operation is not executed. syntax: <operation = RAMP_TO> <target-level> [priority] [ramp-rate]
STEP_UP	Commands Present_Value to a value equal to the Tracking_Value plus the step-increment at the specified priority. The step-up operation is implemented by writing the Tracking_Value plus step-increment to the specified slot in the priority array. If the result of the addition is greater than 100.0%, the value shall be set to 100.0%. If the starting level of Tracking_Value is 0.0%, then this operation is ignored. syntax: <operation = STEP_UP> [priority] [step-increment]
STEP_DOWN	Commands Present_Value to a value equal to the Tracking_Value minus the step-increment at the specified priority. The step-down operation is implemented by writing the Tracking_Value minus step-increment to the specified slot in the priority array. If the result of the subtraction is less than 1.0%, the value shall be set to 1.0%. If the starting level of Tracking_Value is 0.0%, then this operation is ignored. syntax: <operation = STEP_DOWN> [priority] [step-increment]
STEP_ON	Same as STEP_UP except when Tracking_Value is 0.0%, in which case, 1.0% is written to the specified slot in the priority array. syntax: <operation = STEP_ON> [priority] [step-increment]
STEP_OFF	Same as STEP_DOWN except when Tracking_Value is 1.0%, in which case, 0.0% is written to the specified slot in the priority array. syntax: <operation = STEP_OFF> [priority] [step-increment]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Operation	Description
WARN	Executes a blink-warn notification at the specified priority. After the blink-warn notification has been executed, the value at the specified priority is unchanged. The blink-warn notification shall not occur if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The value at the specified priority is 0.0%, or (c) Blink_Warn_Enable is FALSE. See Clause 12.54.6.2 Blink-Warn Behavior. syntax: <operation = WARN> [priority]
WARN_RELINQUISH	Executes a blink-warn notification at the specified priority and then relinquishes the value at the specified priority slot after a delay of Egress_Time seconds. The blink-warn notification shall not occur, and the value at the specified priority shall be relinquished immediately if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The value at the specified priority is 0.0% or NULL, or (c) The value of the next highest non-NULL priority, including Relinquish_Default, is greater than 0.0%, or (d) Blink_Warn_Enable is FALSE. See Clause 12.54.6.2 Blink-Warn Behavior. syntax: <operation = WARN_RELINQUISH> [priority]
WARN_OFF	Executes a blink-warn notification at the specified priority and then writes the value 0.0% to the specified slot in the priority array after a delay of Egress_Time seconds. The blink-warn notification shall not occur and the value 0.0% written at the specified priority immediately if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The Present_Value is 0.0%, or (c) Blink_Warn_Enable is FALSE See Clause 12.54.6.2 Blink-Warn Behavior. syntax: <operation = WARN_OFF> [priority]
STOP	Stops any FADE_TO or RAMP_TO command in progress at the specified priority and writes the current value of Tracking_Value to that slot in the priority array and sets In_Progress to IDLE. Cancels any WARN_RELINQUISH or WARN_OFF command in progress at the specified priority and cancels the blink-warn egress timer. The value in the priority array at the specified priority remains unchanged. If there is no fade, ramp, or warn command currently executing at the specified priority, then this operation is ignored. syntax: <operation = STOP> [priority]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Some lighting devices may incorporate remote subnetworks or other technology that may introduce latency or non-linearity in the behavior of the physical light being controlled. Consequently, the absolute timing resolution of lighting operations should not be assumed. Some lighting devices may not be capable of achieving the performance implied by a given operation, in which case, the device shall use its best effort to carry out the intended operation.

The Lighting_Command property shall indicate the last written value or NONE if it has not yet been written.

If a BACnetLightingCommand is sent that includes an optional field that is not explicitly described for that operation in Table 12-67, then the field value shall be ignored. Lighting commands written with a required or optional field, explicitly specified for this command, which are outside of the allowable range of values, shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.6.1 Halting a Lighting Command in Progress

Some lighting commands (i.e. RAMP_TO, FADE_TO, WARN_RELINQUISH, and WARN_OFF) are executed over a period of time. While a lighting command, at a specific priority, is in progress, it shall be halted under the following conditions:

(a) A valid command, other than STOP, is written to the Lighting_Command property at a higher priority than the command in progress, or
(b) The Present_Value is written at a higher priority than the command in progress.

When a RAMP_TO or FADE_TO command that is currently in progress is halted, the internal ramp or fade algorithm is halted, and the corresponding slot in the priority array remains unchanged.

When a WARN_RELINQUISH command that is currently in progress is halted, the blink-warn egress timer is immediately expired, and the corresponding value of the priority array is relinquished.

When a WARN_OFF command that is currently in progress is halted, the egress timer is immediately expired, and the value 0.0% is written to the priority array at the specified priority.

A lighting command that is in progress is implicitly halted when it is overwritten by another lighting command at the same priority or by a write to the Present_Value at the same priority.

There may only be one lighting command currently in progress at any time.

12.54.6.2 Blink-Warn Behavior

The WARN, WARN_RELINQUISH, and WARN_OFF lighting commands, as well as writing one of the special values to the Present_Value property, cause a blink-warn notification to occur at the specified priority. A blink-warn notification is used to warn the occupants that the lights are about to turn off, giving the occupants the opportunity to exit the space or to override the lights for a period of time.

The actual blink-warn notification mechanism shall be a local matter. The physical lights may blink once, multiple times, or repeatedly. They may also go bright, go dim, or signal a notification through some other means. In some circumstances, no blink-warn notification will occur at all. The blink-warn notification shall not be reflected in the priority array or the tracking value.

The WARN_RELINQUISH and WARN_OFF lighting commands include an egress time in which the lights are held at the current level until the egress time expires or the command is halted. The number of seconds for egress is specified in the Egress_Time property. The egress timer shall start when the WARN_RELINQUISH or WARN_OFF command is written. While the egress timer is active, the property Egress_Active shall be set to TRUE. When the egress timer expires or the command is halted, then Egress_Active shall be set to FALSE. There may only be one priority slot with an active egress timer at any time.

If the Blink_Warn_Enable property has the value FALSE, then the blink-warn notification shall not occur, and the effect of the operation shall occur immediately without an egress delay.

The relationship between Egress_Time and the Egress_Active property is shown in Figure 12-15.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

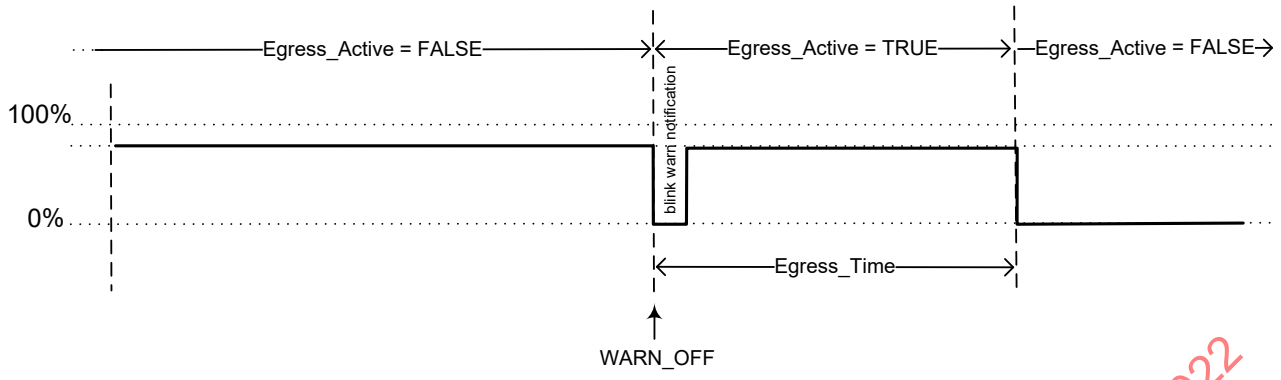


Figure 12-15. Blink-warn and Egress Time

12.54.7 In_Progress

This property, of type BACnetLightingInProgress, shall indicate processes in the lighting output object that may cause the Tracking_Value and Present_Value to differ temporarily. The processes indicated in the property are summarized in Table 12-68.

Table 12-68. BACnetLightingInProgress Values

Value	Description
IDLE	The default value that indicates that no processes are executing which would cause the Present_Value and Tracking_Value to differ.
RAMP_ACTIVE	Indicates that a ramp lighting command is currently being executed.
FADE_ACTIVE	Indicates that a fade lighting command is currently being executed.
NOT_CONTROLLED	Indicates that on startup or reset the physical output has not been updated with the current value of Present_Value.
OTHER	Indicates that the Tracking_Value and Present_Value may differ but none of the other conditions describe the nature of the process.

12.54.8 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.54.9 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Lighting Output object. Two of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).

FAULT — Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

OVERRIDDEN — Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context, "overridden" is taken to mean that the Present_Value property is not changeable through BACnet services. Otherwise, the value is logical FALSE (0).

OUT_OF_SERVICE — Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.54.10 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.54.11 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

12.54.12 Out_Of_Service

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the physical point that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the physical output;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical output;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred to the physical output;
(e) the Present_Value property shall still be controlled by the BACnet command prioritization mechanism and lighting command (see Clause 19).

12.54.13 Blink_Warn_Enable

This property, of type BOOLEAN, specifies whether a blink-warn is executed (TRUE) or not (FALSE) when a WARN, WARN_RELINQUISH, or WARN_OFF command is written to the Lighting_Command property or one of the special values is written to the Present_Value. When this property is FALSE and a warn operation is written, a blink-warn notification shall not occur, and the effect of the operation shall occur immediately without an egress delay.

12.54.14 Egress_Time

This property, of type Unsigned, specifies the egress time in seconds when a WARN_RELINQUISH or WARN_OFF value is written to the Lighting_Command property or when the special values -2.0 or -3.0 are written to the Present_Value property. The egress time is the time for which the light level is held at its current level before it is relinquished or set to 0.0%.

12.54.15 Egress_Active

This property, of type BOOLEAN, shall be TRUE whenever the Egress_Time for a WARN_RELINQUISH or WARN_OFF lighting operation is in effect and FALSE otherwise.

12.54.16 Default_Fade_Time

This property, of type Unsigned, indicates the amount of time in milliseconds over which changes to the normalized value reflected in the Tracking_Value property of the lighting output shall occur when the Lighting_Command property is written with a fade request that does not include a fade-time value. The range of allowable fade-time values is 100 ms to 86400000 ms (1 day) inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.17 Default_Ramp_Rate

This property, of type REAL, indicates the rate in percent-per-second at which changes to the normalized value reflected in the Tracking_Value property of the lighting output shall occur when the Lighting_Command property is written with

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

a ramp request that does not include a ramp-rate value. The range of allowable ramp-rate values is 0.1 %/s to 100.0 %/s inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.18 Default_Step_Increment

This property, of type REAL, indicates the amount to be added to the Tracking_Value when the Lighting_Command property is written with a step request that does not include a step-increment value. The range of allowable values is 0.1% to 100.0% inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.19 Transition

This property, of type BACnetLightingTransition, specifies how a change in the Present_Value transitions from the current level to the target level. A transition comes into effect when the Present_Value is directly commanded or when the current highest priority value has been relinquished. Writing the Lighting commands FADE_TO, RAMP_TO, STEP_ON, STEP_OFF, STEP_UP, or STEP_DOWN shall ignore the Transition property.

The transition may be one of NONE, FADE, or RAMP. The transition NONE causes the Present_Value to immediately be set to the target level when the highest priority value has been relinquished. If this property does not exist, then the transition type shall be assumed to be NONE.

FADE or RAMP transitions allow a smooth transition of the lighting level when the Present_Value changes. A FADE transition executes a fade operation from the Tracking_Value to the target level using the fade time specified in Default_Fade_Time. A RAMP transition executes a ramp operation from the Tracking_Value to the target level using the ramp rate specified in Default_Ramp_Rate.

When a transition results in an operation that may cause the Tracking_Value to differ from the Present_Value, then the In_Progress property shall be set to the value that reflects the operation in progress.

12.54.20 Feedback_Value

This property, of type REAL, shall indicate the actual value of the physical lighting output within the normalized range.

If the actual value of the physical lighting output in the non-normalized range is not off but is less than the Min_Actual_Value, then Feedback_Value shall be set to 1.0%. If the actual value in the non-normalized range is greater than Max_Actual_Value, then Feedback_Value shall be set to 100.0%.

The manner by which the Feedback_Value is determined shall be a local matter.

12.54.21 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.54.22 Relinquish_Default

This property, of type REAL, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. See Clause 19.2.

12.54.23 Power

This property, of type REAL, is the nominal power consumption of the load(s) controlled by this object when the light level is 100.0% of the non-normalized range. The units shall be kilowatts.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.54.24 Instantaneous_Power

This property, of type REAL, is the nominal power consumption of the load(s) controlled by this object at this moment. The units shall be kilowatts.

12.54.25 Min_Actual_Value

This property, of type REAL, shall specify the physical output level that corresponds to a Present_Value of 1.0%. Changing Min_Actual_Value to a value greater than Max_Actual_Value shall force Max_Actual_Value to become equal to Min_Actual_Value. Min_Actual_Value shall always be a positive number in the range 1.0% to 100.0%.

12.54.26 Max_Actual_Value

This property, of type REAL, shall specify the physical output level that corresponds to a Present_Value of 100.0%. Changing Max_Actual_Value to a value less than Min_Actual_Value shall force Min_Actual_Value to become equal to Max_Actual_Value. Max_Actual_Value shall always be a positive number in the range 1.0% to 100.0%.

12.54.27 Lighting_Command_Default_Priority

This property, of type Unsigned, shall specify a write priority of 1 to 16 that indicates the element of the Priority_Array controlled by the Lighting_Command property when the BACnetLightingCommand priority field is absent.

The priority value 6 shall not be used for this property. If a value of 6 is written to this property, then a Result(-) shall be returned with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.54.28 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.54.29 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.54.30 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.54.31 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.54.32 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.54.33 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.54.34 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.54.35 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.54.36 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.54.37 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.54.38 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.54.39 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

12.54.40 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.54.41 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.54.42 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.54.43 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.54.44 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.54.45 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.54.46 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.54.47 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.54.48 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.54.49 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.55 Binary Lighting Output Object Type

The Binary Lighting Output object type defines a standardized object whose properties represent the externally visible characteristics of a binary lighting output and includes dedicated functionality specific to lighting control that would otherwise require explicit programming.

This object is binary in nature and can be either in a logical on or off state. The object is commanded by writing to the Present_Value property. This property is commandable and uses a priority array to arbitrate between multiple writers to the binary lighting output.

The binary lighting output also provides a blink-warn mechanism to notify room occupants that the lights are about to turn off. The blink-warning mechanism is internal to the object and may cause the physical lights to blink on and off or issue a notification in some other manner. A blink-warn command is issued by writing a special value to the Present_Value. The blink-warn command comes in three different variants. The WARN command causes a blink-warn notification but then leaves the level of the lights unaffected. The WARN_RELINQUISH command executes the blink-warn notification and then keeps the lights ON for a predetermined amount of time (egress time) and then relinquishes the value at the given priority. The WARN_OFF command executes the blink-warn notification and then keeps the lights ON and then writes OFF at the given priority.

The following example illustrates how a Binary Lighting Output object may be used in a typical office scenario. Prior to 7:00 am the lights are off as the Lighting Output object is being controlled at the relinquish default value (OFF). At 7:00 am a scheduler (e.g., a BACnet Schedule object or other automated process) turns the physical lights on by writing ON to the Present_Value property at priority 9. At 6:00 pm a WARN_RELINQUISH lighting command is executed at priority 9, which causes an immediate blink-warn notification to occur but leaves the lights on until the egress timer has expired. Assuming a 10 minute (600 seconds) egress time is specified, the value at priority 9 is relinquished at 6:10 pm. This scenario is shown in Figure 12-16.

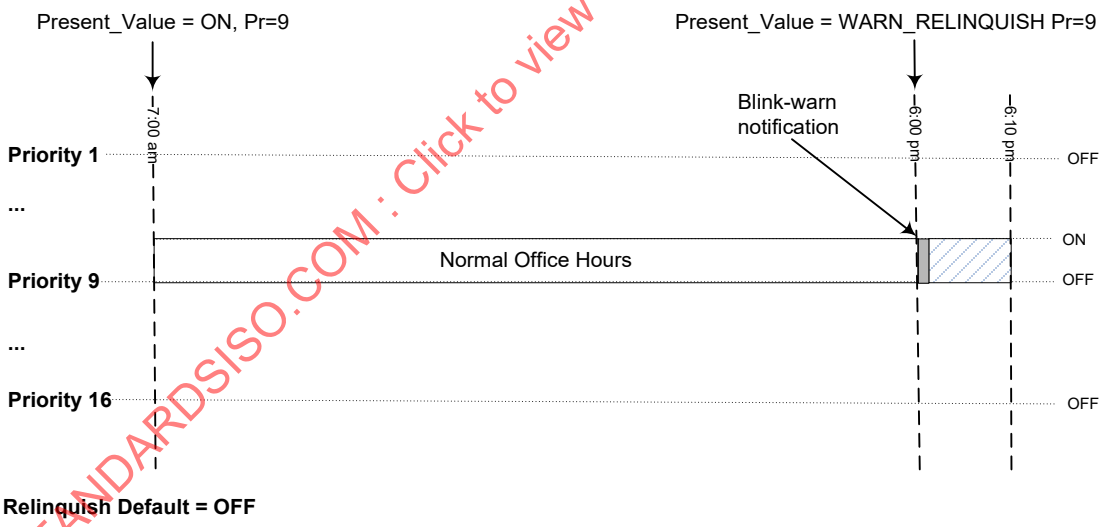


Figure 12-16. Daily Schedule with Blink-Warn Example

Binary Lighting Outputs may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Binary Lighting Output objects that support intrinsic reporting shall apply the NONE event algorithm.

The object and its properties are summarized in Table 12-69 and described in detail in this clause.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Table 12-69. Properties of the Binary Lighting Output Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetBinaryLightingPV	W
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[1]
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Blink_Warn_Enable	BOOLEAN	R
Egress_Time	Unsigned	R
Egress_Active	BOOLEAN	R
Feedback_Value	BACnetBinaryLightingPV	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	BACnetBinaryLightingPV	R
Power	REAL	O
Polarity	BACnetPolarity	O
Elapsed_Active_Time	Unsigned32	O[2]
Time_Of_Active_Time_Reset	BACnetDateTime	O[2]
Strike_Count	Unsigned	O[3]
Time_Of_Strike_Count_Reset	BACnetDateTime	O[3]
Event_Detection_Enable	BOOLEAN	O[1,4]
Notification_Class	Unsigned	O[1,4]
Event_Enable	BACnetEventTransitionBits	O[1,4]
Acked_Transitions	BACnetEventTransitionBits	O[1,4]
Notify_Type	BACnetNotifyType	O[1,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[5]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Current_Command_Priority	BACnetOptionalUnsigned	R
Value_Source	BACnetValueSource	O[6,8,10]
Value_Source_Array	BACnetARRAY[16] of BACnetValueSource	O[7,9]
Last_Command_Time	BACnetTimeStamp	O[7,9]
Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O[9]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Audit_Priority_Filter	BACnetOptionalPriorityFilter	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting
[2] If one of the optional properties Elapsed_Active_Time or Time_Of_Active_Time_Reset is present, then both of these properties shall be present.
[3] If one of the optional properties Strike_Count or Time_Of_Strike_Count_Reset is present, then both of these properties shall be present.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] If this property is present, then the Reliability property shall be present.
[6] This property is required if the object supports the value source mechanism.
[7] These properties are required if the object supports the value source mechanism and is commandable.
[8] This property shall be present only if the object supports the value source mechanism.
[9] These properties shall be present only if the object supports the value source mechanism and is commandable.
[10] This property shall be writable as described in Clause 19.5.
[11] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.55.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.55.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.55.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be BINARY_LIGHTING_OUTPUT.

12.55.4 Present_Value (Commandable)

This property, of type BACnetBinaryLightingPV, reflects the logical state of the Binary Lighting Output. The logical state of the output shall be either ON or OFF.

The Present_Value also supports special values to provide blink-warn functionality. Each special value written corresponds to a specific operation which has an effect on the present value; however, the special value itself is not written to the priority array. The special values of the Present_Value are summarized in Table 12-70.

Table 12-70. Special Values of the Present_Value Property

Value	Description
WARN	Executes a blink-warn notification at the specified priority. After the blink-warn notification has been executed the value at the specified priority remains ON. The blink-warn notification shall not occur if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The value at the specified priority is OFF, or (c) Blink_Warn_Enable is FALSE. (see Clause 12.55.10.1 Blink-Warn Behavior)
WARN_OFF	Executes a blink-warn notification at the specified priority and then writes the value OFF to the specified slot in the priority array after a delay of Egress_Time seconds. The blink-warn notification shall not occur and the value OFF written at the specified priority immediately if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The Present_Value is OFF, or (c) Blink_Warn_Enable is FALSE. (see Clause 12.55.10.1 Blink-Warn Behavior).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Value	Description
WARN_RELINQUISH	Executes a blink-warn notification at the specified priority and then relinquishes the value at the specified priority slot after a delay of Egress_Time seconds. The blink-warn notification shall not occur, and the value at the specified priority shall be relinquished immediately if any of the following conditions occur: (a) The specified priority is not the highest active priority, or (b) The value at the specified priority is OFF or NULL, or (c) The value of the next highest non-NULL priority, including Relinquish_Default, is ON, or (d) Blink_Warn_Enable is FALSE. (See Clause 12.55.10.1 Blink-Warn Behavior).
STOP	Cancels any WARN_RELINQUISH or WARN_OFF operation in progress at the specified priority and cancels the blink-warn egress timer. The value in the priority array at the specified priority remains unchanged. If there is no warn operation currently executing at the specified priority, then this value is ignored.

The physical lighting output shall be updated whenever the Present_Value is commanded. However, when the device starts up or is reset, it is a local matter as to whether the physical lighting output is updated with the current value of Present_Value or whether the value of the physical output before startup or reset is retained.

12.55.4.1 Halting Warn Operation in Progress

The WARN_OFF and WARN_RELINQUISH operations are executed over a period of time. While a lighting operation, at a specific priority, is in progress it shall be halted when the Present_Value is written at a higher priority than the operation in progress.

When a WARN_RELINQUISH operation currently in progress is halted, the blink-warn egress timer is immediately expired and the corresponding value of the priority array is relinquished.

When a WARN_OFF operation currently in progress is halted, the egress timer is immediately expired and the value OFF is written to the priority array at the specified priority.

A WARN_OFF or WARN_RELINQUISH operation which is in progress is implicitly halted when it is overwritten by a write to the Present_Value at the same priority.

12.55.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.55.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Binary Lighting Output object. Two of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).

FAULT — Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OVERRIDDEN	Logical TRUE (1) if the output has been overridden by some mechanism local to the BACnet device, otherwise logical FALSE (0). In this context "overridden" is taken to mean that the physical output is no longer tracking changes to the Present_Value property, and the Reliability property is no longer a reflection of the physical output.
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.55.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.55.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical output in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

12.55.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical point that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the physical output;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical output;
(c) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties as if those changes had occurred to the physical output;
(e) the Present_Value property shall still be controlled by the BACnet command prioritization mechanism (see Clause 19).

12.55.10 Blink_Warn_Enable

This property, of type BOOLEAN, specifies whether a blink-warn is executed (TRUE) or not (FALSE) when a WARN, WARN_RELINQUISH, or WARN_OFF value is written to Present_Value. When this property is FALSE and a warn operation is written, a blink-warn notification shall not occur, and the effect of the operation shall occur immediately without an egress delay.

12.55.10.1 Blink-Warn Behavior

The WARN, WARN_RELINQUISH, and WARN_OFF special values to the Present_Value property, cause a blink-warn notification to occur at the specified priority. A blink-warn notification is used to warn the occupants that the lights are about to turn off, giving the occupants the opportunity to exit the space or to override the lights for a period of time.

The actual blink-warn notification mechanism shall be a local matter. The physical lights may blink once, multiple times, or repeatedly. They may also go bright, go dim, or signal a notification through some other means. In some circumstances, no blink-warn notification will occur at all. The blink-warn notification shall not be reflected in the priority array.

The WARN_RELINQUISH and WARN_OFF operations include an egress time in which the lights are held ON until the egress time expires or the operation is halted. The number of seconds for egress is specified in the Egress_Time property. The egress timer shall start when the WARN_RELINQUISH or WARN_OFF operation is written. While the egress timer is active, the property Egress_Active shall be set to TRUE. When the egress timer expires or the operation is halted, then Egress_Active shall be set to FALSE. There may only be one priority slot with an active egress timer at any time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the Blink_Warn_Enable property has the value FALSE, then the blink-warn notification shall not occur, and the effect of the operation shall occur immediately without an egress delay.

The relationship between Egress_Time and the Egress_Active property is shown in Figure 12-17.

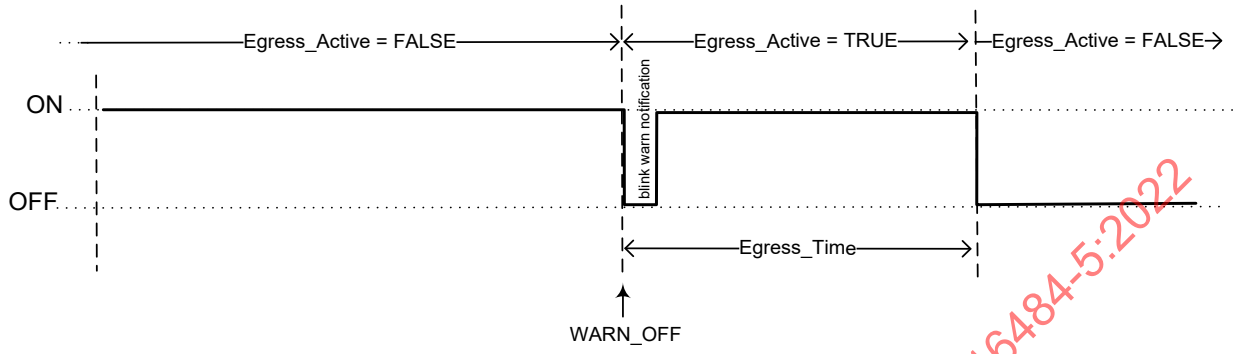


Figure 12-17. Blink-warn and Egress Time

12.55.11 Egress_Time

This property, of type Unsigned, specifies the egress time in seconds when a WARN_RELINQUISH or WARN_OFF value is written to the Present_Value property. The egress time is the time in which the light level is held at its current level before turning off.

12.55.12 Egress_Active

This property, of type BOOLEAN, shall be TRUE whenever the Egress_Time for a WARN_RELINQUISH or WARN_OFF lighting operation is in effect and FALSE otherwise.

12.55.13 Feedback_Value

This property, of type BACnetBinaryLightingPV, shall indicate the actual value of the physical lighting output and shall have the value ON (i.e. light is physically on) or OFF (i.e. light is physically off).

If this property is writable an attempt to write a value other than ON or OFF shall be denied and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

The manner by which the Feedback_Value is determined shall be a local matter.

12.55.14 Priority_Array

This read-only property, of type BACnetPriorityArray, is an array that contains prioritized commands that are in effect for this object. See Clause 19 for a description of the prioritization mechanism.

12.55.15 Relinquish_Default

This property, of type BACnetBinaryLightingPV, is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value. This property shall have the value ON or OFF.

If this property is writable an attempt to write a value other than ON or OFF shall be denied and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned. See Clause 19.

12.55.16 Power

This property, of type REAL, is the nominal power consumption of the load(s) controlled by this object when the physical lighting output is on. The units shall be kilowatts.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.55.17 Polarity

This property, of type BACnetPolarity, indicates the relationship between the physical state of the lighting output and the logical state represented by the Present_Value property. If the Polarity property is NORMAL, then the ON state of the Present_Value property is also the ON state of the physical output as long as Out_Of_Service is FALSE. If the Polarity property is REVERSE, then the ON state of the Present_Value property is the OFF state of the physical output as long as Out_Of_Service is FALSE.

If Out_Of_Service is TRUE, then the Polarity property shall have no effect on the physical output state.

12.55.18 Elapsed_Active_Time

This property, of type Unsigned32, represents the accumulated number of seconds that the Present_Value property or the Feedback_Value Property has had the value ON since the date and time indicated by the Time_Of_Active_Time_Reset property. Whether Present_Value or Feedback_Value is used as the indicator for the calculation of the Elapsed_Active_Time is a local matter.

When this property is set to zero, the Time_Of_Active_Time_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_Active_Time_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

12.55.19 Time_Of_Active_Time_Reset

This property, of type BACnetDateTime, indicates the date and time at which the accumulation of active time, indicated by the Elapsed_Active_Time property, has been started.

If the Elapsed_Active_Time property accepts writes of non-zero values, then the Time_Of_Active_Time_Reset property shall be writable, otherwise it shall be read-only.

12.55.20 Strike_Count

This property, of type Unsigned, represents the number of times that the Present_Value property or Feedback_Value property has transitioned from OFF to ON since the date and time indicated by the Time_Of_Strike_Count_Reset property. Whether Present_Value or Feedback_Value is used as the indicator for the calculation of the Strike_Count is a local matter.

When this property is set to zero, the Time_Of_Strike_Count_Reset property shall be set to the current date and time. When this property is set to a non-zero value, the value of the Time_Of_Strike_Count_Reset property shall not change. If this property is writable, it shall be a local matter whether the property accepts writes of zero only.

The Strike_Count property shall have a range of 0-65535 or greater.

12.55.21 Time_Of_Strike_Count_Reset

This property, of type BACnetDateTime, indicates the date and time at which the counting of lamp strikes, indicated by the Strike_Count property, has been started.

If the Strike_Count property accepts writes of non-zero values, then the Time_Of_Strike_Count_Reset property shall be writable, otherwise it shall be read-only.

12.55.22 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.55.23 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.55.24 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.55.25 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.55.26 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.55.27 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.55.28 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.55.29 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.55.30 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.55.31 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.55.32 Current_Command_Priority

This read-only property, of type BACnetOptionalUnsigned, indicates the currently active priority.

The value of this property shall be equal to the index of the entry in the Priority_Array from which the Present_Value's value has been taken. If Present_Value has taken on the value of Relinquish_Default, this property shall have the value Null.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.55.33 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.55.34 Value_Source_Array

This read-only property, of type BACnetARRAY[16] of BACnetValueSource, indicates the source of the last command at each priority. See Clause 19.5 for a description of the value source mechanism.

If no commands have been received at a particular priority, the associated entry in the array shall have the value 'None'.

12.55.35 Last_Command_Time

This read-only property, of type BACnetTimeStamp, indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed. See Clause 19.5 for a description of the value source mechanism.

12.55.36 Command_Time_Array

This read-only property, of type BACnetARRAY[16] of BACnetTimeStamp, indicates the time at which each priority was last commanded or relinquished. See Clause 19.5 for a description of the value source mechanism.

12.55.37 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.55.38 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.55.39 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.55.40 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.55.41 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.55.42 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.56 Network Port Object Type

The Network Port object provides access to the configuration and properties of network ports of a device. All BACnet devices shall contain at least one Network Port object per physical port which the device can be configured to communicate BACnet through (unless the port is currently for communications on a network other than the current BACnet internetwork and this use precludes its use for the current BACnet internetwork). It is a local matter whether or not Network Port objects exist for non-configured ports. It is a local matter whether or not the Network Port object is used for non-BACnet ports.

Verification and validation of property values within a Network Port object is a local matter.

Property values which are required to maintain proper operation of the network shall be retained across a device reset.

The Network Port object type can be implemented as a single interface through which all of the settings for a network port are accessed, or the Network Port objects can be organized in a hierarchy which separates the settings for each communication protocol level. See Clause 12.56.10 for more details on hierarchical Network Port objects.

Network Port objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Network Port objects that support intrinsic reporting shall apply the NONE event algorithm.

As specified in Table 12-71 and the text below, some properties of the Network Port object are required if the object is used to represent a network of a given type. For example, a Network Port object whose Network_Type is MSTP and the node is an MS/TP master node shall include the Max_Master property, and a Network Port object whose Network_Type is IPV4 shall include the IP_Subnet_Mask property. Aside from the properties so required, it is a local matter whether a Network Port object contains properties that do not apply to its Network_Type. For example, a Network Port object whose Network_Type is MSTP may include the IP_Subnet_Mask property, although the value of this property would not be used by the network. Some vendors may find it convenient to have all of their Network Port objects support the same list of properties regardless of Network_Type. This is permitted, but not required.

Table 12-71. Properties of the Network Port Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Network_Type	BACnetNetworkType	R
Protocol_Level	BACnetProtocolLevel	R
Reference_Port	Unsigned	O
Network_Number	Unsigned16	O[1,25]
Network_Number_Quality	BACnetNetworkNumberQuality	O[25]
Changes_Pending	BOOLEAN	R
Command	BACnetNetworkPortCommand	O[2]
MAC_Address	OCTET STRING	O[3]
APDU_Length	Unsigned	O[25]
Link_Speed	REAL	R
Link_Speeds	BACnetARRAY[N] of REAL	O[4]
Link_Speed_Autonegotiate	BOOLEAN	O
Network_Interface_Name	CharacterString	O
BACnet_IP_Mode	BACnetIPMode	O[5]
IP_Address	OCTET STRING	O[6]
BACnet_IP_UDP_Port	Unsigned16	O[5]
IP_Subnet_Mask	OCTET STRING	O[6]
IP_Default_Gateway	OCTET STRING	O[6]
BACnet_IP_Multicast_Address	OCTET STRING	O[7]
IP_DNS_Server	BACnetARRAY[N] of OCTET STRING	O[6]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
IP_DHCP_Enable	BOOLEAN	O[8]
IP_DHCP_Lease_Time	Unsigned	O
IP_DHCP_Lease_Time_Remaining	Unsigned	O
IP_DHCP_Server	OCTET STRING	O
BACnet_IP_NAT_Traversal	BOOLEAN	O[9]
BACnet_IP_Global_Address	BACnetHostNPort	O[10]
BBMD_Broadcast_Distribution_Table	BACnetLIST of BACnetBDTEntry	O[11]
BBMD_Accept_FD_Registrations	BOOLEAN	O[11]
BBMD_Foreign_Device_Table	BACnetLIST of BACnetFDTEntry	O[12]
FD_BBMD_Address	BACnetHostNPort	O[13]
FD_Subscription_Lifetime	Unsigned16	O[13]
BACnet_IPv6_Mode	BACnetIPMode	O[14]
IPv6_Address	OCTET STRING	O[15]
IPv6_Prefix_Length	Unsigned8	O[15]
BACnet_IPv6_UDP_Port	Unsigned16	O[14]
IPv6_Default_Gateway	OCTET STRING	O[15]
BACnet_IPv6_Multicast_Address	OCTET STRING	O[14]
IPv6_DNS_Server	BACnetARRAY[N] of OCTET STRING	O[15]
IPv6_Auto_Addressing_Enable	BOOLEAN	O[16]
IPv6_DHCP_Lease_Time	Unsigned	O
IPv6_DHCP_Lease_Time_Remaining	Unsigned	O
IPv6_DHCP_Server	OCTET STRING	O
IPv6_Zone_Index	CharacterString	O[17]
Max_Master	Unsigned8(0..127)	O[18]
Max_Info_Frames	Unsigned8	O[18]
Slave_Proxy_Enable	BOOLEAN	O[19]
Manual_Slave_Address_Binding	BACnetLIST of BACnetAddressBinding	O[19]
Auto_Slave_Discovery	BOOLEAN	O[20]
Slave_Address_Binding	BACnetLIST of BACnetAddressBinding	O[21]
Virtual_MAC_Address_Table	BACnetLIST of BACnetVMACEntry	O[22]
Routing_Table	BACnetLIST of BACnetRouterEntry	O
Event_Detection_Enable	BOOLEAN	O[23,24]
Notification_Class	Unsigned	O[23,24]
Event_Enable	BACnetEventTransitionBits	O[23,24]
Acked_Transitions	BACnetEventTransitionBits	O[23,24]
Notify_Type	BACnetNotifyType	O[23,24]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[23,24]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[24]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[24]
Event_State	BACnetEventState	O[23]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[26]
Auditable_Operations	BACnetAuditOperationFlags	O[26]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] Required to be writable in routers and any other device that requires knowledge of the network number for proper operation.
[2] Shall be present if, and only if, the object supports execution of any of the values of the Command property. If present, this property shall be writable.
[3] Required if the port is not a PTP link. Read-only if the port is a BACnet/IP port or if the network represented by this object requires VMAC addressing.
[4] Required if Link_Speed is writable.
[5] Required to be present if the port is a BACnet/IP port.
[6] Required if the port is a BACnet/IP port. If the IP_DHCP_Enable property is TRUE, and this property is configured by DHCP, this property shall be read-only.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[7] Required to be present if Network_Type is IPV4, Protocol_Level is BACNET_APPLICATION and the port supports multicast.
[8] Shall be present if, and only if, Network_Type is IPV4 and the port can be configured by DHCP.
[9] Required to be present if the Network_Type is IPV4, Protocol_Level is BACNET_APPLICATION, and the device is capable of communicating through a NAT router as described in Clause J.7.5.
[10] Required if Network_Type is IPV4, Protocol_Level is BACNET_APPLICATION, and the device is configured to communicate through a NAT router as described in Clause J.7.5
[11] Required to be present if Network_Type is IPV4 or IPV6 and the device is capable of functioning as a BBMD.
[12] Required if Network_Type is IPV4 or IPV6 and the device is capable of functioning as a BBMD.
[13] Required to be present if Network_Type is IPV4 or IPV6 and BACnet_IP_Mode or BACnet_IPv6_Mode respectively is set to FOREIGN.
[14] Required to be present if Network_Type is IPV6 and Protocol_Level is BACNET_APPLICATION.
[15] Required to be present if Network_Type is IPV6. Read-only if the value is configured by automatic address assignment.
[16] Required to be present if Network_Type is IPV6 and the port supports automatic IPv6 address assignment.
[17] Required to be present if Network_Type is IPV6 and the node supports multiple IPv6 link-local addresses.
[18] Required to be present and writable if Network_Type is MSTP, the device is an MS/TP master node, and the device supports the WriteProperty service.
[19] Required to be present and writable if Network_Type is MSTP, and the device is capable of being a Slave-Proxy device.
[20] Required if Network_Type is MSTP, Protocol_Level is BACNET_APPLICATION, and the device is capable of being a Slave-Proxy device that implements automatic discovery of slaves.
[21] Required if Network_Type is MSTP, Protocol_Level is BACNET_APPLICATION, and the device is capable of being a Slave-Proxy device.
[22] Required if the network represented by this object requires VMAC addressing.
[23] These properties are required if the object supports intrinsic reporting.
[24] These properties shall be present only if the object supports intrinsic reporting.
[25] Required to be present if Protocol_Level is BACNET_APPLICATION.
[26] This property shall be present only if the device supports audit reporting.

For convenience, the following table further illustrates the properties that are required based on the value of the Network_Type property and the various capabilities of the network types. This table is not meant to be an exhaustive list and is shown for informative purposes.

Table 12-72. Required Properties of the Network Port Object Type Based on Network_Type when Protocol_Level is BACNET_APPLICATION.

Network_Type	Additional Required Properties
ETHERNET	MAC_Address
IPV4 (BACnet_IP_Mode is NORMAL)	MAC_Address BACnet_IP_Mode IP_Address BACnet_IP_UDP_Port IP_Subnet_Mask IP_Default_Gateway IP_DNS_Server
IPV4 (BACnet_IP_Mode is FOREIGN)	MAC_Address BACnet_IP_Mode IP_Address BACnet_IP_UDP_Port IP_Subnet_Mask IP_Default_Gateway IP_DNS_Server FD_BBMD_Address FD_Subscription_Lifetime

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Network_Type	Additional Required Properties
IPV4 (BACnet_IP_Mode is BBMD)	MAC_Address BACnet_IP_Mode IP_Address BACnet_IP_UDP_Port IP_Subnet_Mask IP_Default_Gateway IP_DNS_Server BBMD_Broadcast_Distribution_Table BBMD_Accept_FD_Registrations BBMD_Foreign_Device_Table
IPV6 (BACnet_IPv6_Mode is NORMAL)	MAC_Address BACnet_IPv6_Mode IPv6_Prefix_Length IPv6_Address BACnet_IPv6_UDP_Port BACnet_IPv6_Multicast_Address IPv6_Default_Gateway IPv6_DNS_Server
IPV6 (BACnet_IPv6_Mode is FOREIGN)	MAC_Address BACnet_IPv6_Mode IPv6_Prefix_Length IPv6_Address BACnet_IPv6_UDP_Port BACnet_IPv6_Multicast_Address IPv6_Default_Gateway IPv6_DNS_Server FD_BBMD_Address FD_Subscription_Lifetime
IPV6 (BACnet_IPv6_Mode is BBMD)	MAC_Address BACnet_IPv6_Mode IPv6_Prefix_Length IPv6_Address BACnet_IPv6_UDP_Port BACnet_IPv6_Multicast_Address IPv6_Default_Gateway IPv6_DNS_Server BBMD_Broadcast_Distribution_Table BBMD_Accept_FD_Registrations BBMD_Foreign_Device_Table
MSTP (Slave node)	MAC_Address
MSTP (Master node)	MAC_Address Max_Master Max_Info_Frames
MSTP (capable of Slave Proxy)	MAC_Address Max_Master Max_Info_Frames Slave_Proxy_Enable Manual_Slave_Address_Binding Auto_Slave_Discovery Slave_Address_Binding

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 12-73. Expected Properties of the Network Port Object Type by Network_Type and Protocol_Level.

Network_Type	Protocol_Level	Properties	Conformance
ARCNET ETHERNET LONTALK VIRTUAL ZIGBEE <proprietary values>	PHYSICAL	MAC_Address Link_Speed Link_Speeds Link_Speed_Autonegotiate Network_Interface_Name	R R O O O
SERIAL	PHYSICAL	Link_Speed Link_Speeds Link_Speed_Autonegotiate Network_Interface_Name	R O O O
IPV4	PROTOCOL	IP_Address IP_Subnet_Mask IP_Default_Gateway IP_DNS_Server IP_DHCP_Enable IP_DHCP_Lease_Time IP_DHCP_Lease_Time_Remaining IP_DHCP_Server	R R R R O O O O
IPV6	PROTOCOL	IPv6_Address IPv6_Prefix_Length IPv6_Default_Gateway IPv6_DNS_Server IPv6_Auto_Addressing_Enable IPv6_DHCP_Lease_Time IPv6_DHCP_Lease_Time_Remaining IPv6_DHCP_Server IPv6_Zone_Index	R R R R O O O O O
MSTP (Slave node)	PROTOCOL	MAC_Address	R
MSTP (Master node)	PROTOCOL	MAC_Address Max_Master Max_Info_Frames	R R R
PTP	PROTOCOL		
<proprietary values>	PROTOCOL		
<proprietary values>	NON_BACNET_APPLICATION		
any (except SERIAL)	BACNET_APPLICATION	all properties	

Table 12-74. Properties of the Network Port Object Type Only Used when Protocol_Level is BACNET_APPLICATION.

Network_Type	Properties
ETHERNET ARCNET LONTALK PTP VIRTUAL <proprietary values>	
ZIGBEE	Virtual_MAC_Address_Table
MSTP	Slave_Proxy_Enable Manual_Slave_Address_Binding Auto_Slave_Discovery Slave_Address_Binding

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Network_Type	Properties
IPV4	BACnet_IP_Mode BACnet_IP_UDP_Port BACnet_IP_Multicast_Address BACnet_IP_NAT_Traversal BACnet_IP_Global_Address BBMD_Broadcast_Distribution_Table BBMD_Accept_FD_Registrations BBMD_Foreign_Device_Table FD_BBMD_Address FD_Subscription_Lifetime
IPV6	BACnet_IPv6_Mode BACnet_IPv6_UDP_Port BACnet_IPv6_Multicast_Address BBMD_Broadcast_Distribution_Table BBMD_Accept_FD_Registrations BBMD_Foreign_Device_Table FD_BBMD_Address FD_Subscription_Lifetime
all	Network_Number Network_Number_Quality APDU_Length Routing_Table

12.56.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it. When the Protocol_Level property has a value of BACNET_APPLICATION, the instance number (see Clause 20.2.14) shall correspond to the Port ID of the associated network as described in Clause 6.

It is suggested that instances greater than 255 be used for Network Port objects with a Protocol_Level of PHYSICAL or PROTOCOL.

12.56.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.56.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be NETWORK_PORT.

12.56.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.56.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Network. The four flags are:

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM — Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.56.6 Reliability

This property, of type BACnetReliability, provides an indication of whether the Network Port object, the network port, and the network connected to the port are "reliable" as far as the BACnet device can determine and, if not, why.

Reliability of a Network Port object affects all Network Port objects that reference it directly or indirectly. If this Network Port object is otherwise not in fault, but its referenced Network Port object is, then the Reliability property shall have the value REFERENCED_OBJECT_FAULT.

12.56.7 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the network port is out of service.

When Out_Of_Service is TRUE:

(a) all BACnet communication through the port shall be disabled;
(b) writing any value other than RESTART_PORT, DISCONNECT or DISCARD_CHANGES to the Command property shall result in an error response with an 'Error Class' of PROPERTY and 'Error Code' of VALUE_OUT_OF_RANGE;
(c) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the port;
(d) the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(e) other functions that depend on the state of the Reliability property shall respond to changes made to the property, as if those changes had occurred to the port.

12.56.8 Network_Type

This property, of type BACnetNetworkType, represents the type of network this Network Port object is representing.

This property shall have one of the following values:

ARCNET		
IPV4		
IPV6		
ETHERNET		
LONTALK		
MSTP		MS/TP, as defined in Clause 9.
PTP		Point-To-Point, as defined in Clause 10.
SERIAL		A physical serial port.
ZIGBEE		ZigBee as defined in Annex O
VIRTUAL		Indicates that this port represents the configuration and properties of a virtual network as described in Clause H.2.
<Proprietary Values>	Enum	A vendor may use other proprietary enumeration values to indicate that this port represents the use of message structures, procedures, and medium access control techniques other than those contained in this standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

When the Protocol_Level is BACNET_APPLICATION, the Network_Type indicates the protocol over which BACnet is operating and implies that the requirements laid out in the appropriate clause are being met. For example, if the Network_Type is IPV4, then the port is operating as a BACnet/IP port as defined in Annex J.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.56.9 Protocol_Level

This property, of type BACnetProtocolLevel, indicates whether the object represents a physical network interface (PHYSICAL), a non-BACnet protocol (PROTOCOL), the BACnet use of the protocol (BACNET_APPLICATION), or a non-BACnet use of the protocol (NON_BACNET_APPLICATION).

12.56.10 Reference_Port

This property, of type Unsigned, shall specify the instance of the Network Port object that this Network Port object uses as its lower protocol layer (i.e. transport, routing, data link, etc). This property allows the Network Port objects in the device to describe the hierarchy of protocols and physical ports in order to support complex network configuration required by some advanced BACnet products.

If this property is absent and the Protocol_Level is BACNET_APPLICATION, then it represents all protocol layers in a single object.

If this property has a value of 4194303, then this object has not been assigned a lower protocol layer. If the object is capable of representing all protocol layers in a single object, then this is a valid configuration and the object shall behave as if this property were absent. If the object is not capable of representing all protocol layers in a single object, then this is an indication that the object is not yet configured.

Object_Identifier	Network Port, 4
Object_Name	BACnet/MSTP on USB1::COM1
Reference_Port	4194303
Protocol_Level	BACNET_APPLICATION
Network_Type	MSTP
Link_Speed	76800
Link_Speeds	9600,38400,76800
Link_Speed_Autonegotiate	FALSE
Network_Interface_Name	USB1::COM1
MAC_Address	1
Max_Master	12
Max_Info_Frames	3
Slave_Proxy_Enable	FALSE
Manual_Slave_Address_Binding	...
Auto_Slave_Discovery	FALSE
Slave_Address_Binding	...
Network_Number	40
Network_Number_Quality	CONFIGURED
APDU_Length	480
Routing_Table	...

Figure 12-18. Example Network Port With No Hierarchy Chain

A Network Port object is misconfigured if the referenced Network Port object has a Protocol_Level of BACNET_APPLICATION, or the referenced Network Port object does not exist.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.10.1 Network Port Hierarchies

Support for Network Port object hierarchies is optional.

In the normal case, a single hierarchy chain consists of a Network Port object with a Protocol_Level of PHYSICAL at the bottom; one or more Network Port objects with their Protocol_Level set to PROTOCOL, and a Network Port object with

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

a Protocol_Level of BACNET_APPLICATION at the top. Multiple Network Port objects can reference a PROTOCOL or PHYSICAL Network Port object.

A Network Port object with a Protocol_Level of BACNET_APPLICATION or PHYSICAL shall not be in the middle of a hierarchy chain.

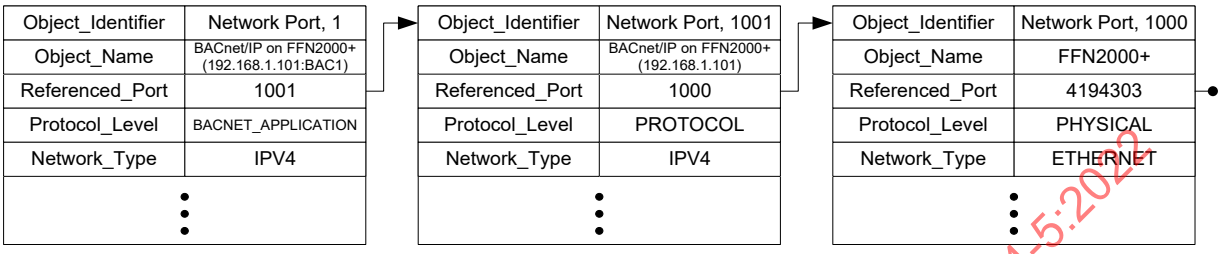


Figure 12-19. Example Network Port Hierarchy Chain

12.56.10.1.1 Property Inheritance

In a hierarchy chain of Network Port objects, a Network Port object with a Protocol_Level of BACNET_APPLICATION shall inherit property values related to configuration of the protocol or physical port from all Network Port objects in the chain.

Where a property is specified in multiple Network Port objects in the hierarchy chain, the property's value in the Network Port object nearest to the top of the chain shall be the value reflected in the topmost Network Port object. For example, in a Network Port object with a Protocol_Level of BACNET_APPLICATION, the Network_Type property shall be the same as the Network_Type of the directly referenced Network Port object.

Network Port objects with a Protocol_Level of NON_BACNET_APPLICATION are allowed to inherit property values. Network Port objects with a Protocol_Level other than BACNET_APPLICATION or NON_BACNET_APPLICATION shall not inherit property values.

Property inheritance allows clients to read and write current network settings by accessing the topmost Network Port object. It is required that inherited properties which are writable in the source (lower) Network Port object be writable in the inheriting Network Port object. When inherited properties are written, in either the source or the inheriting object, the values shall be written through to the other Network Port object. It is acceptable to make inherited properties in the source object read-only and the corresponding properties in the inheriting object be writable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

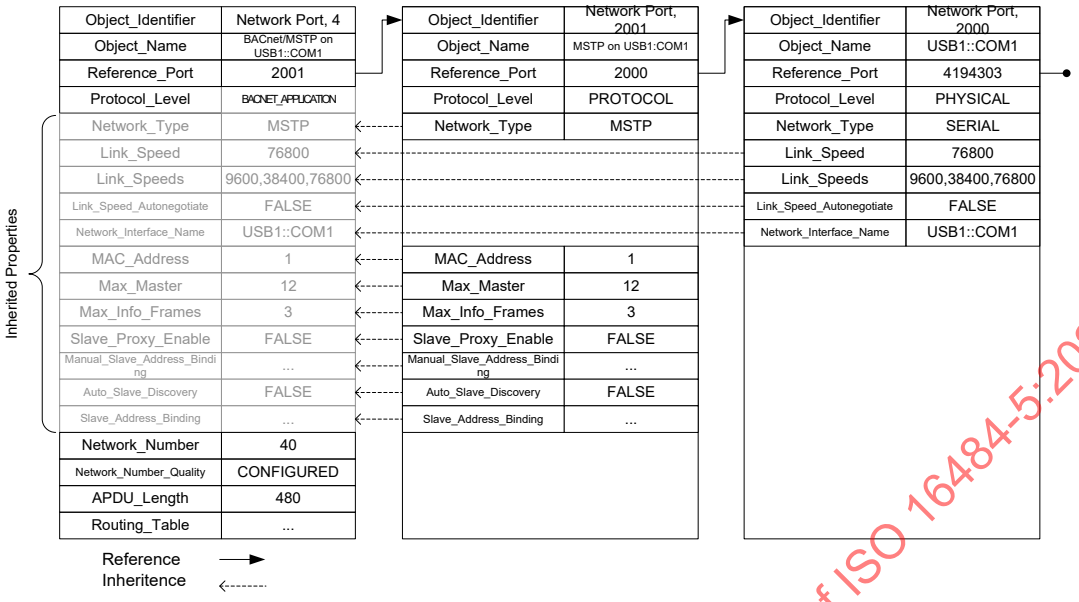


Figure 12-20. Example Network Port Hierarchy Chain Showing Property Inheritance

12.56.10.1.2 Pending Changes

In a hierarchy of Network Port objects, changes to a Network Port object may result in pending changes to multiple Network Port objects due to the sharing of inherited property values.

When an inherited property is written, and writes to the property are controlled via the pending changes functionality, it is a local matter whether or not the new property value is reflected in all affected objects immediately, or only once the changes have been activated. If the property value shows up immediately in all affected objects, then the Changes_Pending property in each of those objects shall be set to TRUE.

12.56.11 Network_Number

This property, of type Unsigned16, represents the BACnet network number associated with this network.

The range for this property shall be 0...65534. A Network_Number of 0 indicates that the Network_Number is not known or cannot be determined. Writing 0 to the Network_Number property shall force the value of the Network_Number_Quality property to UNKNOWN and allows the device to attempt to learn the network number, if possible. Writing a value other than 0 shall force the Network_Number_Quality property to CONFIGURED.

If the Network_Type is PTP, then this property shall be read-only and contain a value of 0.

This property shall be writable in routers and any other device that requires knowledge of the network number for proper operation. Routers are permitted to refuse a value of 0. In that case, the write request shall result in an error response with 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.12 Network_Number_Quality

This read-only property, of type BACnetNetworkNumberQuality, represents the current quality of the Network_Number property. If the Network_Type is PTP, the Network_Number_Quality shall be CONFIGURED.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

This property shall have one of the following values:

UNKNOWN	None of the below meanings.
LEARNED	The Network_Number was learned via receipt of a Network-Number-Is message with a flag value of 0 (learned).
LEARNED_CONFIGURED	The Network_Number was learned via receipt of a Network-Number-Is message with a flag value of 1 (configured).
CONFIGURED	The Network_Number is configured for this port.

12.56.13 Changes_Pending

This property, of type BOOLEAN, indicates whether the configuration settings in the Network Port object map to the current configuration settings. A value of FALSE indicates that the configuration settings reflect the current port configuration information. A value of TRUE indicates the configuration settings have been modified but have not been activated on the port.

When a value is written to a property that requires activation, the value of the Changes_Pending property shall automatically be set to TRUE, indicating that the current property values are not the values actively in use.

It is necessary for the client to initiate a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART in order to activate the currently visible configuration settings. This interaction is necessary to support atomic updating of multiple properties when modifying a network port configuration.

It is a local matter as to whether or not resetting the device by means other than a ReinitializeDevice service with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART discards pending changes, activates pending changes, or leaves the changes pending.

It is a local matter whether, or not, a device refuses requests to write to a Network Port object if:

(a) any Network Port object has pending changes,
(b) the write request is from a device other than that which wrote the existing pending changes, and
(c) the write would result in pending changes in any Network Port object.

When refusing such a request, the device shall return a Result(-) with an 'Error Class' of DEVICE and an 'Error Code' of CONFIGURATION_IN_PROGRESS.

12.56.14 Command

This property, of type BACnetNetworkPortCommand, is used to request that the Network Port object perform various actions.

When this property is written, the sequence of operations shall be as follows:

(1) Perform any necessary validation. If Result(-) is returned, this property shall retain the value that it had before the write was attempted and no change shall be made to any other property of the object.
(2) If validation succeeds, this property shall be set to the value written and a Result(+) be returned.
(3) The device shall begin performing the requested command actions.
(4) When the object is able to accept another command, the Command property shall be set to IDLE. This may occur immediately, when the actions have completed, or when the actions have proceeded to a point that allows the implementation to accept another command. The exact timing is a local matter.

When this property has a value other than IDLE, any attempt to write to it shall result in the return of a Result(-) with an 'Error Class' of OBJECT and an 'Error Code' of BUSY.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Writing a value of IDLE to this property shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the value of the Changes_Pending property is TRUE, writing a value other than DISCARD_CHANGES (if supported) to the Command property shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of INVALID_VALUE_IN_THIS_STATE.

Any of the following commands may be written to this property:

DISCARD_CHANGES

If the device supports this command, this object shall revert to the set of property values that were contained in this object when Changes_Pending was last equal to FALSE. Changes_Pending shall be set to FALSE, and Command shall be set to IDLE.

If the device does not support this command, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED. Changes_Pending shall remain TRUE, and Command shall be set to IDLE.

RENEW_FD_REGISTRATION

This port shall attempt to renew its foreign device registration with the BBMD indicated in FD_BBMD_Address.

If the value of Network_Type is not IPV4 and not IPV6, or if the IP mode is not FOREIGN, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the attempt to renew the foreign device registration fails, then the value of the Reliability property shall be set to RENEW_FD_REGISTRATION_FAILURE.

It is a local matter whether the value of this property remains at RENEW_FD_REGISTRATION until the registration has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the registration process has been initiated and the object is prepared to accept another command.

RESTART_SLAVE_DISCOVERY

The port shall restart the slave detection algorithm as described in Clauses 12.56.53 through 12.56.56, and 16.10.2.

If the value of Network_Type is not MSTP, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE. If the value of Network_Type is MSTP but the device does not support MS/TP Slave Proxy functionality, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

The value of the Command property shall return to IDLE as soon as discovery has been initiated. The discovery process will typically require a significant amount of additional time.

RENEW_DHCP

If DHCP is supported, then this port shall attempt to renew the DHCP lease for this port.

If the port cannot be made to renew the DHCP lease or otherwise determine the address automatically in IPv6, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the value of Network_Type is IPV4 or IPV6 and the IP_DHCP_Enable or the IPv6_Auto_Addressing_Enable property respectively is not present, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If the value of Network_Type is not IPV4 or IPV6, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the value of Network_Type is IPV4 and the value of IP_DHCP_Enable is FALSE, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If the attempt to renew the DHCP address fails, then the value of the Reliability property shall be set to RENEW_DHCP_FAILURE.

It is a local matter whether the value of this property remains at RENEW_DHCP until the renewal has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the renewal process has been initiated and the object is prepared to accept another command.

RESTART_AUTONEGOTIATION — This port shall restart its link speed auto-negotiation algorithm.

If the value of the Link_Speed_Autonegotiate property is FALSE, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If this port does not support auto-negotiation when the command is issued, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If the auto-negotiation algorithm fails, the value of the Reliability property shall be set to RESTART_AUTONEGOTIATION_FAILURE.

It is a local matter whether the value of this property remains at RESTART_AUTONEGOTIATION until the auto-negotiation has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the auto-negotiation process has been initiated and the object is prepared to accept another command.

DISCONNECT — This port shall terminate the network connection.

If the value of Network_Type is not PTP, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

It is a local matter whether the value of this property remains at DISCONNECT until the disconnection has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the disconnection process has been initiated and the object is prepared to accept another command.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

RESTART_PORT	This port shall attempt to restart and reconnect to the network as if the device were reinitialized. All data that was learned, cached, or otherwise automatically determined for the port's operation shall be cleared. All initialization, negotiation, and registration functions the port is expected to perform upon device initialization shall be performed again. If the restart fails, the value of the Reliability property shall be set to RESTART_FAILURE. It is a local matter whether the value of this property remains at RESTART_PORT until the restart has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the restart process has been initiated and the object is prepared to accept another command. If the device cannot perform the restart of the port without a reinitialization of the entire device, writing this value shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow command values other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard. A proprietary command failure shall result in the value of the Reliability property being set to PROPRIETARY_COMMAND_FAILURE and the value of this property being set to IDLE. It is a local matter whether the value of this property remains at the proprietary value until the proprietary action has completed (whether in success or failure), and then returns to IDLE; or whether the property returns to IDLE once the action has been initiated and the object is prepared to accept another command.

This enumerated value is extensible. Writing an unknown value to this property shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.56.15 MAC_Address

This property, of type OCTET STRING, contains the BACnet MAC address used on this network. The value of this property shall be conveyed with the most significant octet first. If Network_Type is IPV4 and the Protocol_Level is BACNET_APPLICATION, then the value of this property shall contain the six octet combination of the IP_Address and BACnet_IP_UDP_Port and shall be read-only. If the value of Network_Type is a value that represents a port that requires VMAC addressing, then the value of this property shall be read-only and contain the VMAC address.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. The value of this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.16 APDU_Length

This property, of type Unsigned, is the maximum number of octets that may be contained in a single indivisible application protocol data unit sent or received on this port. The value of this property shall be greater than or equal to 50. This property also indicates the maximum number of octets that may be contained in a single individual network service data unit sent or received on this port.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.56.17 Link_Speed

This property, of type REAL, represents the network communication rate as the number of bits per second. A value of 0 indicates an unknown communication rate.

If the value of the Link_Speed_Autonegotiate property is TRUE, then this property shall be read-only.

If this property is writable, writing a value to this property that is not present in the Link_Speeds property shall result in the return of a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.18 Link_Speeds

This read-only property, of type BACnetARRAY of REAL, is an array of the link speeds supported by this network port.

12.56.19 Link_Speed_Autonegotiate

This property, of type BOOLEAN, represents the auto negotiation setting of the network port.

A value of TRUE indicates that the device automatically determines the speed of this network port. In this case, Link_Speed shall be read-only and indicate the determined speed, if available. A value of FALSE indicates that the link speed is determined by the value of the Link_Speed property.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.20 Network_Interface_Name

This property, of type CharacterString, is used to identify the network interface hardware to which this network port is bound. For example, if Network_Type is IPV4, the value of this property identifies the Ethernet hardware interface that this network port is using to communicate.

This property shall be read-only if it is inherited from another Network Port object.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.21 BACnet_IP_Mode

This property, of type BACnetIPMode, indicates the BACnet/IP mode of this network port.

This property shall have one of the following values:

NORMAL The device is operating as neither a foreign device nor a BBMD over this network port.

FOREIGN The device is operating as a foreign device over this network port.

BBMD The device is operating as a BBMD over this network port.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.22 IP_Address

This property, of type OCTET_STRING, indicates the IP address of this network port. This property shall be conveyed most significant octet first. If the IP_DHCP_Enable property is TRUE, this property shall be read-only.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.23 BACnet_IP_UDP_Port

This property, of type Unsigned16, indicates the UDP port number of this network port.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.24 IP_Subnet_Mask

This property, of type OCTET STRING, indicates the subnet mask for this network. This property shall be conveyed with the most significant octet first. If the IP_DHCP_Enable property is TRUE, this property shall be read-only.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.25 IP_Default_Gateway

This property, of type OCTET STRING, indicates the IP address of the default gateway for this network. This property shall be conveyed with the most significant octet first. If the IP_DHCP_Enable property is TRUE, this property shall be read-only.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.26 BACnet_IP_Multicast_Address

This property, of type OCTET STRING, contains the BACnet/IP multicast group address to be used for the distribution of BACnet broadcast messages. See Clause J.8. The value of this property shall be conveyed with the most significant octet first.

A value of X'00000000' indicates that BACnet/IP multicast is not used.

If present, this property shall be writable. Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.27 IP_DNS_Server

This property, of type BACnetARRAY of OCTET STRING containing at least one entry, indicates the DNS server used by this network port for Internet host name resolution. The values of this property shall be conveyed with the most significant octet first.

A value of X'00000000' in an array entry indicates that the DNS server address is not available or is not configured.

If the IP_DHCP_Enable property is TRUE, and this property value is configured by DHCP, then this property shall be read-only.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.28 IP_DHCP_Enable

This property, of type BOOLEAN, indicates whether or not this network is configured via DHCP. A value of TRUE indicates that DHCP configuration is enabled, FALSE indicates it is not.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

This property is required if DHCP configuration is supported by this network port.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.29 IP_DHCP_Lease_Time

This read-only property, of type Unsigned, indicates the lease time in seconds of the last DHCP lease obtained for the port. If IP_DHCP_Enable is FALSE, or no lease has been acquired, or the value is unknown, this property shall be 0.

12.56.30 IP_DHCP_Lease_Time_Remaining

This read-only property, of type Unsigned, indicates the lease time in seconds remaining of the last DHCP lease obtained for the port. If IP_DHCP_Enable is FALSE, or no lease has been acquired, or the value is unknown, this property shall be 0.

12.56.31 IP_DHCP_Server

This read-only property, of type OCTET STRING, indicates the address of the DHCP server from which the last DHCP lease was obtained for the port. If the address of the DHCP server cannot be determined, the value of this property shall be X'00000000'.

12.56.32 BACnet_IP_NAT_Traversal

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) this port is configured to operate in a NAT environment, as described in Clause J.7.5, and the global address is indicated by the value of the BACnet_IP_Global_Address property.

If present, this property shall be writable. Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.33 BACnet_IP_Global_Address

This property, of type BACnetHostNPort, indicates the global address and UDP port from which the network port can be accessed from the global side of a NAT router. How the public IP address and UDP port are determined is a local matter.

The 'none' choice in the BACnetHostAddress portion and a value of X'0000' in the port portion indicates that the global address cannot be determined or is not yet configured.

If the device does not automatically determine the global address, then this property shall be writable.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.34 BBMD_Broadcast_Distribution_Table

This property, of type BACnetLIST of BACnetBDTEntry, is required to be present and writable if BACnet_IP_Mode is BBMD.

The value of this property maps to the BDT (as specified in Annex J and Annex U) for this port as follows:

(a) If the Network_Type is IPV4, the current value of the BDT may be read at any time with the Read-Broadcast-Distribution-Table BVLL message.
(b) If this property has no pending changes, reading this property shall return the current value of the BDT.
(c) If this property has pending changes, reading this property shall return the last value written to the property, and not the current value of the BDT.
(d) If this property has pending changes, reading the BDT via Read-Broadcast-Table BVLL shall return the current value of the BDT.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(e) If a list entry contains a host name, then the corresponding entry in the Read-Broadcast-Distribution-Table-Ack BVLL message shall contain the IP address and port of the resolved host name, or X'000000000000' to indicate that the host name has not been resolved.

The 'none' choice of the BACnetHostAddress portion shall not be used for list entries.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.35 BBMD_Accept_FD_Registrations

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) this device shall accept foreign device registrations. This property is required to be present and writable if BACnet_IP_Mode is BBMD.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.36 BBMD_Foreign_Device_Table

This read-only property, of type BACnetLIST of BACnetFDTEntry, is required to be present if BBMD_Accept_FD_Registrations is TRUE.

The value of this property reflects the current value of the foreign device table. Each entry shall contain the following information:

bacnetip-address	The 6-octet B/IP address or the 18-octet B/IPv6 address of the registered foreign device.
time-to-live	The time to live for the entry, as provided at the time of registration.
remaining-time-to-live	The remaining time to live for the entry. This includes the grace period added at the time of registration.

12.56.37 FD_BBMD_Address

This property, of type BACnetHostNPort, indicates the BBMD with which the local device is to register as a foreign device when BACnet_IP_Mode is FOREIGN. This property shall be present and writable if BACnet_IP_Mode is FOREIGN.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.38 FD_Subscription_Lifetime

This property, of type Unsigned16, indicates the Time-To-Live value, in seconds, to be used in the Register-Foreign-Device BVLL message. This property shall be present and writable if BACnet_IP_Mode is FOREIGN.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.39 BACnet_IPv6_Mode

This property, of type BACnetIPMode, indicates the BACnet/IPv6 mode of this network port.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

This property shall have one of the following values:

NORMAL The device is operating as neither a foreign device nor a BBMD over this network port.

FOREIGN The device is operating as a foreign device over this network port.

BBMD The device is operating as a BBMD for this network port.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.40 IPv6_Address

This property, of type OCTET_STRING, indicates the IPv6 address of this network port. This property shall be conveyed most significant octet first. If the IPv6 address is obtained automatically, this property shall be read-only, and the value of this property shall contain the address with the highest precedence. See RFC 6724.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.41 IPv6_Prefix_Length

This property, of type Unsigned8, indicates the length in bits of the subnet prefix of the IPv6 address of this network port. The value of this property shall be in the range 1 to 128. If the IPv6 address is obtained automatically, this property shall be read-only.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.42 BACnet_IPv6_UDP_Port

This property, of type Unsigned16, indicates the BACnet/IPv6 UDP port number of this network port.
If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when a value of ACTIVATE is written to the Command property.

12.56.43 IPv6_Default_Gateway

This property, of type OCTET STRING, indicates the IPv6 address of the default gateway for this network. This property shall be conveyed with the most significant octet first. If the IPv6 address is obtained automatically, this property shall be read-only.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.44 BACnet_IPv6_Multicast_Address

This property, of type OCTET STRING, contains the IPv6 multicast address UDP port to be used for the distribution of BACnet broadcast messages in the local multicast domain. See Clause U.4. The value of this property shall be comprised of the IPv6 multicast address followed by the UDP port, both of which shall be conveyed with the most significant octet first.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.56.45 IPv6_DNS_Server

This property, of type BACnetARRAY of OCTET STRING containing at least one entry, indicates the IPv6 address of the DNS server used by this network port for Internet host name resolution. The values of this property shall be conveyed with the most significant octet first. If the DNS server addresses are obtained automatically, this property shall be read-only.

A value of X'00000000000000000000000000000000' in an array entry indicates that the DNS server address is not available or is not configured.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.46 IPv6_Auto_Addressing_Enable

This property, of type BOOLEAN, indicates whether or not this network is configured for automatic address assignment via DHCPv6, Stateless Auto Address Configuration (SLAAC, RFC 4862), or neighbor discovery. A value of TRUE indicates that automatic address assignment is enabled, FALSE indicates it is not.

This property is required if any form of IPv6 automatic address configuration is supported by this network port.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.47 IPv6_DHCP_Lease_Time

This read-only property, of type Unsigned, indicates the lease time in seconds of the last DHCPv6 lease obtained for the port. If IPv6_Auto_Addressing_Enable is FALSE, or DHCPv6 is not in use, or no lease has been acquired, or the value is unknown, this property shall be 0.

12.56.48 IPv6_DHCP_Lease_Time_Remaining

This read-only property, of type Unsigned, indicates the lease time in seconds remaining of the last DHCPv6 lease obtained for the port. If IPv6_Auto_Addressing_Enable is FALSE, or DHCPv6 is not in use, or no lease has been acquired, or the value is unknown, this property shall be 0.

12.56.49 IPv6_DHCP_Server

This read-only property, of type OCTET STRING, indicates the address of the DHCPv6 server from which the last DHCPv6 lease was obtained for the port. If the address of the DHCPv6 server cannot be determined, or DHCPv6 is not in use, the value of this property shall be X'00000000000000000000000000000000'.

12.56.50 IPv6_Zone_Index

This property, of type CharacterString, contains the zone index for the B/IPv6 link local address when the node supports multiple IPv6 link local addresses.

According to RFC 4007, because all link-local addresses in a host have a common prefix, normal routing procedures cannot be used to choose the outgoing interface when sending packets to a link-local destination. A special identifier, known as a zone index, is needed to provide the additional routing information.

If this property is writable, then a successful write to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.51 Max_Master

This property, of type Unsigned8, shall be present if the Network_Type is MSTP and the device is a master node on the MS/TP network connected to this port. If the device supports execution of the WriteProperty service, then this property

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

shall be writable and the valid range for the value of this property shall be 0 to 127. Otherwise, its value shall be 127. See Clause 9.5.3.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.52 Max_Info_Frames

This property, of type Unsigned8, shall be present if the Network_Type is MSTP and the device is a master node on the MS/TP network connected to this port. The value of Max_Info_Frames specifies the maximum number of information frames the node may send on this port before it must pass the token. If the device supports execution of the WriteProperty service, then this property shall be writable and the valid range for the value of this property shall be 1 to 255. Otherwise, its value shall be 1. See Clause 9.5.3.

Writing to this property shall set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.53 Slave_Proxy_Enable

This property, of type BOOLEAN, is an indication of whether (TRUE) or not (FALSE) the device will perform Slave-Proxy functions for this port. This property shall be present and writable if the device is capable of performing the functions of a Slave-Proxy device on this port.

12.56.54 Manual_Slave_Address_Binding

This property, of type BACnetLIST of BACnetAddressBinding, describes the manually configured set of slave devices for which this device is acting as a Slave Proxy as described in Clause 16.10.2. This property shall be present and writable if the device is capable of performing the functions of a Slave-Proxy device on this port.

This property is used to manually configure a set of slave devices connected to this port for which this device will be a proxy. This property allows a Slave-Proxy that does not support automatic slave discovery to be configured with a set of slaves for which this device will be a proxy. It also allows a Slave-Proxy device to be a proxy for Slave devices that do not support the special object instance of 4194303 as described in Clause 12. When enabled, the Slave-Proxy device shall periodically check each device that is in this list, and not in the Slave_Address_Binding list, by reading the device's Protocol_Services_Supported property from the device's Device object using the ReadProperty service. If the device responds and indicates that it does not execute the Who-Is service, it shall be added to the Slave_Address_Binding property. The period at which the devices are checked is a local matter.

12.56.55 Auto_Slave_Discovery

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the device will perform automatic slave detection functions for this port. This property shall be present if the device is capable of performing the functions of a Slave-Proxy device on this port.

Slave detection shall be accomplished by the proxy device using ReadProperty services to read, at a minimum, the Device object's Protocol_Services_Supported property for each MAC address on the network connected to this port. The ReadProperty service shall use the special object instance of 4194303 as described in Clause 15.5.2. If the device is found to support execution of the Who-Is service, it is ignored; otherwise, the device shall be added to the Slave_Address_Binding property. The slave detection algorithm shall be repeated periodically. The period at which it is repeated is a local matter.

12.56.56 Slave_Address_Binding

This property, of type BACnetLIST of BACnetAddressBinding, describes the set of slave devices for which this device is acting as a Slave-Proxy on this port as described in Clause 16.10.2. This property shall be present if the device is capable of performing the functions of a Slave-Proxy device on this port.

The set of devices described by the Slave_Address_Binding property consists of those devices described in the Manual_Slave_Address_Binding and those devices that are automatically discovered. When enabled, the Slave-Proxy

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

device shall periodically check each device in this list by reading the device's Protocol_Services_Supported property from the device's Device object using the ReadProperty service. If the device fails to respond or indicates that it executes the Who-Is service, it shall be removed from the list. The period at which the devices are checked is a local matter.

12.56.57 Virtual_MAC_Address_Table

This property, of type BACnetLIST of BACnetVMACEntry, is the list of VMAC entries as described in Clause H.7.

VMAC table entries shall contain the following information:

virtual-mac-address	The virtual MAC address used for the native MAC address indicated in the native-mac-address portion. The maximum size of the Virtual MAC address shall be 6 octets.
native-mac-address	The native MAC address used by the data link to address the node identified by the virtual MAC address.

If this property is writable, then a successful write to this property will set the Changes_Pending property to TRUE. A value written to this property shall become effective when the device receives a ReinitializeDevice service request with a 'Reinitialized State of Device' of ACTIVATE_CHANGES or WARMSTART.

12.56.58 Routing_Table

This read-only property, of type BACnetLIST of BACnetRouterEntry, contains the table of first hop routers to remote networks reachable through this port.

Router table entries shall contain the following information:

network-number	The network number reachable through the router specified by mac-address.
mac-address	The MAC address of the router on the network connected to this port that leads directly or indirectly to that network number.
status	Conveys whether the associated network is able to receive traffic. The values for this field are: AVAILABLE, BUSY, and DISCONNECTED.
performance-index	This optional field is used to convey the performance index as conveyed in an I-Could-Be-Router-To-Network network layer message. See Clause 6.4.3.

12.56.59 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.56.60 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.56.61 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.56.62 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.56.63 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.56.64 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.56.65 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.56.66 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.56.67 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.56.68 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.56.69 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.56.70 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.56.71 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.56.72 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.56.73 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.56.74 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.57 Timer Object Type

The Timer object type defines a standardized object whose properties represent the externally visible characteristics of a countdown timer.

The Timer object provides a network-visible view of selected parameters of a countdown timer. The operating state of the timer may be viewed and controlled through these properties.

The Timer object type supports a pre-configured timeout as well as providing a specific timeout on activation of the timer. A boolean indication of when the timer is counting down supports applications that monitor a boolean flag for temporary behavior. The commanding mechanism, similar to the Schedule object, supports applications that require commands to be initiated on change of the timer's state. The intrinsic event reporting supports applications that need to be notified on change of the timer's state.

The countdown timer represented by this object type maintains a state, represented by its Timer_State property, whose state-event diagram is depicted by the following figure.

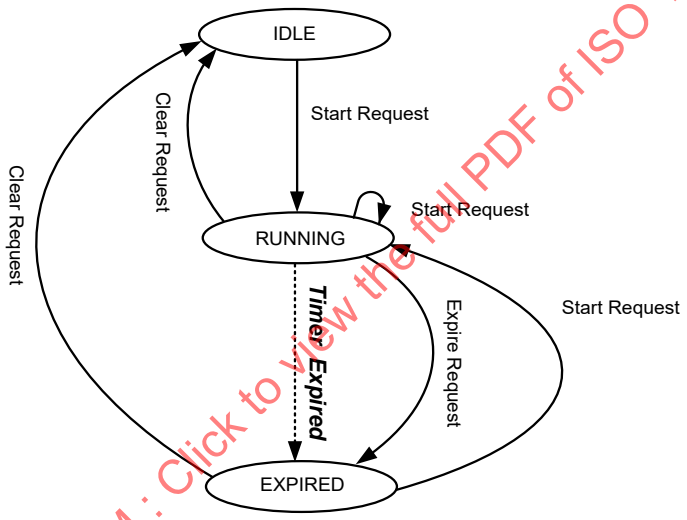


Figure 12-21. State Transitions for the Timer object.

The timer operation may be controlled by writing to the Present_Value property, the Timer_Running property, or the Timer_State property. For the description of allowed values or value ranges, see the state machine description below, and property descriptions in respective sub-clauses.

The timer's state machine shall operate as follows:

State IDLE

In the IDLE state, the timer is inactive, waiting to be activated. The timer may be activated using a default timeout or a specific timeout. The Present_Value property shall have a value of zero. The Timer_Running property shall have a value of FALSE. The Expiration_Time property shall indicate the unspecified datetime value.

Start Request with Default Timeout

If a value of TRUE is written to the Timer_Running property,

then set Initial_Timeout to the value of Default_Timeout; set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to IDLE_TO_RUNNING; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the IDLE_TO_RUNNING transition if present; and enter the RUNNING state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Start Request with Specific Timeout
If a value within the supported range is written to the Present_Value property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to IDLE_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the IDLE_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request
If a value of IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

Expire Request
If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

Out of Range Request
If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated, no state transition shall occur, and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

State RUNNING

In the RUNNING state, the timer is active and is counting down the remaining time. The Present_Value property shall indicate the remaining time until expiration. The Timer_Running property shall have a value of TRUE. The Expiration_Time property shall indicate the date and time when the timer will expire. The value of Expiration_Time shall be calculated at the time the property is read.

Timer Expired
If the remaining time indicated by Present_Value reaches zero,

then set Timer_Running to FALSE; set Timer_State to EXPIRED; set Last_State_Change to RUNNING_TO_EXPIRED; set Expiration_Time to the current date and time; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_EXPIRED transition if present; and enter the EXPIRED state.

Expire Request
If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,

then set Timer_Running to FALSE; set Timer_State to EXPIRED; set Last_State_Change to FORCED_TO_EXPIRED; set Present_Value to zero; set Expiration_Time to the current date and time; set Update_Time to the current date and time; initiate the write requests for the FORCED_TO_EXPIRED transition if present; and enter the EXPIRED state.

Start Request with Default Timeout
If a value of TRUE is written to the Timer_Running property,

then set Last_State_Change to RUNNING_TO_RUNNING; set Initial_Timeout to the value of Default_Timeout; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_RUNNING transition if present; and enter the RUNNING state.

Start Request with Specific Timeout
If a value within the supported range is written to the Present_Value property,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

then set Last_State_Change to RUNNING_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request
If a value of IDLE is written to the Timer_State property,

then set Timer_Running to FALSE; set Timer_State to IDLE; set Last_State_Change to RUNNING_TO_IDLE; set Present_Value to zero; set Expiration_Time to the unspecified datetime value; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_IDLE transition if present; and enter the IDLE state.

Out of Range Request
If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated; no state transition shall occur; and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

State EXPIRED

In the EXPIRED state, the timer has expired or was forced to be expired. The Present_Value property shall have a value of zero. The Timer_Running property shall be FALSE. The Expiration_Time property shall indicate the date and time when this state was entered.

Start Request with Default Timeout
If a value of TRUE is written to the Timer_Running property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to EXPIRED_TO_RUNNING; set Initial_Timeout to the value of Default_Timeout; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the EXPIRED_TO_RUNNING transition if present; and enter the RUNNING state.

Start Request with Specific Timeout
If a value within the supported range is written to the Present_Value property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to EXPIRED_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the EXPIRED_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request
If a value of IDLE is written to the Timer_State property,

then set Timer_State to IDLE; set Last_State_Change to EXPIRED_TO_IDLE; set Expiration_Time to the unspecified datetime value; set Update_Time to current date and time; initiate the write requests for the EXPIRED_TO_IDLE transition if present; and enter the IDLE state.

Expire Request
If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

Out of Range Request
If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated, no state transition shall occur; and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Since the timer is counting down a duration that is set at the start of the timer, changes to the Local_Date or Local_Time properties have no effect on the timer operation. The Expiration_Time property, in timer state RUNNING, indicates the date and time of expiration. While in this state, Expiration_Time shall be calculated at the time the property is accessed. In other states, Expiration_Time shall be set upon entry to that state.

If the properties State_Change_Values, List_Of_Object_Property_References and Priority_For_Writing are present and writable, then the Timer object shall be capable of writing values of type NULL, BOOLEAN, Unsigned, INTEGER, REAL, and ENUMERATED to properties in the local device referenced by List_Of_Object_Property_References. Support for writing to properties in other devices is optional.

Devices that support Timer objects that contain the Update_Time or the Expiration_Time properties shall support the Local_Date and Local_Time properties in their Device object.

Timer objects that support intrinsic reporting shall apply the CHANGE_OF_TIMER event algorithm.

The Timer object type and its standardized properties are summarized in Table 12-75 and described in detail in this clause.

Table 12-75. Properties of the Timer Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	Unsigned	R[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	O[3]
Reliability	BACnetReliability	O[1]
Out_Of_Service	BOOLEAN	O
Timer_State	BACnetTimerState	R[2]
Timer_Running	BOOLEAN	R[2]
Update_Time	BACnetDateTime	O[3]
Last_State_Change	BACnetTimerTransition	O
Expiration_Time	BACnetDateTime	O
Initial_Timeout	Unsigned	O
Default_Timeout	Unsigned	O
Min_Pres_Value	Unsigned	O[4]
Max_Pres_Value	Unsigned	O[4]
Resolution	Unsigned	O
State_Change_Values	BACnetARRAY[7] of BACnetTimerStateChangeValue	O[5]
List_Of_Object_Property_References	BACnetLIST of BACnetDeviceObjectPropertyReference	O[5]
Priority_For_Writing	Unsigned(1..16)	O[5]
Event_Detection_Enable	BOOLEAN	O[3,6]
Notification_Class	Unsigned	O[3,6]
Time_Delay	Unsigned	O[3,6]
Time_Delay_Normal	Unsigned	O[3,6]
Alarm_Values	BACnetLIST of BACnetTimerState	O[3,6]
Event_Enable	BACnetEventTransitionBits	O[3,6]
Acked_Transitions	BACnetEventTransitionBits	O[3,6]
Notify_Type	BACnetNotifyType	O[3,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,7]
Reliability_Evaluation_Inhibit	BOOLEAN	O[8]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[9]
Auditable_Operations	BACnetAuditOperationFlags	O[9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required to be writable when Out_Of_Service is TRUE.
[2] If Present_Value is writable when Out_Of_Service is FALSE, then these properties shall be writable.
[3] These properties are required if the object supports intrinsic reporting.
[4] If Present_Value is writable when Out_Of_Service is FALSE, then these properties are required to be present.
[5] If one of these properties is present, then all of these properties shall be present.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[8] If this property is present, then the Reliability property shall be present.
[9] This property shall be present only if the device supports audit reporting.

12.57.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.57.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.57.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be TIMER.

12.57.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.57.5 Present_Value

This property, of type Unsigned, indicates, in the RUNNING state, the remaining time, in milliseconds, until the timer expires and the EXPIRED state is entered. This property shall indicate a value of zero if the timer is in the IDLE or EXPIRED state.

Writing a value to this property that is within the supported range, defined by Min_Pres_Value and Max_Pres_Value, shall force the timer to transition to the RUNNING state. The value written shall be used as the initial timeout and set into the Initial_Timeout property.

Writing a value of zero to this property while the timer is in the RUNNING state shall be considered an expire request and force the timer state to transition to state EXPIRED. If a value of zero is written to the property while the timer is in the EXPIRED or IDLE state, then no transition of the timer state shall occur.

Writing a value to this property that is outside the supported range and not zero shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.57.6 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.57.8 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the timer is "reliable" as far as the BACnet device or operator can determine and, if not, why. This property shall be writable when Out_Of_Service is TRUE.

12.57.9 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the timer this object represents is in service and will count down.

When Out_Of_Service is FALSE, the timer is functioning as specified.

When Out_Of_Service is TRUE:

(a) the object shall behave as specified, except that Present_Value shall not automatically count down in the RUNNING state;
(b) the Present_Value property and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating states and transitions, or for testing purposes;
(c) other functions that depend on the state of the Present_Value and Reliability properties, such as timer state changes, shall respond to changes made to that, as if those changes had occurred while Out_Of_Service was FALSE.

12.57.10 Timer_State

This property, of type BACnetTimerState, indicates the current state of the timer this object represents. To clear the timer, i.e. to request the timer to enter the IDLE state, a value of IDLE is written to this property.

The values that may be taken on by this property are:

IDLE	The timer is not running and not expired. Writing this value to this property while in the RUNNING or EXPIRED state will force the timer to enter the IDLE state. If already in the IDLE state, no state transition occurs if this value is written.
RUNNING	The timer is counting down, and will expire when Present_Value reaches zero or the timer is forced to expire.
EXPIRED	The timer has expired.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

Writing a value other than IDLE to this property shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.57.11 Timer_Running

This property, of type BOOLEAN, shall have a value of TRUE if the current state of the timer is RUNNING, otherwise FALSE. This property may be used by other objects that require a simple BOOLEAN flag for determining if the timer is in RUNNING state.

Writing a value of TRUE to this property, in any timer state, shall be considered a start request. Present_Value shall be set to the value specified in the Default_Timeout property.

Writing a value of FALSE to this property while the timer is in the RUNNING state shall be considered an expire request and shall force the timer to transition to state EXPIRED. When writing a value of FALSE to this property while the timer is in the EXPIRED or IDLE state, no transition of the timer state shall occur.

12.57.12 Update_Time

This read-only property, of type BACnetDateTime, indicates the date and time of the last transition of the timer state.

If a transition of the timer state has never occurred, then this property shall take on the unspecified datetime value.

If the object supports event reporting, then this property shall be the pUpdateTime parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.13 Last_State_Change

This read-only property, of type BACnetTimerTransition, shall indicate the last transition the timer state machine performed. The values that may be taken on by this property are:

NONE	No timer state transition has occurred since initialization of the object.
IDLE_TO_RUNNING	The last state transition performed by the timer was from IDLE to RUNNING, using either the default or a specific initial timeout.
RUNNING_TO_IDLE	The last state transition performed by the timer was from RUNNING to IDLE. A write of value IDLE to Timer_State occurred.
RUNNING_TO_RUNNING	The last state transition performed by the timer was from RUNNING to RUNNING, using either the default or a specific initial timeout.
RUNNING_TO_EXPIRED	The last state transition performed by the timer was from RUNNING to EXPIRED. The Present_Value reached zero while counting down.
FORCED_TO_EXPIRED	The last state transition performed by the timer was from RUNNING to EXPIRED. A value of zero was written to Present_Value, or a value of FALSE was written to Timer_Running while in the RUNNING state, before Present_Value reached zero.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

EXPIRED_TO_IDLE	The last state transition performed by the timer was from EXPIRED to IDLE. The timer expired and a clear request was performed.
EXPIRED_TO_RUNNING	The last state transition performed by the timer was from EXPIRED to RUNNING, using either the default or a specific initial timeout.

If the object supports event reporting, then this property shall be the pLastStateChange parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.14 Expiration_Time

This read-only property, of type BACnetDateTime, indicates the date and time when the timer will expire or has expired.

The value indicated by this property depends on the current timer state, as follows:

In timer state IDLE	This property shall contain the unspecified datetime value.
In timer state RUNNING	The value indicated by this property is the date and time when the timer will expire and will perform a transition to EXPIRED timer state. The value shall be calculated at the time this property is accessed.
In timer state EXPIRED	This property indicates the time at which the object transitioned to the EXPIRED state.

If the object supports event reporting, then this property shall be the pExpirationTime parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.15 Initial_Timeout

This read-only property, of type Unsigned, indicates the initial timeout, in milliseconds, that was taken as initial duration to count down when the timer last transitioned to state RUNNING.

If the object supports event reporting, then this property shall be the pInitialTimeout parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.16 Default_Timeout

This property, of type Unsigned, specifies the default initial timeout, in milliseconds. This default timeout is used as the initial timeout when a value of TRUE is written to the Timer_Running property.

The value of this property shall be within the supported range specified by the values of the Min_Pres_Value and Max_Pres_Value properties. If this property is writable, writing a value to this property that is outside the supported range shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.57.17 Min_Pres_Value

This property, of type Unsigned, indicates the minimum initial timeout the timer supports, in milliseconds.

12.57.18 Max_Pres_Value

This property, of type Unsigned, indicates the maximum initial timeout the timer supports, in milliseconds.

12.57.19 Resolution

This property, of type Unsigned, indicates the resolution of the timer, in milliseconds. The time of expiration of the timer may vary plus or minus this resolution from the remaining time indicated by the Present_Value property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.57.20 State_Change_Values

This property, of type BACnetARRAY[7] of BACnetTimerStateChangeValue, represents the values that are to be written to the referenced properties when a change of the timer state occurs. Each of the elements 1-7 of this array may contain a value to be written for the respective change of timer state. The array index of the element is equal to the numerical value of the BACnetTimerTransition enumeration for the respective timer state change. The timer state change NONE has no corresponding array element.

Each element of this array is of type BACnetTimerStateChangeValue, which is a choice of the following options.

no-value	This option, of type NULL, is used to indicate that no value shall be written when the respective change of timer state occurs.
null, boolean, unsigned, integer, real, double, octetstring, characterstring, bitstring, enumerated, date, time, objectidentifier	These options specify a value of the respective primitive datatype. The option 'null' may be used in combination with other datatypes for specifying a relinquish command for a commandable property.
constructed-value	This option specifies a value of complex datatype. This option shall not be selected if the respective data type is supported by a defined option.
datetime	This option specifies a value of datatype BACnetDateTime.
lighting-command	This option specifies a value of datatype BACnetLightingCommand.

All non-NULL values used in this property shall be of the same datatype, and all properties referenced by the List_Of_Object_Property_References shall be writable with that datatype. If these conditions are not met, then the Reliability property shall have the value CONFIGURATION_ERROR.

If this property is written with a state change value containing a datatype not supported by this instance of the Timer object (e.g., the List_Of_Object_Property_References property cannot be configured to reference a property of the unsupported datatype), the device shall return a Result(-) response, specifying an 'Error Class' of PROPERTY and an 'Error Code' of DATATYPE_NOT_SUPPORTED.

12.57.21 List_Of_Object_Property_References

This property, of type BACnetLIST of BACnetDeviceObjectPropertyReference, specifies the Device Identifiers, Object Identifiers, and Property Identifiers of the properties to be written with specific values at changes of the timer state.

If this property is writable, it may be restricted to only support references to objects inside of the device containing the Timer object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with 'Error Class' of PROPERTY and 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property is set to reference an object outside the device containing the Timer object, the method used for writing to the referenced property for the purpose of controlling the property is a local matter. The only restriction on the method of writing to the referenced property is that the device be capable of using WriteProperty for this purpose so as to be interoperable with all BACnet devices.

12.57.22 Priority_For_Writing

This property, of type Unsigned, defines the priority at which the referenced properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. The value 1 is considered the highest priority and 16 the lowest. See Clause 19.2.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.57.23 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.57.24 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.57.25 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.26 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.27 Alarm_Values

This property, of type BACnetLIST of BACnetTimerState, is the pAlarmValues parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.57.28 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.57.29 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.57.30 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.57.31 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.57.32 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.57.33 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.57.34 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.57.35 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.57.36 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.57.37 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.57.38 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.57.39 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.57.40 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.57.41 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.57.42 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.58 Elevator Group Object Type

The Elevator Group object type defines a standardized object whose properties represent the externally visible characteristics of a group of lifts or escalators (a group being defined as those lifts or escalators controlled by a single supervisory controller).

The following figure illustrates an example of a structure of objects that represent lifts and escalators of a building.

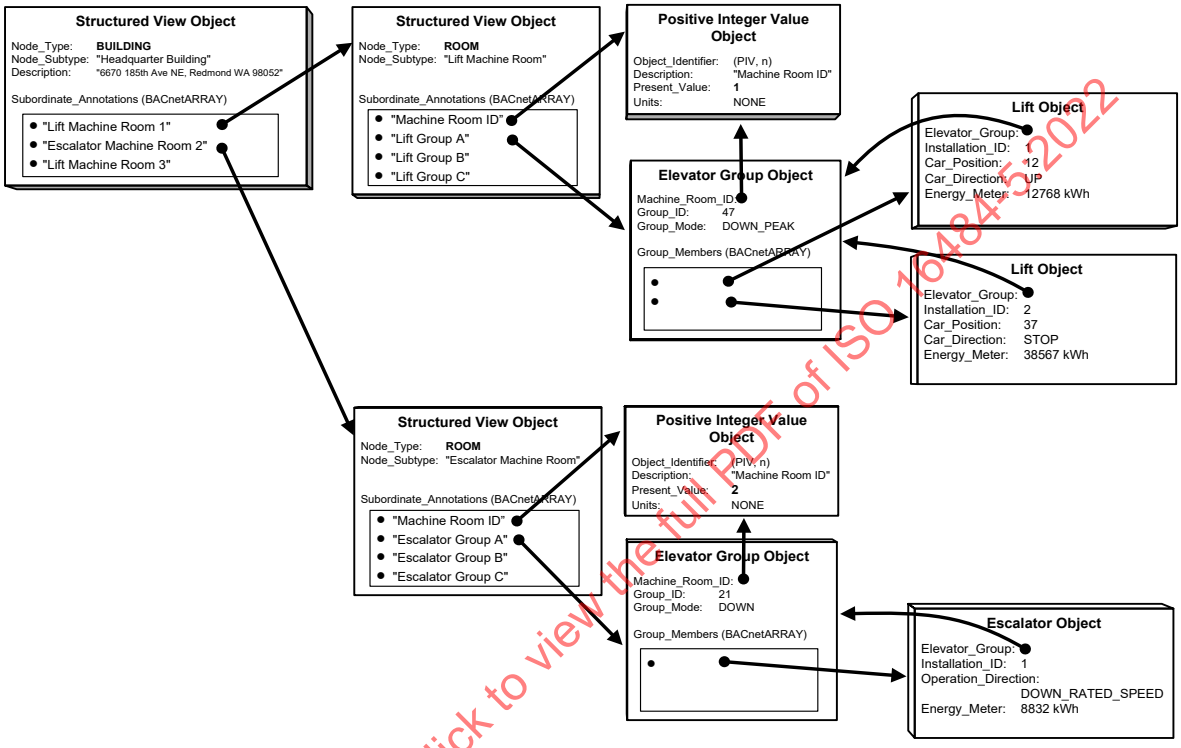


Figure 12-22. Elevator Object Structure Example

The Elevator Group object type and its properties are summarized in Table 12-76 and described in detail in this clause.

Table 12-76. Properties of the Elevator Group Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Machine_Room_ID	BACnetObjectIdentifier	R
Group_ID	Unsigned8	R
Group_Members	BACnetARRAY[N] of BACnetObjectIdentifier	R
Group_Mode	BACnetLiftGroupMode	O[1]
Landing_Calls	BACnetLIST of BACnetLandingCallStatus	O[1]
Landing_Call_Control	BACnetLandingCallStatus	O[1]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[2]
Auditable_Operations	BACnetAuditOperationFlags	O[2]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties shall be present only if this object represents a group of lifts.
[2] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.58.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.58.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.58.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ELEVATOR_GROUP.

12.58.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.58.5 Machine_Room_ID

This property, of type BACnetObjectIdentifier, shall reference the Positive Integer Value Object whose Present_Value property contains the identification number for the machine room that contains the group of lifts or escalators represented by this object.

If there is no such identification number, this property shall contain an object instance number of 4194303.

12.58.6 Group_ID

This property, of type Unsigned8, shall represent the identification number for the group of lifts or escalators represented by this object. This identification number shall be unique for the groups in this machine room, but might not be otherwise unique in the building.

12.58.7 Group_Members

This property, of type BACnetARRAY of BACnetObjectIdentifier, references the member objects of the group. Each element shall contain the object identifier of a Lift or Escalator object representing lifts or escalators contained within the group represented by this object.

For inclusion of a multi-deck lift, represented by a chain of Lift objects, only one Lift object out of the chain is referenced. The selection of which Lift object is referenced is a local matter.

12.58.8 Group_Mode

This property, of type BACnetLiftGroupMode, shall convey the operating mode of the group of lifts. This is used to represent some special traffic modes of control of the supervisory controller of a group of lifts. Supervisory controllers are not required to support all modes. Under a special traffic mode, the car dispatching algorithm may be different.

UNKNOWN	The current operating mode of the lift group is unknown.
NORMAL	The lift group is in normal operating mode, and no special operating mode is in place.
DOWN_PEAK	Most passengers want to leave the building. This usually happens before lunch break or at the close of business.
TWO_WAY	Many passengers want to get to, or leave, a particular floor. This usually happens when there is a special function, ceremony, meeting or conference on a particular floor.
FOUR_WAY	Many passengers want to move between two particular floors. This can happen, for example, at a school when a group of students wants to travel from classroom to classroom on two different floors.
EMERGENCY_POWER	The whole lift group is operating under an emergency power supply. In such a condition, only limited services are provided and most lifts are in a homing mode. This situation can also occur during a fire alarm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

UP_PEAK	Most passengers gather at the main terminal, usually the ground floor, to get to different floors of the building. This situation usually happens in the morning right before office hours or right after lunch.

12.58.9 Landing_Calls

This property, of type BACnetLIST of BACnetLandingCallStatus, may be present if the Elevator Group object represents a group of lifts. Each element of this list shall represent a currently active call for the group of lifts.

12.58.10 Landing_Call_Control

This property, of type BACnetLandingCallStatus, may be present if the Elevator Group object represents a group of lifts. If it is present, it shall be writable. A write to this property is equivalent to a passenger pressing a call button at a landing, indicating either desired direction of travel or destination floor.

12.58.11 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.58.12 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.58.13 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.58.14 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.58.15 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.58.16 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.59 Lift Object Type

The lift object type defines a standardized object whose properties represent the externally visible characteristics of a lift.

As there could be multiple car doors on a lift car, there could also be up to the same number of landing doors for the lift car at each floor. Normally, a landing door is driven by the car door as landing doors are not powered. So the status of a lift car door also reflects the status of the corresponding landing door at a particular floor. The following figure illustrates a lift car that has two doors. On the floor shown, there is no landing door for car door [1].

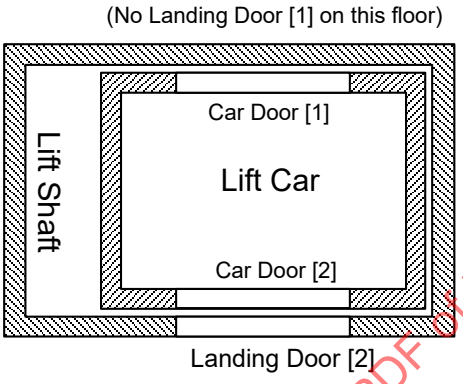


Figure 12-23. Example Two Door Lift Car

Properties that are related to the doors of a car are of type BACnetARRAY. In each of these arrays, the element with the same array index is related to the same car door or corresponding landing door. The array properties Car_Door_Text, Assigned_Landing_Calls, Making_Car_Call, Registered_Car_Call, Car_Door_Status, Car_Door_Command, and Landing_Door_Status, if present, shall be of the same size. The number of array elements in these properties shall be equal to the number of car doors present in the lift car. The assignment of a car door to an array index is a local matter.

Lift objects may represent multi-deck lift cars. In this configuration, each deck is represented by its own Lift object. A Lift object representing a deck shall reference, in the properties Lower_Deck and Higher_Deck, the Lift objects representing its adjacent lower and higher decks, respectively. In the Lift object for the lowest deck, the Lower_Deck property may be absent. If the property is present, the object instance shall be 4194303. In the Lift object for the highest deck, the Higher_Deck property may be absent. If the property is present, the object instance shall be 4194303. Any synchronization or sharing of property values among all Lift objects of a multi-deck lift is a local matter. Write requests to properties that are synchronized or shared shall be equally accepted by any of the objects whose respective properties are synchronized or shared. The following figure illustrates an example of Lift objects for an elevator group of one double deck lift car and one triple deck lift car.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

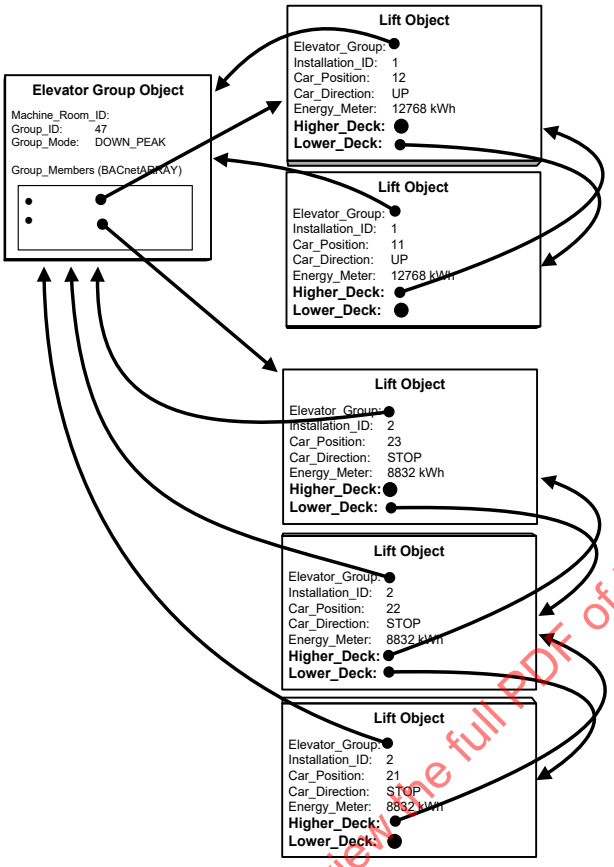


Figure 12-24. Example Elevator Group of Multi-Deck Lift Cars

Lift objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm on the Passenger_Alarm property. The pAlarmValues parameter shall contain the value TRUE.

For reliability-evaluation, the FAULT_LISTED fault algorithm can be applied.

The Lift object type and its properties are summarized in Table 12-77 and described in detail in this clause.

Table 12-77. Properties of the Lift Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Elevator_Group	BACnetObjectIdentifier	R
Group_ID	Unsigned8	R
Installation_ID	Unsigned8	R
Floor_Text	BACnetARRAY[N] of CharacterString	O
Car_Door_Text	BACnetARRAY[N] of CharacterString	O
Assigned_Landing_Calls	BACnetARRAY[N] of BACnetAssignedLandingCalls	O
Making_Car_Call	BACnetARRAY[N] of Unsigned8	O
Registered_Car_Call	BACnetARRAY[N] of BACnetLiftCarCallList	O
Car_Position	Unsigned8	R
Car_Moving_Direction	BACnetLiftCarDirection	R
Car_Assigned_Direction	BACnetLiftCarDirection	O

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Car_Door_Status	BACnetARRAY[N] of BACnetDoorStatus	R
Car_Door_Command	BACnetARRAY[N] of BACnetLiftCarDoorCommand	O
Car_Door_Zone	BOOLEAN	O
Car_Mode	BACnetLiftCarMode	O
Car_Load	REAL	O
Car_Load_Units	BACnetEngineeringUnits	O[1]
Next_Stopping_Floor	Unsigned8	O
Passenger_Alarm	BOOLEAN	R
Time_Delay	Unsigned	O[2,3]
Time_Delay_Normal	Unsigned	O[3]
Energy_Meter	REAL	O
Energy_Meter_Ref	BACnetDeviceObjectReference	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Car_Drive_Status	BACnetLiftCarDriveStatus	O
Fault_Signals	BACnetLIST of BACnetLiftFault	R
Landing_Door_Status	BACnetARRAY[N] of BACnetLandingDoorStatus	O
Higher_Deck	BACnetObjectIdentifier	O
Lower_Deck	BACnetObjectIdentifier	O
Event_Detection_Enable	BOOLEAN	O[2,3]
Notification_Class	Unsigned	O[2,3]
Event_Enable	BACnetEventTransitionBits	O[2,3]
Event_State	BACnetEventState	O[2,3]
Acked_Transitions	BACnetEventTransitionBits	O[2,3]
Notify_Type	BACnetNotifyType	O[2,3]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[2,3]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[3]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[3]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[3]
Event_Algorithm_Inhibit	BOOLEAN	O[3,4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[5]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[6]
Auditable_Operations	BACnetAuditOperationFlags	O[6]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] Car_Load_Units shall be present if, and only if, Car_Load is present
[2] These properties are required if the object supports intrinsic reporting.
[3] These properties shall be present only if the object supports intrinsic reporting.
[4] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[5] If this property is present, then the Reliability property shall be present.
[6] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.59.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.59.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

© ISO 2022 – All rights reserved

12.59.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be LIFT.

12.59.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.59.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the lift. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the properties Making_Car_Call, Car_Door_Command, or Car_Mode have been overridden by some mechanism local to the lift this object represents. In this context, "overridden" is taken to mean that the lift operation is no longer tracking changes to these properties, and these properties are no longer indicating the respective values applied by the lift. Other properties indicating status of the lift are still tracking the current status of the lift. Otherwise logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.59.6 Elevator_Group

This property, of type BACnetObjectIdentifier, shall reference the Elevator Group object whose Group_Members property contains a reference to this Lift object.

If there is no such Elevator Group object, this property shall contain an object instance of 4194303.

12.59.7 Group_ID

This property, of type Unsigned8, shall represent the identification number for the group of lifts that contains the lift represented by this object.

12.59.8 Installation_ID

This property, of type Unsigned8, shall represent the identification number for the lift represented by this object. This identification number shall be unique for the lift in this group, but might not be otherwise unique for other lifts in the machine room or the building.

12.59.9 Floor_Text

This property, of type BACnetARRAY of CharacterString, represents the descriptions or names for the floors. The universal floor number serves as an index into this array. The size of this array shall match the highest universal floor number served by this lift.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.59.10 Car_Door_Text

This property, of type BACnetARRAY of CharacterString, represents the descriptions or names for the doors of the lift car. Each array element represents the description or name for the door of the car assigned to this array element.

12.59.11 Assigned_Landing_Calls

This property, of type BACnetARRAY of BACnetAssignedLandingCalls, shall represent the current landing calls and their direction for the lift represented by this object. Each array element represents the list of assigned landing calls for the door of the car assigned to this array element.

Each element in BACnetAssignedLandingCalls consists of the universal floor number and the direction, of type BACnetLiftCarDirection, which may be one of these values:

UP	The landing call is for upward travel.
DOWN	The landing call is for downward travel.
UP_AND_DOWN	The landing call is for both upward and downward travel having been initiated by two different passengers.

12.59.12 Making_Car_Call

This property, of type BACnetARRAY of Unsigned8, indicates the last car calls written to this property. Writing to this property is equivalent to a passenger requesting that the car stop at the designated floor. Each array element represents the last car call written to this property for the door of the car assigned to this array element. If no car call has been written to an array element, the array element shall indicate a value of zero.

12.59.13 Registered_Car_Call

This property, of type BACnetARRAY of BACnetLiftCarCallList, represents the lists of currently registered car calls (requests to stop at particular floors using a particular door) for this lift. Each array element represents the list of universal floor numbers for which calls are registered for the door of the car assigned to this array element.

12.59.14 Car_Position

This property, of type Unsigned8, indicates the universal floor number of this lift's car position.

12.59.15 Car_Moving_Direction

This property, of type BACnetLiftCarDirection, represents whether or not this lift's car is moving, and if so, in which direction. Car_Moving_Direction can take on one of these values:

UNKNOWN	The current moving direction of the lift is unknown.
STOPPED	The lift car is not moving.
UP	The lift car is moving upward.
DOWN	The lift car is moving downward.

12.59.16 Car_Assigned_Direction

This property, of type BACnetLiftCarDirection, represents the direction the lift is assigned to move, based on current car calls. Car_Assigned_Direction can take on these values:

UNKNOWN	The direction assigned to the lift is unknown.
NONE	No direction is assigned to the lift car.
UP	The lift car is assigned to move upward.
DOWN	The lift car is assigned to move downward.
UP_AND_DOWN	The lift car is assigned to either move upward or downward.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary lift car direction assignments other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.59.17 Car_Door_Status

This property, of type BACnetARRAY of BACnetDoorStatus, indicates the status of the doors on the car. Each array element indicates the status of the car door assigned to this array element. Each array element can have one of these enumerated values:

UNKNOWN	The status of the car door is unknown.
CLOSING	The car door is closing.
CLOSED	The car door is fully closed but not yet locked.
OPENING	The car door is opening.
OPENED	The car door is fully opened.
SAFETY_LOCKED	The car door is fully closed and locked.
LIMITED_OPENED	The car door remains in a position between fully closed and fully opened.

12.59.18 Car_Door_Command

This property, of type BACnetARRAY of BACnetLiftCarDoorCommand, indicates the last pending car door commands written to this property. Writing to this property is equivalent to a passenger requesting that the respective car door be opened or closed. Each array element represents the last pending car door command for the door of the car assigned to this array element.

Once the respective car door command is executed or no longer applicable, e.g., the car is now moving, the respective array element shall revert to NONE.

NONE	No car door command was written, or there is no pending or executing car door command.
OPEN	The car door was commanded to open, and execution is pending or in progress.
CLOSE	The car door was commanded to close, and execution is pending or in progress.

12.59.19 Car_Door_Zone

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the car is in the door zone, the region near the landing where the door is permitted to start opening.

12.59.20 Car_Mode

This property, of type BACnetLiftCarMode, shall indicate the current operational mode of the car. Car_Mode can take on these values:

UNKNOWN	The current operational mode of the lift is unknown.
NORMAL	The lift is operating normally.
VIP	The lift is operating in "very important person" mode between a particular floor and the main terminal. In this mode, a particular lift is reserved for the use of the "VIPs."
HOMING	The lift is returning to the main terminal and is going to stay there and not provide any further service.
PARKING	The lift car is manually or automatically parked at a particular floor and will not provide any further service. This usually happens in a low traffic condition for the purpose of energy saving.
ATTENDANT_CONTROL	The lift is being manually controlled by an attendant in the car.
FIREFIGHTER_CONTROL	The lift is under a firefighter lift or firefighting lift mode of control. This usually happens during a fire alarm when firemen are on the site.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

EMERGENCY_POWER	The lift is operating on emergency power. The lift may be limited in its operation, such as moving to a predefined floor only.
INSPECTION	The lift is under inspection. Control of the lift is performed from a control panel on the car roof. The lift is not available for normal operation.
CABINET_RECALL	Control of the lift is performed from a control panel in the control cabinet. The lift is not available for normal operation.
EARTHQUAKE_OPERATION	The lift will stop operation at a predefined floor for earthquake evacuation.
FIRE_OPERATION	The lift is returning to the fire evacuation terminal and may stay there and may not provide any further service.
OUT_OF_SERVICE	The lift is not available for service.
OCCUPANT_EVACUATION	The lift is under an occupant evacuation mode of control. This usually happens during a fire alarm when the lift is used for evacuation of occupants.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary car operation modes other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.59.21 Car_Load

This property, of type REAL, indicates the load in the car, both passengers and goods. The value of the Car_Load property shall be in units as indicated by the Car_Load_Units property..

12.59.22 Car_Load_Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of the Car_Load property. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.59.23 Next_Stopping_Floor

This property, of type Unsigned8, indicates the universal floor number where the car will stop next when underway. If the car is not in motion, this property indicates the current universal floor number.

12.59.24 Passenger_Alarm

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the passenger alarm has been activated.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.59.25 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.59.26 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.59.27 Energy_Meter

This property, of type REAL, indicates the accumulated energy consumption by the lift. The units shall be kilowatt-hours. When this value reaches 99999 kWh, it shall wrap to a value near zero; the particular value to which it wraps is a local matter.

If the Energy_Meter_Ref property is present and initialized (contains an instance other than 4194303), then the Energy_Meter property, if present, shall contain a value of 0.0.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.59.28 Energy_Meter_Ref

This property, of type BACnetDeviceObjectReference, references the object which indicates the accumulated energy consumption by the lift.

12.59.29 Reliability

This property, of type BACnetReliability, provides an indication of whether the properties of this object or the operation of the lift represented by this object are "reliable" as far as the BACnet device can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.
See Clause 13.4 for fault algorithm parameter descriptions.

12.59.30 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the object is decoupled from the lift that this object represents.

When Out_Of_Service is TRUE:

(a) the object does not track the status of the lift;
(b) the object will not control the lift operation and the value of this property shall have no effect on the operation of the lift this object represents;
(c) the status properties Assigned_Landing_Calls, Registered_Car_Call, Car_Position, Car_Moving_Direction, Car_Assigned_Direction, Car_Door_Status, Car_Door_Zone, Car_Load, Next_Stopping_Floor, Passenger_Alarm, Energy_Meter, Car_Drive_Status, Fault_Signals, and Landing_Door_Status shall not track the status of the lift and shall be writable;
(d) the properties Making_Car_Call, Car_Door_Command, and Car_Mode, shall not track the respective values currently applied by the lift, shall not have any effect on the operation of the lift, and shall be writable
(e) the properties listed in this clause normally indicating status or currently applied control values shall be writable to allow simulating specific conditions or for testing purposes;
(f) other functions that depend on the state of any of the aforementioned properties shall respond to changes made to these properties, as if those changes had occurred in the lift.

12.59.31 Car_Drive_Status

This property, of type BACnetLiftCarDriveStatus, shall indicate the current status of the lift's motor drive system. Car_Drive_Status can take on these values:

UNKNOWN	The status of the lift's motor drive system is unknown.
STATIONARY	The motor is not moving.
BRAKING	The brake is operating.
ACCELERATE	The lift car is moving along an acceleration profile, up or down.
DECELERATE	The lift car is moving along a deceleration profile, up or down.
RATED_SPEED	The lift car is moving under rated speed, up or down.
SINGLE_FLOOR_JUMP	The lift car is going from one floor to the next consecutive floor, up or down.
TWO_FLOOR_JUMP	The lift car is moving two floors up or down.
THREE_FLOOR_JUMP	The lift car is moving three floors up or down.
MULTI_FLOOR_JUMP	The lift car is moving four or more floors up or down.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary car drive status values other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.59.32 Fault_Signals

This property, of type BACnetLIST of BACnetLiftFault, represents a list of values that indicates fault conditions of the lift. The Fault_Signals property may be empty or contain any set of the following values, without duplicates:

CONTROLLER_FAULT	The fault is due to a malfunctioning controller.
DRIVE_AND_MOTOR_FAULT	The fault is related to the motor drive, either electrical or mechanical.
GOVERNOR_AND_SAFETY_GEAR_FAULT	The fault is related to the governor and safety gear system.
LIFT_SHAFT_DEVICE_FAULT	The fault is related to a device inside the lift shaft, such as the position detector, a limit switch, etc.
POWER_SUPPLY_FAULT	The fault is related to the supply of electrical power, not to the lift system itself.
SAFETY_INTERLOCK_FAULT	There is a fault with the chain of car and landing doors locks which must indicate a fully closed and locked condition before a lift car can move.
DOOR_CLOSING_FAULT	A door is failing to close.
DOOR_OPENING_FAULT	A door is failing to open.
CAR_STOPPED_OUTSIDE_LANDING_ZONE	The car stopped outside the landing zone while in normal operation.
CALL_BUTTON_STUCK	Any car or landing call continues to remain registered after the car has stopped at the floor.
START_FAILURE	The lift failed to start moving.
CONTROLLER_SUPPLY_FAULT	The power supply for the lift controller is out of its specified range or failed.
SELF_TEST_FAILURE	Any self-test function failed.
RUNTIME_LIMIT_EXCEEDED	The lift did not reach the expected zone.
POSITION_LOST	The lift control has lost information on the position of the car.
DRIVE_TEMPERATURE_EXCEEDED	The temperature of the drive system of the lift exceeded its limits.
LOAD_MEASUREMENT_FAULT	The car load measurement system is in a fault condition.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary lift fault signals other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

The mechanism for determining the existence of a fault condition is a local matter.

This property is the value of the pMonitoredList parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

12.59.33 Landing_Door_Status

This property, of type BACnetARRAY of BACnetLandingDoorStatus, represents the status of the landing doors on the floors served by this lift. Each element of this array represents the list of landing doors for the door of the car assigned to this array element.

A landing door status includes the universal floor number and the currently active door status for the landing door. The status values that each landing door status can take on are:

UNKNOWN	The landing door status is unknown.
NONE	There is no landing door for the respective car door.
CLOSING	The landing door is closing.
CLOSED	The landing door is fully closed but not locked.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OPENING	The landing door is opening.
OPENED	The landing door is fully opened.
SAFETY_LOCK	The landing door is fully closed and locked.
LIMITED_OPENED	The landing door remains in a state between fully closed and fully opened.

12.59.34 Higher_Deck

This property, of type BACnetObjectIdentifier, references the Lift object that is representing the car deck above the car deck represented by this object.

If this property is present, and there is no higher deck, then the object instance shall be 4194303.

12.59.35 Lower_Deck

This property, of type BACnetObjectIdentifier, references the Lift object that is representing the car deck below the car deck represented by this object.

If this property is present, and there is no lower deck, then the object instance shall be 4194303.

12.59.36 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.59.37 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.59.38 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.59.39 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.59.40 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.59.41 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.59.42 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.59.43 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.59.44 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.59.45 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.59.46 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.59.47 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.59.48 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.59.49 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.59.50 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.59.51 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.59.52 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.59.53 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60 Escalator Object Type

The Escalator object type defines a standardized object whose properties represent the externally visible characteristics of an escalator.

Escalator objects that support intrinsic reporting shall apply the CHANGE_OF_STATE event algorithm on the Passenger_Alarm property. The pAlarmValues parameter shall contain the value TRUE.

For reliability-evaluation, the FAULT_LISTED fault algorithm can be applied.

The Escalator object type and its properties are summarized in Table 12-78 and described in detail in this clause.

Table 12-78. Properties of the Escalator Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Elevator_Group	BACnetObjectIdentifier	R
Group_ID	Unsigned8	R
Installation_ID	Unsigned8	R
Power_Mode	BOOLEAN	O
Operation_Direction	BACnetEscalatorOperationDirection	R
Escalator_Mode	BACnetEscalatorMode	O
Energy_Meter	REAL	O
Energy_Meter_Ref	BACnetDeviceObjectReference	O
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Fault_Signals	BACnetLIST of BACnetEscalatorFault	O
Passenger_Alarm	BOOLEAN	R
Time_Delay	Unsigned	O[1,2]
Time_Delay_Normal	Unsigned	O[2]
Event_Detection_Enable	BOOLEAN	O[1,2]
Notification_Class	Unsigned	O[1,2]
Event_Enable	BACnetEventTransitionBits	O[1,2]
Event_State	BACnetEventState	O[1,2]
Acked_Transitions	BACnetEventTransitionBits	O[1,2]
Notify_Type	BACnetNotifyType	O[1,2]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[1,2]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[2]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[2]
Event_Algorithm_Inhibit	BOOLEAN	O[2,3]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[2]
Reliability_Evaluation_Inhibit	BOOLEAN	O[4]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Audit_Level	BACnetAuditLevel	O[5]
Auditable_Operations	BACnetAuditOperationFlags	O[5]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties are required if the object supports intrinsic reporting.
[2] These properties shall be present only if the object supports intrinsic reporting.
[3] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[4] If this property is present, then the Reliability property shall be present.
[5] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.60.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.60.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ESCALATOR.

12.60.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.60.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the escalator. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the property Escalator_Mode has been overridden by some mechanism local to the escalator this object represents. In this context, "overridden" is taken to mean that the escalator operation is no longer tracking changes to these properties, and these properties are no longer indicating the respective values applied by the escalator. Other properties indicating status of the escalator are still tracking the current status of the escalator. Otherwise logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.60.6 Elevator_Group

This property, of type BACnetObjectIdentifier, shall reference the Elevator Group object whose Group_Members property contains a reference to this Escalator object.

If there is no such Elevator Group object, this property shall contain an object instance of 4194303.

12.60.7 Group_ID

This property, of type Unsigned8, shall represent the identification number for the group of escalators that contains the escalator represented by this object.

12.60.8 Installation_ID

This property, of type Unsigned8, shall represent the identification number for the escalator represented by this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60.9 Power_Mode

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the escalator is powered (independent of whether it is moving).

12.60.10 Operation_Direction

This property, of type BACnetEscalatorOperationDirection, represents the direction and speed in which this escalator is presently moving. Operation_Direction can take on these values:

UNKNOWN	The current operation direction is unknown.
STOPPED	The escalator or slanted passenger conveyor is not moving.
UP_RATED_SPEED	The escalator or slanted passenger conveyor is moving upward at rated speed.
UP_REDUCED_SPEED	The escalator or slanted passenger conveyor is moving upward at a reduced speed. This is for energy conservation when there are no passengers.
DOWN_RATED_SPEED	The escalator or slanted passenger conveyor is moving downward at rated speed.
DOWN_REDUCED_SPEED	The escalator or slanted passenger conveyor is moving downward under reduced speed. Again, this is to save energy.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary escalator operation direction values other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.60.11 Escalator_Mode

This property, of type BACnetEscalatorMode, shall indicate the current operational mode of the escalator. Escalator_Mode can take on these values:

UNKNOWN	The current operational mode of the escalator or slanted passenger conveyor is unknown.
STOP	The escalator or slanted passenger conveyor is not moving.
UP	The escalator or slanted passenger conveyor is moving upward.
DOWN	The escalator or slanted passenger conveyor is moving downward.
INSPECTION	The escalator or slanted passenger conveyor is under inspection. Control of the escalator is performed from a control panel on the escalator. The escalator is not available for normal operation.
OUT_OF_SERVICE	The escalator or slanted passenger conveyor is not available for service.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary escalator or slanted passenger conveyor operation modes other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

12.60.12 Energy_Meter

This property, of type REAL, indicates the accumulated energy consumption by the escalator. The units shall be kilowatt-hours. When this value reaches 99999 kWh, it shall wrap to a value near zero; the particular value to which it wraps is a local matter.

If the Energy_Meter_Ref property is present and initialized (contains an instance other than 4194303), then the Energy_Meter property, if present, shall contain a value of 0.0.

12.60.13 Energy_Meter_Ref

This property, of type BACnetDeviceObjectReference, references the object which indicates the accumulated energy consumption by the escalator.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60.14 Reliability

This property, of type BACnetReliability, provides an indication of whether the properties of this object or the operation of the escalator represented by this object are "reliable" as far as the BACnet device can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.
See Clause 13.4 for fault algorithm parameter descriptions.

12.60.15 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the object is decoupled from the escalator that this object represents.

When Out_Of_Service is TRUE:

(a) the object does not track the status of the escalator;
(b) the object will not control the escalator operation and the value of this property shall have no effect on the operation of the escalator this object represents;
(c) the status properties Power_Mode, Operation_Direction, Energy_Meter, Fault_Signals, and Passenger_Alarm shall not track the status of the escalator and shall be writable;
(d) the property Escalator_Mode shall not have any effect on the operation of the escalator, shall not track the respective value currently applied by the lift and shall be writable;
(e) the properties listed in this clause normally indicating status or currently applied control values shall be writable to allow simulating specific conditions or for testing purposes;
(f) other functions that depend on the state of any of the aforementioned properties shall respond to changes made to these properties, as if those changes had occurred in the escalator.

12.60.16 Fault_Signals

This property, of type BACnetLIST of BACnetEscalatorFault, represents a list of values that indicates fault conditions of the escalator. The Fault_Signals property may be empty or contain any set of the following values, without duplicates.

CONTROLLER_FAULT	The fault is due to a malfunctioning controller.
DRIVE_AND_MOTOR_FAULT	The fault is related to the motor drive, either electrical or mechanical.
MECHANICAL_COMPONENT_FAULT	The fault is related to the failure of a mechanical component.
OVERSPEED_FAULT	The fault is due to overspeed operation, either up or down.
POWER_SUPPLY_FAULT	The fault is related to the electric power supply; one or more phases of the electrical power has failed.
SAFETY_DEVICE_FAULT	The fault is due to the triggering of any of the escalator's safety devices.
CONTROLLER_SUPPLY_FAULT	The power supply for the escalator controller is out its specified range or failed.
DRIVE_TEMPERATURE_EXCEEDED	The temperature of the drive system of the escalator exceeded its limits.
COMB_PLATE_FAULT	A comb plate safety switch is activated. This may indicate that debris is lodged between the comb and steps of the escalator.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary escalator fault signals other than those defined by the standard. For proprietary extensions of this enumeration, see Clause 23.1 of this standard.

The mechanism for determining the existence of a fault condition is a local matter.

This property is the value of the pMonitoredList parameter for the object's fault algorithm. See Clause 13.4 for fault algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60.17 Passenger_Alarm

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the passenger alarm has been activated, thus stopping the escalator, and the alarm has not yet been cleared by a maintenance technician.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.60.18 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.60.19 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.60.20 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.60.21 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.60.22 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.60.23 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.60.24 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.60.25 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.60.26 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.60.27 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.60.28 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.60.29 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.60.30 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.60.31 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.60.32 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.60.33 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.60.34 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.60.35 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.60.36 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.60.37 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.61 Accumulator Object Type

The Accumulator object type defines a standardized object whose properties represent the externally visible characteristics of a device that indicates measurements made by counting pulses.

This object maintains precise measurement of input count values, accumulated over time. The accumulation of pulses represents the measured quantity in unsigned integer units. This object is also concerned with the accurate representation of values presented on meter read-outs. This includes the ability to initially set the Present_Value property to the value currently displayed by the meter (as when the meter is installed), and to duplicate the means by which it is advanced, including simulating a modulo-N divider prescaling the actual meter display value, as shown in Figure 12-25.

Typical applications of such devices are in peak load management and in accounting and billing management systems. This object is not intended to meet all such applications. Its purpose is to provide information about the quantity being measured, such as electric power, water, or natural gas usage, according to criteria specific to the application.

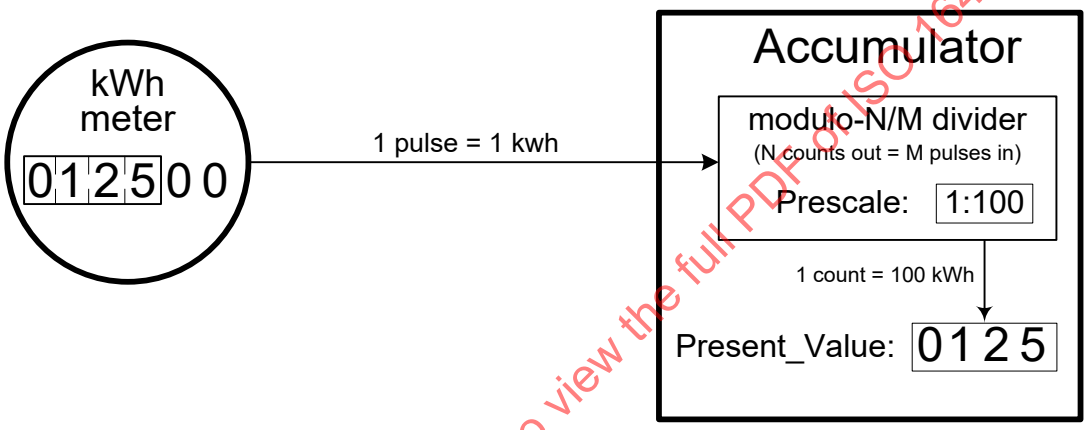


Figure 12-25. Example of an Accumulator object

Accumulator objects that support intrinsic reporting shall apply the UNSIGNED_RANGE event algorithm. For reliability-evaluation, the FAULT_OUT_OF_RANGE fault algorithm may be applied.

The object and its properties are summarized in Table 12-79 and described in detail in this clause.

Table 12-79. Properties of the Accumulator Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Scale	BACnetScale	R
Units	BACnetEngineeringUnits	R
Prescale	BACnetPrescale	O
Max_Pres_Value	Unsigned	R
Value_Change_Time	BACnetDateTime	O[2]
Value_Before_Change	Unsigned	O[2,3]
Value_Set	Unsigned	O[2,3]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Logging_Record	BACnetAccumulatorRecord	O
Logging_Object	BACnetObjectIdentifier	O
Pulse_Rate	Unsigned	O[1,4,7]
High_Limit	Unsigned	O[4,6]
Low_Limit	Unsigned	O[4,6]
Limit_Monitoring_Interval	Unsigned	O[4,7]
Notification_Class	Unsigned	O[4,6]
Time_Delay	Unsigned	O[4,6]
Limit_Enable	BACnetLimitEnable	O[4,6]
Event_Enable	BACnetEventTransitionBits	O[4,6]
Acked_Transitions	BACnetEventTransitionBits	O[4,6]
Notify_Type	BACnetNotifyType	O[4,6]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,6]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[6]
Event_Detection_Enable	BOOLEAN	O[4,6]
Event_Algorithm_Inhibit_Ref	BACnetObjectPropertyReference	O[6]
Event_Algorithm_Inhibit	BOOLEAN	O[6,8]
Time_Delay_Normal	Unsigned	O[6]
Reliability_Evaluation_Inhibit	BOOLEAN	O[9]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Fault_High_Limit	Unsigned	O[10]
Fault_Low_Limit	Unsigned	O[10]
Audit_Level	BACnetAuditLevel	O[11]
Auditable_Operations	BACnetAuditOperationFlags	O[11]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if either Value_Before_Change or Value_Set is writable.
[3] Either Value_Before_Change or Value_Set may be writable, but not both.
[4] These properties are required if the object supports intrinsic reporting.
[5] Footnote removed.
[6] These properties shall be present only if the object supports intrinsic reporting.
[7] If one of these properties is present, then both shall be present.
[8] Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
[9] If this property is present, then the Reliability property shall be present.
[10] These properties are required if the object supports the FAULT_OUT_OF_RANGE fault algorithm.
[11] This property shall be present only if the device supports audit reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.61.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.61.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.61.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCUMULATOR.

© ISO 2022 – All rights reserved

12.61.4 Present_Value

This property, of type Unsigned, indicates the count of the input pulses, prescaled if the Prescale property is present, acquired since the value was most recently set by writing to the Value_Set property.

The value of this property shall remain in the range from zero through Max_Pres_Value. All operations on the Present_Value property are performed modulo (Max_Pres_Value+1).

This property shall be writable when Out_Of_Service is TRUE.

12.61.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.61.6 Device_Type

This property, of type CharacterString, is a text description of the physical device represented by the Accumulator object. It will typically be used to describe the type of sensor represented by the Accumulator.

12.61.7 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Accumulator object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.8 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.61.9 Reliability

This property, of type BACnetReliability, provides an indication of whether the Present_Value property or the operation of the physical input in question is "reliable" as far as the BACnet device or operator can determine and, if not, why.

If a fault algorithm is applied, then this property shall be the pCurrentReliability parameter for the object's fault algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.61.10 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical input that the object represents is not in service.

When Out_Of_Service is TRUE:

(a) the Present_Value and Pulse_Rate properties are decoupled from the physical input and will not track changes to the physical input;
(b) the Reliability property, if present, and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical input;
(c) the Present_Value, and Pulse_Rate properties and the Reliability property, if present and capable of taking on values other than NO_FAULT_DETECTED,shall be writable to allow simulating specific conditions or for testing purposes;
(d) other functions that depend on the state of the Present_Value, Pulse_Rate or Reliability properties shall respond to changes made to these properties, as if those changes had occurred in the physical input.

12.61.11 Scale

This property, of type BACnetScale, indicates the conversion factor to be multiplied with the value of the Present_Value property to provide a value in the units indicated by Units. The choice of options for this property determine how the scaling operation (which is performed by the client reading this object) is performed:

Option	Datatype	Indicated Value in Units
float-scale	REAL	Present_Value x Scale
integer-scale	INTEGER	Present_Value x 10^{Scale}

12.61.12 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of the Present_Value when multiplied with the scaling factor indicated by Scale. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.61.13 Prescale

This property, of type BACnetPrescale, presents the coefficients that are used for converting the pulse signals generated by the measuring instrument into the value displayed by Present_Value. The conversions are performed using integer arithmetic in such a fashion that no measurement-generated pulse signals are lost in the conversion.

These coefficients might simply document a conversion performed prior to the reception of the input pulses by the Accumulator object, or they might actually be used by the Accumulator to convert input pulses into the value displayed by Present_Value. Whichever is done is a local matter.

The coefficients are as follows:

multiplier The numerator of the conversion factor expressed as a ratio of integers.
modulo-divide The denominator of the conversion factor expressed as a ratio of integers.

The conversion algorithm is performed as follows, utilizing a non-displayed variable called an accumulator:

For each input pulse:
Add the value of 'multiplier' to an accumulator and then,
while the accumulator is greater than or equal to the value of 'modulo-divide':
Increment the value of Present_Value by one, and
decrease the value of the accumulator by the value of 'modulo-divide'.

This procedure supports non-integral ratios of measurement pulses to Present_Value. For example, in an electrical metering application, the output of the voltage- and current-measuring systems might be 9000/1200 (scale / voltage*current) pulses per kWh, requiring the Accumulator object to accumulate 2/15 kWh/pulse. With this algorithm such pulses can be accurately accumulated and displayed when the units of Present_Value are KILOWATT_HOURS.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.61.14 Max_Pres_Value

This property, of type Unsigned, indicates the maximum value of the Present_Value property.

12.61.15 Value_Change_Time

This read-only property, of type BACnetDateTime, shall be present if the Present_Value property is adjustable by writing to the Value_Before_Change or Value_Set properties. It represents the date and time of the most recent occurrence of such a write operation. This property shall have an unspecified datetime to indicate that it is uninitialized; otherwise, it shall have a specific datetime value.

12.61.16 Value_Before_Change

This property, of type Unsigned, indicates the value of the Present_Value property just prior to the most recent write to the Value_Set or Value_Before_Change properties. If no such write has yet occurred, this property shall have the value zero. If this property is writable, the Value_Set property shall be read-only.

If this property is writable, the following series of operations, for which the associated properties are present, shall be performed atomically by the object when this property is written:

(1) The value of Present_Value shall be copied to the Value_Set property.
(2) The value written to Value_Before_Change shall be stored in the Value_Before_Change property.
(3) The current date and time shall be stored in the Value_Change_Time property.

While this series of operations is being performed, it is critical that any other process not change the Present_Value, Value_Set and Value_Before_Change properties.

12.61.17 Value_Set

This property, of type Unsigned, indicates the value of the Present_Value property after the most recent write to the Value_Set or Value_Before_Change properties. If no such write has yet occurred, this property shall have the value zero. If this property is writable, the Value_Before_Change property shall be read-only.

If this property is writable, the following series of operations, for which the associated properties are present, shall be performed atomically by the object when this property is written:

(1) The value of Present_Value shall be copied to the Value_Before_Change property.
(2) The value written to Value_Set shall be stored in both the Value_Set and Present_Value properties.
(3) The current date and time shall be stored in the Value_Change_Time property.

While this series of operations are being performed, it is critical that any other process not change the Present_Value, Value_Set and Value_Before_Change properties.

12.61.18 Logging_Record

This read-only property, of type BACnetAccumulatorRecord, is a list of values that must be acquired and returned "atomically" in order to allow proper interpretation of the data.

If the Logging_Object property is present, then, when Logging_Record is acquired by the object identified by Logging_Object, this list of values shall be saved and returned when read by other objects or devices. If the Logging_Object property is present and Logging_Record has not yet been acquired by the object identified by Logging_Object, 'timestamp' shall contain an unspecified datetime, 'present-value' and 'accumulated-value' shall contain the value zero, and 'accumulator-status' shall indicate STARTING.

The list of values ('timestamp', 'present-value', 'accumulated-value', and 'accumulator-status') shall be acquired from the underlying system when they reflect a stable state of the device (for example, they shall not be acquired when Present_Value has just been incremented but the corresponding increment of 'accumulated-value' has not yet occurred).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The items returned in the list of values are:

timestamp — The local date and time when the data was acquired.

present-value — The value of the Present_Value property.

accumulated-value — The short term accumulated value of the counter. The algorithm used to calculate accumulated-value is a function of the value of accumulator-status. If this is the initial read, the value returned shall be zero.

accumulator-status — An indication of the reliability of the data in this list of values.

The accumulator-status parameter may take on any of the following values:

{NORMAL, STARTING, RECOVERED, ABNORMAL, FAILED}

where the values are defined as follows:

NORMAL — No event affecting the reliability of the data has occurred during the period from the preceding to the current qualified reads of the Logging_Record property. In this case 'accumulated-value' shall be represented by the expression:

$$\text{accumulated-value} = \text{Present_Value}_{\text{current}} - \text{Present_Value}_{\text{previous}}$$

STARTING — This value indicates that the data in Logging_Records is either the first data to be acquired since startup by the object identified by Logging_Object (if 'timestamp' has a specific datetime) or that no data has been acquired since startup by the object identified by Logging_Object (in which case 'timestamp' has an unspecified datetime).

RECOVERED — One or more writes to Value_Before_Change or Value_Set have occurred since Logging_Record was acquired by the object identified by Logging_Object. For the case of a single write, 'accumulated-value' shall be represented by the expression:

$$\text{accumulated-value} = (\text{Present_Value}_{\text{current}} - \text{Value_Set}) + (\text{Value_Before_Change} - \text{Present_Value}_{\text{previous}})$$

ABNORMAL — The accumulation has been carried out, but some unrecoverable event such as the clock's time being changed by a significant amount since Logging_Record was acquired by the object identified by Logging_Object. (How much time is considered significant shall be a local matter.)

FAILED — The 'accumulated-value' item is not reliable due to some problem. The criteria for returning this value are a local matter.

Changes in the value of 'accumulator-status' shall occur only when the Logging_Record is acquired by the object identified by Logging_Object.

12.61.19 Logging_Object

This property, of type BACnetObjectIdentifier, indicates the object in the same device as the Accumulator object which, when it acquires Logging_Record data from the Accumulator object, shall cause the Accumulator object to acquire, present and store the data from the underlying system.

12.61.20 Pulse_Rate

This property, of type Unsigned, shall indicate the number of input pulses received during the most recent period specified by Limit_Monitoring_Interval. The mechanism that associates the input signal with the value indicated by this property is a local matter.

This property shall be writable when Out_Of_Service is TRUE.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMonitoredValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.61.21 High_Limit

This property is the pHighLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.22 Low_Limit

This property is the pLowLimit parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.23 Limit_Monitoring_Interval

This property, of type Unsigned, specifies the monitoring period in seconds for determining the value of Pulse_Rate. The use of a fixed or sliding time window for detecting pulse rate is a local matter.

12.61.24 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.61.25 Time_Delay

This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.26 Limit_Enable

This property, of type BACnetLimitEnable, is the pLimitEnable parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.27 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.61.28 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.61.29 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.61.30 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.61.31 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.61.32 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.61.33 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.61.34 Event_Algorithm_Inhibit_Ref

This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.61.35 Event_Algorithm_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read-only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.61.36 Time_Delay_Normal

This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.61.37 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.61.38 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.61.39 Fault_High_Limit

This property, of type Unsigned, shall specify a limit that the Pulse_Rate must exceed before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMaximumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.61.40 Fault_Low_Limit

This property, of type Unsigned, shall specify a limit that the Pulse_Rate must fall below before a fault event is generated by the FAULT_OUT_OF_RANGE fault algorithm.

If the object supports the FAULT_OUT_OF_RANGE fault algorithm, then this property shall be present and shall be the pMinimumNormalValue for the FAULT_OUT_OF_RANGE fault algorithm.

12.61.41 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.61.42 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.61.43 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.61.44 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.61.45 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.62 Staging Object Type

The Staging object type defines a standardized object whose properties represent the externally visible characteristics of a staged value. A "Staging" maps a numeric value onto multiple discrete ranges that define individual “stages” ($\mathbf{N_{stages}}$). Each Staging object is associated with a collection of references to binary valued objects ($\mathbf{N_{references}}$). Each Staging object may therefore control Binary Output, Binary Value, or Binary Lighting Output objects. Every stage specifies an arbitrary combination of ACTIVE/INACTIVE values to be written to these referenced objects. Stages are defined by a limit, a deadband, and the collection of values for the referenced objects.

Figure 12-26 shows a typical Staging object application with four stages ($\mathbf{N_{stages}}$ = 4) and two referenced binary objects ($\mathbf{N_{references}}$ = 2).

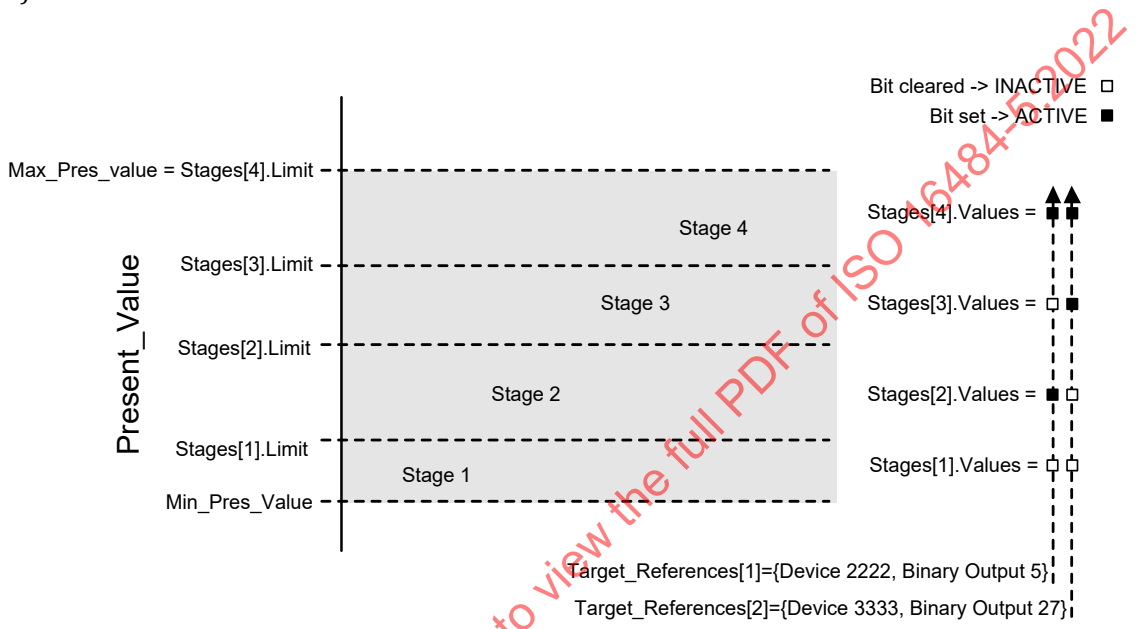


Figure 12-26. Typical Staging Application ($\mathbf{N_{stages}}$ = 4, $\mathbf{N_{references}}$ = 2)

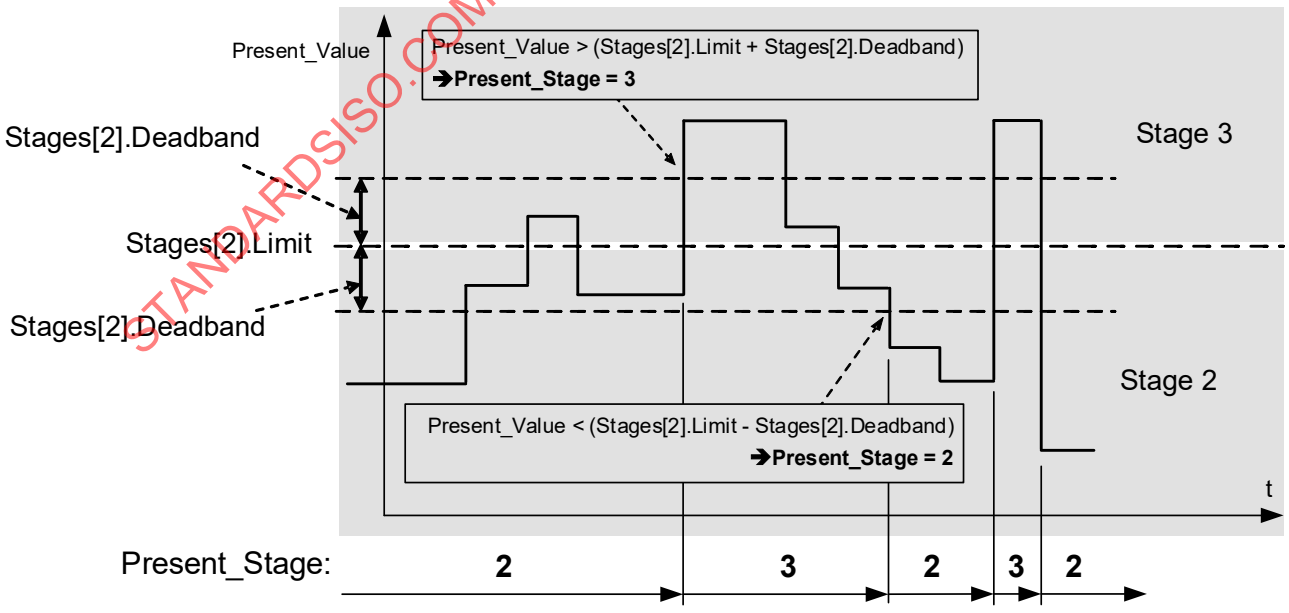


Figure 12-27. Stage Limits incorporate hysteresis through the use of a Deadband around each Limit

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Stages are defined by limits with a symmetrical deadband. A deadband greater than zero is used to prevent unwanted oscillation when the Present_Value is close to a limit. As the Present_Value increases, if it rises above the limit for a stage plus the deadband for that stage, the Present_Stage transitions to that stage+1. Similarly, as the Present_Value decreases, it must fall below the limit for a stage minus the deadband for that stage before Present_Stage transitions to that stage. The deadband is allowed to be zero (0.0).

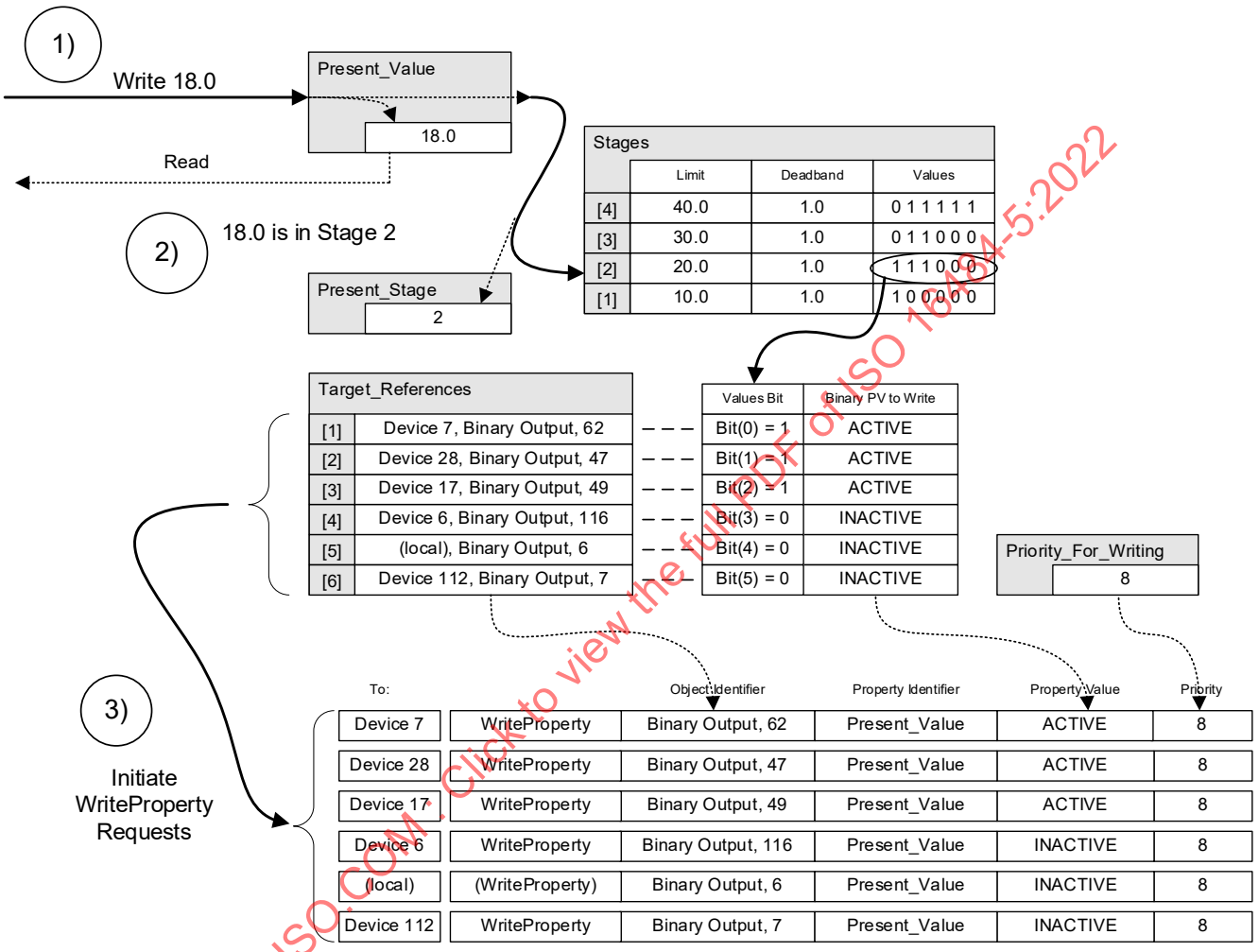


Figure 12-28. Pipeline of operations when Present_Value is written

Staging objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Staging objects that support intrinsic reporting shall apply the NONE event algorithm.

The object and its properties are summarized in Table 12-80 and described in detail in this clause.

Table 12-80. Properties of the Staging Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	W
Present_Stage	Unsigned	R
Stages	BACnetARRAY[N] of BACnetStageLimitValue	R[1]
Stage_Names	BACnetARRAY[N] of CharacterString	O[1]
Status_Flags	BACnetStatusFlags	R

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Property Identifier	Property Datatype	Conformance Code
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Description	CharacterString	O
Units	BACnetEngineeringUnits	R
Target_References	BACnetARRAY[N] of BACnetDeviceObjectReference	R[2]
Priority_For_Writing	Unsigned(1..16)	R
Default_Present_Value	REAL	O
Min_Pres_Value	REAL	R
Max_Pres_Value	REAL	R
COV_Increment	REAL	O[3]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5,6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[4,5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[7,8,9]
Audit_Level	BACnetAuditLevel	O[10]
Auditable_Operations	BACnetAuditOperationFlags	O[10]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

1 The array size of this property is $\mathbf{N}_{\mathbf{stages}}$.
2 The array size of this property is $\mathbf{N}_{\mathbf{references}}$.
3 This property is required if the object supports COV reporting.
4 These properties are required if the object supports intrinsic reporting.
5 These properties shall be present only if the object supports intrinsic reporting.
6 This property, if present, is required to be read-only.
7 This property is required if the object supports the value source mechanism.
8 This property shall be present only if the object supports the value source mechanism.
9 This property shall be writable as described in Clause 19.5.
10 This property shall be present only if the device supports audit reporting.

12.62.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.62.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.62.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be STAGING.

12.62.4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the Staging object. If Present_Value is written with a value less than Min_Pres_Value, then it shall be clamped to Min_Pres_Value. If Present_Value is written with a value greater than Max_Pres_Value, then it shall be clamped to Max_Pres_Value.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Whenever Present_Value is changed, the new value shall be compared with the 'Limit' values for the entire Stages array using the following algorithm:

```
ops = current Present_Stage
npv = new Present_Value

//check if value should remain in the current stage
If (ops != 0) then
        upperBound = Stages[ops].Limit + Stages[ops].Deadband
        if (ops > 1) then
                lowerBound = Stages[ops-1].Limit - Stages[ops-1].Deadband
        else
                lowerBound = Min_Pres_Value
        endif
        if (npv <= upperBound AND npv >= lowerBound) then
                Present_Value = npv
                exit algorithm   //no change to current stage, stop algorithm
        endif
endif

// calculate the new stage
Present_Stage = Nstages
for (i=1 to Nstages-1, step +1)
        if npv <= (Stages[i].Limit) then
                Present_Stage = i
                exit for         //found the correct stage, stop iteration
        endif
next i

Present_Value = npv
```

Figure 12-29. Pseudocode Algorithm for Evaluating Present_Value and Present_Stage

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.62.4.1 Writing to Referenced Objects

Changes to Present_Stage shall cause a write of ACTIVE or INACTIVE to the corresponding object's Present_Value for each Target_References array element. For each bit (Index = 0 to $\mathbf{N}_{\mathbf{references}}$ -1) in the Stages[Present_Stage].Values bitstring, if the bit is set (1), an ACTIVE value shall be written, or if clear (0), then an INACTIVE value shall be written to the Target_References[Index +1] object's Present_Value.

Writes to Present_Value that subsequently trigger writing to referenced objects due to reevaluation of Present_Stage, are not expected to wait until the reference writes occur before returning a Result(+) or Result(-) for the write to Present_Value. Subsequently if any write to a referenced object fails, Reliability shall be changed to COMMUNICATION_FAILURE. The COMMUNICATION_FAILURE shall remain in effect until all reference writes have been completed successfully. How a particular implementation handles other failures during writing to referenced objects shall be a local matter except that Reliability shall indicate a value other than NO_FAULT_DETECTED.

The order of evaluation of references and any referenced object write delay shall be a local matter.

12.62.5 Present_Stage

This property, of type Unsigned, shall indicate the array index (1 to $\mathbf{N}_{\mathbf{stages}}$) that corresponds to the current active stage or 0 meaning that the Present_Stage has not yet been initialized. Upon device restart, or when the Stages property is written to any of its elements, or the size of the Stages array changes, Present_Stage shall be set to 0 temporarily and then Present_Value shall be reevaluated as described in Clause 12.62.4.1.

© ISO 2022 – All rights reserved

Attempts to read Present_Stage when it is internally set to 0 shall return a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_NOT_INITIALIZED.

12.62.6 Stages

This property, of type BACnetARRAY of BACnetStageLimitValue, is an array representing the stages by limit, desired present values for the objects referenced by Target_References, and deadband. The size of the array is $\mathbf{N}_{stages}$, where $\mathbf{N}_{stages}$ shall be greater than 1. BACnetStageLimitValue is a tuple consisting of the following fields:

Limit	REAL
Values	BIT STRING
Deadband	REAL

The 'Limit' values for all elements shall be strictly ascending, such that:

```
for Index=1 to Nstages-1
{
    lowerbound = (Stages[Index].Limit + Stages[Index].Deadband)
    upperbound = (Stages[Index+1].Limit - Stages[Index+1].Deadband)
    lowerbound <= upperbound
}
```

If any of the stages do not meet this criterion, then the Reliability property shall have a value of CONFIGURATION_ERROR.

The bits in 'Values' correspond to references in the Target_References array, such that bit (Index = 0 to $\mathbf{N}_{references}$ - 1) corresponds to Target_References[Index + 1], etc. The length of the 'Values' bitstring shall be $\mathbf{N}_{references}$ bits.

The 'Deadband' shall be zero or positive. A negative value for 'Deadband' shall cause the value of Reliability to be CONFIGURATION_ERROR.

If the size of the Stages array is increased, then the new array elements, for which no initial value is provided, shall be initialized to contain 'Limit' = 0.0, 'Deadband' = 0.0, and 'Values' = {0…0}, and the value of Reliability shall be set to CONFIGURATION_ERROR.

If the size of the Stages array is less than 2, then the Reliability property shall have a value of CONFIGURATION_ERROR.

If Reliability has the value CONFIGURATION_ERROR, then Present_Value shall be set to Min_Pres_Value and Present_Stage to 1.

If Stages[$\mathbf{N}_{stages}$].Limit becomes smaller than Present_Value, then Present_Value shall be set to Stages[$\mathbf{N}_{stages}$].Limit.

If the Stages property is written, the value of Present_Stage shall be reevaluated and corresponding writes of new values to Target_References objects shall be triggered.

If the size of this array is changed, the size of the Stage_Names array shall also be changed to the same size.

12.62.7 Stage_Names

This property, of type BACnetARRAY of CharacterString, is an array representing a name for each stage. The number of array elements in Stage_Names shall be the same as the number of array elements in the Stages property. Stage_Names[1] shall correspond to the name for Present_Stage=1. Stage_Names[2] shall correspond to the name for Present_Stage=2, etc.

If the size of this array is changed, the size of the Stages array shall also be changed to the same size. If the size of Stage_Names is increased, then it shall be a local matter what the uninitialized array elements contain.

12.62.8 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Staging object. Three of the flags are associated with the values of other properties of this object. A more detailed

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE(0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.62.9 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.62.10 Reliability

This property, of type BACnetReliability, provides an indication that the properties of the Staging object are in a consistent state and that Target_References are being reliably written. See Clauses 12.62.4.1, 12.62.6, and 12.62.17.

Table 12-81 summarizes scenarios when Reliability has a value other than NO_FAULT_DETECTED.

Table 12-81. Reliability Scenarios

Scenario	Reliability Value	See Clause
Communication error when writing	COMMUNICATION_FAILURE	12.62.4.1
Non-communication-related failure when writing	Local Matter	12.62.4.1
'Limit' values out of order	CONFIGURATION_ERROR	12.62.6
'Deadband' is negative	CONFIGURATION_ERROR	12.62.6
Stages array size is increased	CONFIGURATION_ERROR	12.62.6
Min_Pres_Value is >= (Stages[1].Limit - Stages[1].Deadband)	CONFIGURATION_ERROR	12.62.17

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.62.11 Out_Of_Service

This property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

(a) changes to the Present_Value property are decoupled from the Target_References. This means that the objects referenced by Target_References shall not be updated;
(b) the Present_Value property and the Reliability property, if capable of taking on values other than NO_FAULT_DETECTED, shall be writable to allow simulating specific conditions or for testing purposes;
(c) other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties, as if those changes had occurred while the object was in service;

© ISO 2022 – All rights reserved

Restrictions on writing to the Present_Value property by software local to the BACnet device do not apply to local human-machine interfaces.

When Out_Of_Service becomes FALSE, it shall trigger updating of referenced objects as described in Clause 12.62.4.1.

12.62.12 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.62.13 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of this object. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.62.14 Target_References

This property, of type BACnetARRAY of BACnetDeviceObjectReference, is an array representing references to Binary Output, Binary Value, or Binary Lighting Output objects. The size of the array is $\mathbf{N}_{\mathbf{references}}$. The Target_References array elements [Index = 1 to $\mathbf{N}_{\mathbf{references}}$] shall correspond to the bit(Index - 1) of all 'Values' bitstrings of the Stages property.

If the property is restricted to referencing objects within the containing device, an attempt to write to a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with an 'Error Class' of PROPERTY and an 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED. Uninitialized Target_References array elements shall be given the instance number 4194303. A Target_References array element whose instance number is equal to 4194303 shall be considered uninitialized and shall be ignored in all operations that may use that reference.

If the size of this array is changed, the size of the 'Values' bitstrings of the Stages property shall also be changed to the same size. If the bitstring length is increased, added bits shall be cleared (0).

12.62.15 Priority_For_Writing

This property, of type Unsigned, defines the priority at which the referenced properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. The value 1 is considered the highest priority and 16 the lowest. See Clause 19.2.

12.62.16 Default_Present_Value

This property, of type REAL, defines the value to be used for Present_Value upon device restart. Upon restart Default_Present_Value shall be copied to Present_Value and cause the reevaluation of Present_Value as described in Clause 12.62.4.

12.62.17 Min_Pres_Value

This property, of type REAL, represents the minimum value for Present_Value. Min_Pres_Value shall always be strictly less than the quantity (Stages[1].Limit - Stages[1].Deadband).

If a change to Min_Pres_Value, Stages[1].Limit, or Stages[1].Deadband violates this rule, then the Reliability property shall have a value of CONFIGURATION_ERROR. If a change to Min_Pres_Value causes it to be greater than Present_Value, then Present_Value shall be written with Min_Pres_Value.

12.62.18 Max_Pres_Value

This read-only property, of type REAL, represents the maximum value for Present_Value. This value shall be defined as the 'Limit' of the last entry of the Stages property array (i.e., Stages[$\mathbf{N}_{\mathbf{stages}}$].Limit).

If the size of the Stages array is zero, then Max_Pres_Value shall be equal to Min_Pres_Value.

12.62.19 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.62.20 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.62.21 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.62.22 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.62.23 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.62.24 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.62.25 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.62.26 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.62.27 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.62.28 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.62.29 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.62.30 Value_Source

This property, of type BACnetValueSource, indicates the source of the value of the Present_Value. The Value_Source property and its use in the value source mechanism are described in Clause 19.5.

12.62.31 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.62.32 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.62.33 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.62.34 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.62.35 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.63 Audit Reporter Object Type

The Audit Reporter object type defines a standardized object whose properties represent the externally visible audit settings for a BACnet device which reports auditable actions. BACnet devices which report auditable actions shall contain at least one Audit Reporter object. A detailed description of BACnet audit reporting can be found in Clause 19.6.

Audit Reporter objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Audit Reporter objects that support intrinsic reporting shall apply the NONE event algorithm.

The Audit Reporter object and its properties are summarized in Table 12-82 and described in detail in this clause.

Table 12-82. Properties of the Audit Reporter Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Event_State	BACnetEventState	R
Audit_Level	BACnetAuditLevel	R
Audit_Source_Reporter	BOOLEAN	R
Auditable_Operations	BACnetAuditOperationFlags	R
Audit_Priority_Filter	BACnetPriorityFilter	R
Issue_Confirmed_Notifications	BOOLEAN	R
Monitored_Objects	BACnetARRAY[N] of BACnetObjectSelector	O
Maximum_Send_Delay	Unsigned	O[1,2]
Send_Now	BOOLEAN	O[1,2]
Event_Detection_Enable	BOOLEAN	O[3,5]
Notification_Class	Unsigned	O[3,5]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4,5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

1 If present, these properties shall be writable.
2 If the object supports delaying of audit notifications, then these properties shall be present.
3 These properties are required if the object supports intrinsic reporting.
4 This property, if present, is required to be read-only.
5 These properties shall be present only if the object supports intrinsic reporting.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

12.63.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.63.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

© ISO 2022 – All rights reserved

12.63.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be AUDIT_REPORTER.

12.63.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.63.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Audit Reporter object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.63.6 Reliability

This property, of type BACnetReliability, provides an indication that the properties of the Audit Reporter object are in a consistent state and whether the object is operating properly.

If this Audit Reporter object is enabled and configured to monitor one or more objects that another enabled Audit Reporter object is also configured to monitor, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Reporter object is enabled and the device is unable to resolve the Audit_Notification_Recipient device, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Reporter object fails to successfully send audit notifications, or fails to receive acknowledgements when using ConfirmedAuditNotification requests, the Reliability property shall be set to COMMUNICATION_FAILURE. The existence of this reliability condition shall not stop the object from attempting to send audit notifications. It is a local matter whether or not the object will retry sending of audit notifications.

12.63.7 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.63.8 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for all objects this Audit Reporter object reports for which do not have an Audit_Level property, or for which the Audit_Level property is set to DEFAULT.

This property shall not have the value DEFAULT.

If not configurable, this property shall not have the value NONE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

For details on auditing and the use of this property, see Clause 19.6.2.

12.63.9 Audit_Source_Reporter

This read-only property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) this audit reporter object is the reporter object which controls operation source audit reporting.

If this property is TRUE, this object generates notifications for auditable operations initiated by this device.

For details on auditing and the use of this property, see Clause 19.6.2.

12.63.10 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the object will report.

If not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

For details on auditing and the use of this property, see Clause 19.6.3.

12.63.11 Audit_Priority_Filter

This property, of type BACnetPriorityFilter, specifies the auditable command priorities for write operations on the commandable property of the objects for which this object performs audit reporting. Bits which are set indicate the priorities for which audit notifications will be generated. The bit number is 1 less than the priority it is associated with (bit 7 is associated with priority 8).

For details on auditing and the use of this property, see Clause 19.6.3.

12.63.12 Issue_Confirmed_Notifications

This property, of type BOOLEAN, shall convey whether confirmed (TRUE) or unconfirmed (FALSE) audit notifications shall be issued.

12.63.13 Monitored_Objects

This property, of type BACnetARRAY of BACnetObjectSelector, is used to indicate the objects that the Audit Reporter object will generate notifications for.

If the array is of length 0, or all entries in the array are NULL entries, then the Audit Reporter object does not generate notifications for any objects in the device.

Entries of type BACnetObjectIdentifier indicate specific objects that are reportable by the Audit Reporter object. Entries of type BACnetObjectType indicate that all objects of the specified type are reportable by the Audit Reporter object. Entries of type NULL are ignored.

If there are multiple Audit Reporter objects which are configured to report for a specific object, only the one with the lowest object instance shall generate notifications for the object. This allows a generic catch-all Audit Reporter object to be created with a high instance number and lower instance number Audit Reporter objects which provide customized audit reporting for specific objects.

12.63.14 Maximum_Send_Delay

This property, of type Unsigned, specifies the maximum amount of time, in seconds, that the Audit Reporter object will delay sending audit notifications for the purpose of batch sending notifications.

The value in this property does not mandate that notifications shall be delayed, but rather allows the object to delay sending of notifications by up to the specified number of seconds.

This property shall support all values in the range 0 to 3600.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.63.15 Send_Now

This property, of type BOOLEAN, when written to TRUE, forces the Audit Reporter object to send any and all audit notifications that are being delayed. After all delayed audit notifications are sent, the Audit Reporter object resets the property to FALSE.

It is a local matter whether or not the Audit Reporter object stops attempting to immediately send its delayed notifications if the value is written to FALSE.

12.63.16 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.63.17 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.63.18 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.63.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.63.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.63.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.63.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.63.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.63.24 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.63.25 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.63.26 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.63.27 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.63.28 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.64 Audit Log Object Type

An Audit Log object combines audit notifications from operation sources and operation targets and stores the combined record in an internal buffer for subsequent retrieval. Each timestamped buffer entry is called an audit log "record."

Each Audit Log object maintains an internal, persistent, optionally fixed-size log buffer. This log buffer fills or grows as audit log records are added. If the log buffer becomes full, the least recent log records are overwritten when new log records are added. Log buffers are transferred as a list of BACnetAuditLogRecord values using the ReadRange and AuditLogQuery services. Each log record in the log buffer has an implied sequence number that is equal to the value of the Total_Record_Count property immediately after the record is added. See Clause 19.6 for a full description of how audit notifications are added to audit logs.

As records are added into the log, the Audit Log object will scan existing entries for a matching record. A record is a match if:

(a) the record contains the timestamp for the opposite actor (the record contains the operation source timestamp when merging in an operation target notification and vice-versa);
(b) the operation-source, operation, invoke-id, target-device, target-property, are all equal;
(c) if the user-id, user-role, target-value fields are provided in both notifications then they are equal; and
(d) if the source-timestamp and target-timestamp values are approximately equal (+/- APDU_Timeout * 2).

If a match is found, the existing log record is updated. Otherwise, a new record is created. If a match is found, and it already contains both an operation source and an operation target portion, then the notification is dropped. When creating a new record, those fields which are not supplied in the notification (such as the 'Source Timestamp' when a server notification is received) shall be absent from the record. When updating an existing record, those fields not supplied in the original notification are updated from the new notification, if present. For the 'Current Value' field, a value provided by the operation target device shall always take precedence over a value provided by an operation source device. As such, if the values provided in the peer notifications differ, the operation target value shall be the one used in the record.

Logging may be enabled and disabled through the Enable property. Audit Log enabling and disabling is recorded in the audit log buffer.

Unlike other log objects, Audit Log objects do not use the BUFFER_READY event algorithm.

The acquisition of log records by remote devices has no effect upon the state of the Audit Log object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its log records at any time.

Audit Log objects may optionally support forwarding of audit notifications to “parent” audit logs. This functionality improves the reliability of the audit system by allowing intermediaries to buffer audit notifications in the case where the ultimate audit logger is offline for a short period of time. It is expected that intermediaries be capable of storing a larger number of records than devices which report auditable actions. It is also useful for buffering of audit notifications so they can be sent in bulk to the parent audit log. When operating in this mode, with the Delete_On_Forward property set to TRUE, the object is not required to perform audit notification matching and combining.

Audit Log objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Audit Log objects that support intrinsic reporting shall apply the NONE event algorithm.

The Audit Log object and its properties are summarized in Table 12-83 and described in detail in this subclause.

Table 12-83. Properties of the Audit Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property Identifier	Property Datatype	Conformance Code
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Buffer_Size	Unsigned32	R
Log_Buffer	BACnetLIST of BACnetAuditLogRecord	R
Record_Count	Unsigned64	R
Total_Record_Count	Unsigned64	R
Member_Of	BACnetDeviceObjectReference	O[1]
Delete_On_Forward	BOOLEAN	O[1,2]
Issue_Confirmed_Notifications	BOOLEAN	O[1]
Event_Detection_Enable	BOOLEAN	O[3,4]
Notification_Class	Unsigned	O[3,4]
Event_Enable	BACnetEventTransitionBits	O[3,4]
Acked_Transitions	BACnetEventTransitionBits	O[3,4]
Notify_Type	BACnetNotifyType	O[3,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4,5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[6]
Audit_Level	BACnetAuditLevel	O[7]
Auditable_Operations	BACnetAuditOperationFlags	O[7]
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

1 These properties shall be present if the object supports forwarding.
2 This property shall be read-only and TRUE if the object does not support matching and combining of records.
3 These properties are required if the object supports intrinsic reporting.
4 These properties shall be present only if the object supports intrinsic reporting.
5 This property, if present, is required to be read-only.
6 If this property is present, then the Reliability property shall be present.
7 This property shall be present only if the device supports audit reporting.

12.64.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.64.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.64.3 Object_Type

This read-only property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be AUDIT_LOG.

12.64.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.64.5 Status_Flags

This read-only property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Audit Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be Logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.64.6 Event_State

This read-only property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.64.7 Reliability

This property, of type BACnetReliability, provides an indication that the properties of the Audit Log object are in a consistent state and whether the object is operating properly.

If the Audit Log object is enabled and configured to forward audit notifications, and the device is unable to resolve the Member_Of device, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Log object, when configured to forward audit notifications, fails to successfully send audit notifications, or fails to receive acknowledgements when using ConfirmedAuditNotification requests, the Reliability property shall be set to COMMUNICATION_FAILURE. The existence of this reliability condition shall not stop the object from attempting to send audit notifications. It is a local matter whether or not the object will retry sending of audit notifications.

12.64.8 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging is enabled. Logging occurs if and only if Enable is TRUE. Log_Buffer records of type log-status are recorded without regard to the value of the Enable property.

12.64.9 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

12.64.10 Log_Buffer

This property, of type BACnetLIST of BACnetAuditLogRecord, is a list of timestamped log records of datatype BACnetAuditLogRecord, each of which conveys the audit notification parameters or status changes in the Audit Log object. Each log record has data fields as follows:

Timestamp	The local date and time that the entry was placed into the audit log.
Log Datum	The audit notification information, or a change in status or operation of the Audit Log object itself.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The choices available for the 'Log Datum' are listed below:

log-status	This choice represents a change in the status or operation of the Audit Log object. Whenever one of the events represented by the flags listed below occurs, a log record shall be appended to the buffer.	
	LOG_DISABLED	This flag is changed whenever collection of log records by the Audit Log object is enabled or disabled. It shall be TRUE if Enable is FALSE; otherwise it shall be FALSE.
	BUFFER_PURGED	This flag shall be set to TRUE whenever the log buffer is cleared. After this value is recorded in the log buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the log buffer shall be placed into the log buffer even if logging is disabled, the local time is outside of the time range defined by the Start_Time and Stop_Time properties, or the log buffer was completely empty before the request for clearing.
	LOG_INTERRUPTED	This flag indicates that the collection of log records by the Audit Log object was interrupted by a power failure, device reset, object reconfiguration, or other such disruption, such that samples prior to this log record might have been missed.
audit-notification	This choice represents an audit notification that was received. It consists of the audit information from the operation source and operation target generated ConfirmedAuditNotification and/or UnconfirmedAuditNotification requests. If the audit notification was generated locally, this choice shall hold what would be received if the Audit Log object existed on a remote device.	
time-change	This choice represents a change in the clock setting in the device, records the number of seconds and fraction of a second by which the clock changed. If, and only if, the number is not known, such as when the clock is initialized for the first time, the value recorded shall be 0.0. This log record shall be recorded after changing the local time of the device, and the timestamp shall reflect the new local time of the device.	

Also associated with each log record is an implied sequence number, the value of which is equal to Total_Record_Count at the point where the log record has been added into the log buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the Audit Log object is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one. While no interoperable method is provided for resetting the Audit Log object, catastrophic failures can result in replacement of the logging device which will be equivalent to resetting of the object. It is for situations such as this that clients need to be prepared.

The log buffer is not network accessible except through the use of the ReadRange and AuditLogQuery services in order to avoid problems with record sequencing when segmentation is required.

12.64.11 Record_Count

This read-only property, of type Unsigned64, shall represent the number of log records currently resident in the log buffer. Unlike other log object types, this property is not writable as audit logs are never expected to be reset.

12.64.12 Total_Record_Count

This property, of type Unsigned64, shall represent the total number of log records collected by the Audit Log object since creation. When the value of Total_Record_Count reaches its maximum possible value of 264 - 1, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.64.13 Member_Of

This property, of type BACnetDeviceObjectReference, shall indicate the device which this Audit Log object forwards all audit log records to.

All log records collected by the Audit Log shall be sent to the referenced device via audit notification requests.

12.64.14 Delete_On_Forward

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the notifications will be removed from the log buffer after being successfully forwarded.

12.64.15 Issue_Confirmed_Notifications

This property, of type BOOLEAN, shall convey whether confirmed (TRUE) or unconfirmed (FALSE) audit notifications shall be issued when forwarding audit notifications.

12.64.16 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.64.17 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.64.18 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.64.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.64.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.64.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type BACnetDateTime shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.64.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

12.64.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.64.24 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.64.25 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.64.26 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.6.3.

If this property is present and not configurable, the value shall not be NONE.

12.64.27 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.6.3.

12.64.28 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tags and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.64.29 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.64.30 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13 ALARM AND EVENT SERVICES

This clause describes the conceptual approach and application services used in BACnet to manage communication related to events. Object types relating to event management are defined in Clause 12. In general, "events" are changes of value of certain properties of certain objects, or internal status changes, that meet predetermined criteria. There are two mechanisms provided in BACnet for managing events: change of value reporting and event reporting.

Change of value (COV) reporting allows a COV-client to subscribe with a COV-server, on a permanent or temporary basis, to receive reports of some changes of value of some referenced property based on fixed criteria. Certain BACnet standard objects may optionally support COV reporting. If a standard object provides COV reporting, then changes of value of specific properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to one or more subscriber clients. Typically, COV notifications are sent to supervisory programs in COV-client devices or to operators or logging devices. Proprietary objects may support COV reporting at the implementor's option.

Event reporting allows BACnet devices to contain one or more event objects (event-initiating objects) that generate event notifications that may be directed to one or more destinations. Event reporting comes in three forms: intrinsic reporting, algorithmic reporting and alert reporting. Typically, event notifications are sent to operators or logging devices represented by "processes" within a notification-client device.

13.1 Change of Value Reporting

Change of value (COV) reporting allows a COV-client to subscribe with a COV-server, on a permanent or temporary basis, to receive reports of some changes of value of some referenced property based on fixed criteria. If an object provides COV reporting, then changes of value of any subscribed-to properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to subscribing clients. Typically, COV notifications are sent to supervisory programs in COV-client devices or to operators or logging devices. Any object, proprietary or standard, may support COV reporting at the implementor's option.

COV subscriptions are established using the SubscribeCOV service, the SubscribeCOVProperty service, or the SubscribeCOVPropertyMultiple service. The subscription establishes a connection between the change of value detection and reporting mechanism within the COV-server device and a "process" within the COV-client device. Notifications of changes are issued by the COV-server when changes occur after the subscription has been established. The ConfirmedCOVNotification and UnconfirmedCOVNotification services, or the ConfirmedCOVNotificationMultiple and UnconfirmedCOVNotificationMultiple services, are used by the COV-server to convey change notifications. The choice of confirmed or unconfirmed service is specified in the subscription.

When a BACnet standard object, of a type listed in Table 13-1, supports COV reporting it shall support COV reporting for the property as listed in Table 13-1. At the implementor's discretion, COV reporting may also be supported for any other property of the object. For properties listed in Table 13-1 that represent a numeric quantity, the COV increment used to determine when to generate notifications will be the COV_Increment property of the object unless a COV_Increment parameter is supplied in the SubscribeCOVProperty or SubscribeCOVPropertyMultiple service. For other properties that represent a numeric quantity, the COV increment to use when not supplied with the SubscribeCOVProperty or SubscribeCOVPropertyMultiple service shall be a local matter. This is to allow multiple subscribers that do not require a specific increment to use a common increment to allow for the reduction of the processing burden on the COV-server. The criteria for COV reporting for properties other than those listed in Table 13-1 is based on the datatype of the property subscribed to and is described in Table 13-1a.

If an object supports the COV_Period property and COV_Period is non-zero, it shall issue COV notifications to all subscribed recipients at the regular interval specified by COV_Period, in addition to the notifications initiated by the change of value of the monitored property. The value of the monitored property conveyed by the periodic COV notification shall be the basis for determining whether a subsequent COV notification is required by the change in value of the monitored property. If COV_Period is zero, the periodic notifications shall not be issued.

It is the responsibility of the COV-server to maintain the list of active subscriptions for each object that supports COV notification. This list of subscriptions shall be capable of holding at least a single subscription for each object that supports COV notification, although multiple subscriptions may be supported at the implementor's option. The list of subscriptions is network-visible through the Device object's Active_COV_Subscriptions property. Subscriptions may be created with finite lifetimes, meaning that the subscription may lapse and be automatically canceled after a period of time. Optionally with SubscribeCOV, the lifetime may be specified as infinite, meaning that no automatic cancellation

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

occurs. However, the COV-server is not required to guarantee preservation of subscriptions across power failures or "restarts." Periodic resubscription is allowed and expected and shall simply succeed as if the subscription were new, extending the lifetime of the subscription.

For COV-servers that support SubscribeCOVPropertyMultiple, it is the responsibility of the COV-server to maintain the list of active COV-multiple contexts. A COV-multiple context shall hold all COV-multiple subscription specifications for a COV-client and form of notification, established through one or multiple SubscribeCOVPropertyMultiple requests from that COV-client. This list of COV-multiple contexts shall be capable of holding at least a single subscription for each object that supports COV-multiple notification, although multiple subscriptions may be supported at the implementer's option. The list of COV-multiple contexts is network-visible through the Device object's Active_COV_Multiple_Subscriptions property. Subscriptions are created with finite lifetimes, meaning that the subscription will lapse and be automatically canceled after a period of time. The COV-server is not required to guarantee preservation of COV-multiple subscriptions across power failures or "restarts." Periodic resubscription is expected and shall simply succeed as if the subscription were new, extending the lifetime of the subscription. For simplified re-subscription, the COV-client may resubscribe without being required to provide all COV notification specifications again in the SubscribeCOVPropertyMultiple service request for re-subscription. See Clause 13.16.

For COV-multiple subscriptions that request notification of timestamped changes, the COV-server may queue up such timestamped changes and initiate COV-multiple notifications later. When queueing changes, the server shall not delay the sending of notifications longer than 'Max Notification Delay' after the earliest timestamped change in the queue. The notification conveys all timestamped changes to subscribed-to properties since the last COV-multiple notification. Multiple notification messages may be required to convey all changes. If a change occurs to a property that is subscribed to without timestamps in the same COV-multiple context, a notification is initiated immediately and all changes that are currently queued up for timestamped notification shall be sent together with the change that is not timestamped.

The different standard objects that support standardized COV reporting use different criteria for determining that a "change of value" has occurred, which are summarized in Table 13-1. Proprietary object types, or other standard object types not listed in Table 13-1, that support COV reporting of the Present_Value property, should follow these criteria whenever possible. Any objects that may optionally provide COV or COV-multiple support and the change of value algorithms they shall employ are summarized in Tables 13-1 and 13-1a.

Table 13-1. Standardized Objects That May Support COV Reporting

Object Type	Criteria	Properties Reported
Access Door	If Present_Value changes at all or Status_Flags changes at all or Door_Alarm_State changes at all (if the object has a Door_Alarm_State property)	Present_Value, Status_Flags, Door_Alarm_State (if the object has a Door_Alarm_State property)
Access Point [1]	If Access_Event_Time changes at all or Status_Flags changes at all	Access_Event, Status_Flags, Access_Event_Tag, Access_Event_Time, Access_Event_Credential, Access_Event_Authentication_Factor (if present)
Analog Input, Analog Output, Analog Value, Integer Value, Large Analog Value, Lighting Output, Positive Integer Value	If Present_Value changes by COV_Increment or Status_Flags changes at all	Present_Value, Status_Flags

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Object Type	Criteria	Properties Reported
Binary Input, Binary Lighting Output, Binary Output, Binary Value, CharacterString Value, Date Value, Date Pattern Value, DateTime Value, DateTime Pattern Value, Life Safety Point, Life Safety Zone, Multi-state Input, Multi-state Output, Multi-state Value, OctetString Value, Time Value, Time Pattern Value	If Present_Value changes at all or Status_Flags changes at all	Present_Value, Status_Flags
Credential Data Input	If Update_Time changes at all or Status_Flags changes at all	Present_Value, Status_Flags, Update_Time
Load Control	If Present_Value, Status_Flags, Requested_Shed_Level, Start_Time, Shed_Duration, or Duty_Window changes at all	Present_Value, Status_Flags, Requested_Shed_Level, Start_Time, Shed_Duration, Duty_Window
Loop	If Present_Value changes by COV_Increment or Status_Flags changes at all	Present_Value, Status_Flags, Setpoint, Controlled_Variable_Value
Pulse Converter	If Present_Value changes by COV_Increment or Status_Flags changes at all or If COV_Period expires	Present_Value, Status_Flags, Update_Time
Staging	If Present_Value changes by COV_Increment or Status_Flags changes at all or Present_Stage changes at all	Present_Value, Status_Flags, Present_Stage

[1] For COV contexts for this object type in Active_COV_Subscriptions in the Device object, the Monitored Property Reference shall contain the Access_Event property identifier.

Table 13-1a. Criteria Used for COV Reporting of Properties Other Than Those Listed in Table 13-1.

Value	Criteria	Properties Reported
A value that represents a numeric quantity	If the value of the property changes by the increment (from the service if provided; otherwise, as determined by the device) or Status_Flags changes at all (if the object has a Status_Flags property)	The subscribed-to property, Status_Flags (if the object has a Status_Flags property)
All other values	If the property changes at all or Status_Flags changes at all (if the object has a Status_Flags property)	The subscribed-to property, Status_Flags (if the object has a Status_Flags property)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 13-1a-2. Criteria for COV Reporting for Properties with Specialized Criteria.

Property	Criteria	Properties Reported
Value_Source (for commandable properties)	If criteria for the COV reporting for the object are met (as per Table 13-1) or Value_Source changes[1] or Current_Command_Priority changes	The values listed for the object in Table 13-1 (if present in the table, otherwise Present_Value and Status_Flags), Value_Source, Last_Command_Time, Current_Command_Priority
Value_Source (for non-commandable properties)	If criteria for the COV reporting for the object are met (as per Table 13-1) or Value_Source changes[1]	The values listed for the object in Table 13-1 (if present in the table, otherwise Present_Value and Status_Flags), Value_Source

[1] See Clause 19.5.2 for further requirements on notifications related to Value_Source changes.

13.1.1 Unsubscribed COV Notifications

Some objects may share information by generating UnconfirmedCOVNotification messages without using COV subscriptions. As described in Clause 13.7, such notifications set the Subscriber Process Identifier parameter to zero to identify them as unsubscribed.

The use of UnconfirmedCOVNotification messages in this manner is not restricted, and any object can use this mechanism to distribute its properties' values to one or more recipients. The selection of which properties to send and the criteria for when to send them are a local matter. A single object is not restricted to sending a single set of properties and thus may use this mechanism for different purposes with different collections of properties to different recipients.

Some standardized objects have standardized usages of this mechanism, and those are listed in Table 13-1b. Inclusion in the table does not restrict other collections of properties from being sent for other purposes.

Table 13-1b. Standardized Objects That May Support Standardized Unsubscribed COV Reporting

Object Type	Distribution Controlled By	Criteria	Properties Reported
Device	Restart_Notification_Recipients	Device has completed the restart process. See Clause 19.3	System_Status, Time_Of_Device_Restart, Last_Restart_Reason
Global Group	COVU_Recipients	Periodic, as determined by COVU_Period	Member_Status_Flags[1], Elements of Present_Value[1]

[1] For Global Group, the elements of Present_Value shall be encoded individually each in its own BACnetPropertyValue production and shall include its array index. The elements shall be sent in index order and the first element shall be the Unsigned value at array index 0 (to inform recipients of the total array size). If the total Present_Value array is too large to fit within a single message, then multiple notifications shall be sent in order to convey all the elements. If a single element is too large to fit in a single message, it shall be encoded as an Error production with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_TOO_LONG. When multiple notifications are required, the index 0 element of Present_Value and the Member_Status_Flags shall be present only in the first notification.

13.1.2 Transmitting Unconfirmed COV Notification Requests

There are three circumstances under which a device may transmit an UnconfirmedCOVNotification or UnconfirmedCOVNotificationMultiple request:

(a) The device has an active COV or COV-multiple subscription that specifies 'Issue-Confirmed-Notifications' is FALSE, and a change of value occurs to the subscribed-to object or property or a notification delay elapses;
(b) The device is configured, as a local matter, to send unsubscribed UnconfirmedCOVNotification or UnconfirmedCOVNotificationMultiple requests;
(c) The device performs the device restart procedure as defined in Clause 19.3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

In the first case, since the subscriber device is known, the UnconfirmedCOVNotification or UnconfirmedCOVNotificationMultiple request shall be transmitted as a unicast to the subscriber device.

In the second case, the server may elect to deliver the unsubscribed UnconfirmedCOVNotification or UnconfirmedCOVNotificationMultiple request using a unicast or broadcast address as configured. The 'Process Identifier' parameter for requests being broadcast shall be zero.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.2 Event Reporting

Event reporting is used to detect and report conditions that are broadly categorized into one of three possible groups: fault, offnormal, and normal. A "fault" condition is a malfunction, nearly always representing a failure within the automation system itself. An "offnormal" condition is a condition within the system that is not normally expected or is outside the bounds of ideal operation. A "normal" condition is anything else.

Objects which support event reporting are called event-initiating objects. Event-initiating objects identify their "event state" from moment to moment as one of any number of possibly unique event states. Notifications are triggered by the "transition" of conditions for an object, usually from one unique event state to another. In these contexts, all states that are not normal and not fault are offnormal states, and transitions that result in an offnormal state are considered to be TO_OFFNORMAL transitions. Transitions to any fault state are considered to be TO_FAULT transitions. All other transitions are, by definition, TO_NORMAL transitions.

Intrinsic reporting consists of an object monitoring its own properties, whereas algorithmic reporting consists of an object monitoring properties of other objects. Certain BACnet standard objects may optionally support intrinsic reporting by supporting optional properties that define the type of event to be generated and options for handling and routing of the notifications. Proprietary objects may support intrinsic reporting at the implementor's option.

Algorithmic reporting allows the monitoring of objects that do not provide intrinsic reporting, or the monitoring of an object with an algorithm or algorithm parameters that differ from those configured in the object. Any of the standardized algorithms described in Clause 13.3 may be used to establish criteria for algorithmic reporting. Algorithmic reporting differs from intrinsic reporting in that Event Enrollment objects are used to determine the event condition(s).

Alert reporting allows any object to provide event reports that are unrelated to the object's event state and the intrinsic reporting algorithm of the object. Conceptually, when the need for an alert message is identified by an object, the alert is passed to an Alert Enrollment object for distribution. Alerts differ from intrinsic reporting and algorithmic reporting in that there are no standard conditions under which alerts are generated and in that alerts are stateless and cannot be acknowledged.

Events may be selectively identified as belonging to the category of "alarms" or "events." Event-initiating objects indicate this distinction through the Notify_Type property. Conceptually, alarms are events that are intended to be seen and reacted to by human operators. Operator workstation software is written, for example, to facilitate the reviewing and acknowledgment of alarms, possibly conveying the acknowledgment over the network via the AcknowledgeAlarm service. Events may or may not be of interest to operators and are typically intended for machine-to-machine communication. Applications of events include equipment interlocks, temperature overrides, the transition to a load-shedding condition, and so on. In the BACnet protocol, the singular distinction is that alarms will be reported by the GetAlarmSummary service, while all other events will not. In every other respect, BACnet makes no distinction between an alarm and an event.

The event notification services contain a 'Time Stamp' parameter that indicates the chronological order of events. This 'Time Stamp' may be the actual time as determined by the local device clock or, if the device has no clock, a sequence number. Sequence numbers are required to increase monotonically up to their maximum value, at which point the number "wraps around" to zero. A device may have a single sequence number for all event-initiating objects, or it may have a separate sequence number for each object.

Event notifications may be specified to use either confirmed or unconfirmed services for notification messages. By providing two kinds of notification mechanisms, BACnet allows the application designer to determine the relative importance of each event and whether or not notification of its occurrence is essential or merely desirable. In the former case, notification can be carried out with a confirmed service and repeated for as many recipients as required. In the latter case, an unconfirmed service using a broadcast or multicast address may be used.

Notification Class objects provide event classification and specify the destination devices for notification messages using BACnetRecipients. The recipients may be individual devices, groups of devices with a common multicast address, or all devices reachable by a broadcast address. If a broadcast is used, the scope may be limited to all devices on a single network or it may be extended to encompass all devices on a BACnet internetwork. The Notification Class object defines the priorities to be used in event notification messages, whether acknowledgment by an application process or a human operator is required, and at what time periods during the week given destinations are to be used.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet event reporting provides support for event acknowledgment whereby an operator indicates that the event transition has been reviewed. The need for acknowledgments is enabled or disabled by event class through the Notification Class object and by transition type (TO_OFFNORMAL, TO_FAULT, or TO_NORMAL).

13.2.1 Event Detection and Reporting Model

The BACnet event detection and reporting model as outlined in this clause applies to all BACnet objects, both standard and proprietary. The event algorithms are described in Clause 13.3.

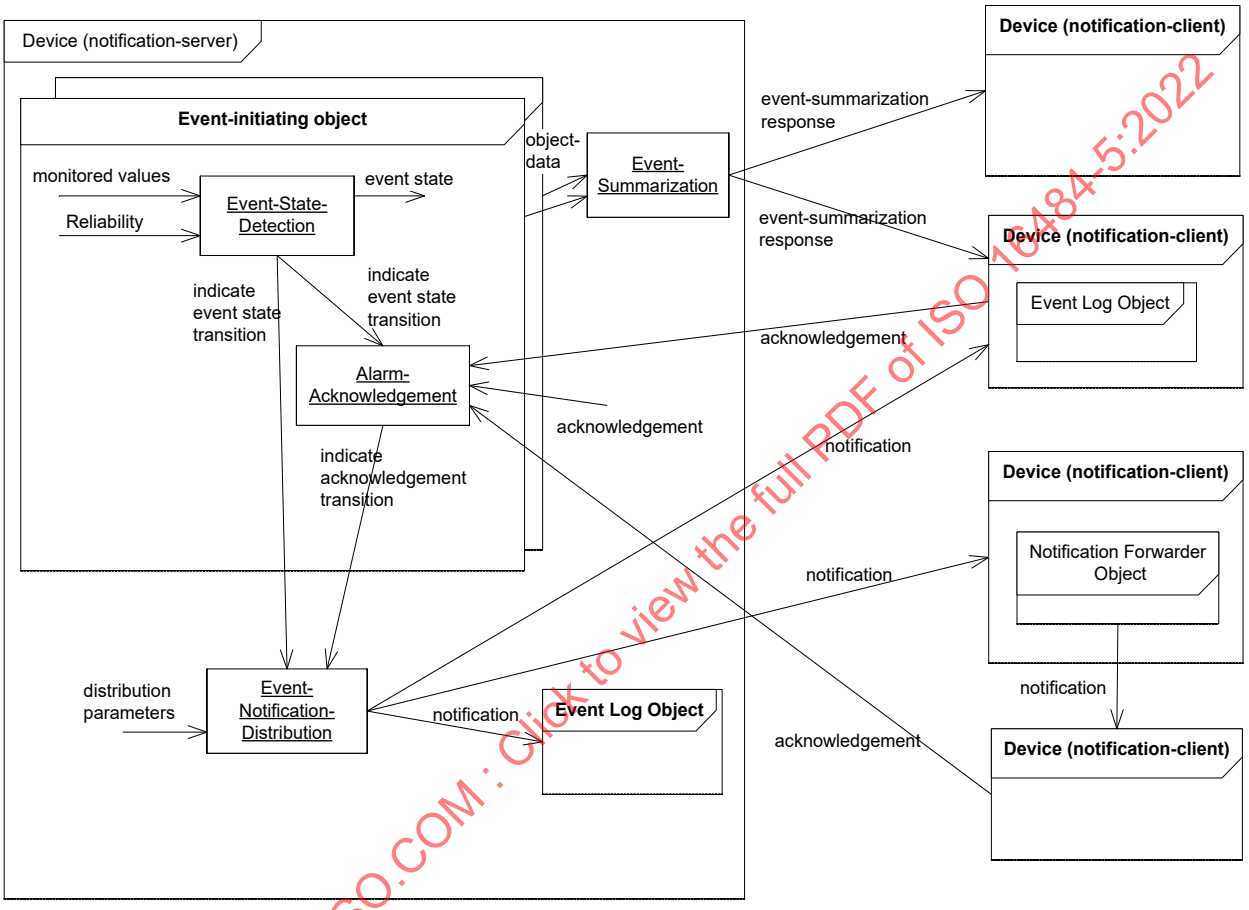


Figure 13-1. Overview of the Event Detection and Reporting Model

The event-detection and reporting model used in BACnet is separated into four main concepts: event-state-detection, alarm-acknowledgment, event-summarization, and event-notification-distribution. The flow of data between the main parts of the model and the roles that the different objects play in event reporting are shown in Figure 13-1.

Devices that contain event-initiating objects (notification-servers) interact with one or more devices which process the event information (notification-clients). A notification-client can receive the event information through an event notification or by querying the notification-server via an event-summarization service.

Event-state-detection consists of the monitoring of one or more values (including a Reliability value) in order to evaluate the object's event state.

An object that supports event-state-detection (intrinsic reporting or algorithmic reporting) may also support alarm-acknowledgment. Alarm-acknowledgment is an optional functionality whereby an event transition requires acknowledgment by an operator. A notification-client performs an acknowledgment by issuing an AcknowledgeAlarm service request. Additionally, the acknowledgment may be performed by means local to the device (e.g., an alarm reset button).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Event-summarization provides the ability for a device to retrieve event information independent of notifications, allowing notification-clients that have missed notifications or that are not subscribed for notifications to easily determine the event state of all objects in a device. A device is required to support event-summarization if it is capable of containing objects that support event-state-detection.

Event-notification-distribution refers to the process involved in sending notifications of event state transitions and acknowledgment transitions using ConfirmedEventNotifications and UnconfirmedEventNotifications to a set of notification-clients and to local Event Log objects. Event-notification-distribution is provided via Notification Class objects and, optionally, Notification Forwarder objects. Devices that support event-state-detection shall support event-notification-distribution.

Objects support event-state-detection via intrinsic reporting, or event-state-detection can be provided for the object via another object performing algorithmic reporting.

The following clauses specify the Event Detection and Reporting model independent of the object types. The Event Log and Notification Forwarder object processes notifications and are included in the overview for information. These objects and their associations to event notifications are specified in Clause 12.

13.2.2 Event-State-Detection

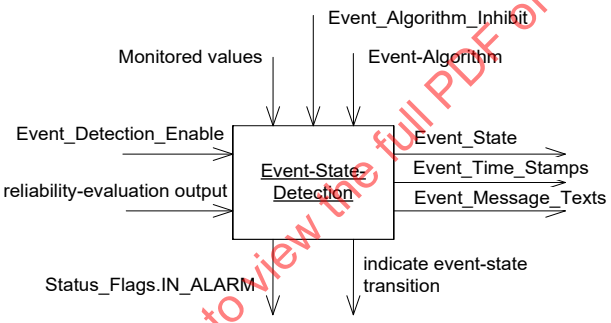


Figure 13-2. The Event-State-Detection Process

In the event-state-detection process, the event algorithm and the Reliability property together determine the event state of an object. The event algorithm determines the normal or offnormal states and the Reliability property determines whether or not the event state will indicate a fault. Fault detection takes precedence over the detection of normal and offnormal states. As such, when Reliability has a value other than NO_FAULT_DETECTED, the event-state-detection process will determine the object's event state to be FAULT.

When the event-state-detection process is disabled via the Event_Detection_Enable, both the event algorithm and the Reliability value are ignored, and Event_State remains NORMAL.

When the monitoring of values is disabled by Event_Algorithm_Inhibit, the result of the event algorithm will be ignored and all TO_OFFNORMAL and TO_NORMAL transitions will be disabled with the exception of TO_NORMAL transitions from FAULT. Event_Algorithm_Inhibit does not impact transitions to or from the fault state.

For intrinsic reporting in standard object types, the event algorithm is implied by the object type. For algorithmic reporting, the Event Enrollment object contains the Event_Type property which indicates the event algorithm. An object, standard or proprietary, shall use only a single event algorithm. For objects in which the event algorithm is configurable, there is an expectation that the event algorithm does not change dynamically.

In objects that support event-state-detection, the reporting of changes in Reliability cannot be disabled while event-state-detection is enabled. Instead, the object may be stopped from detecting faults through the Reliability_Evaluation_Inhibit property. If an object that supports event-state-detection does not have a Reliability property, then the reliability-evaluation is assumed to indicate NO_FAULT_DETECTED and no fault transitions shall occur.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A transition of the event state is indicated to the Alarm-Acknowledgment process (see Clause 13.2.3) and the event-notification-distribution process (see Clause 13.2.5).

13.2.2.1 Event-State-Detection State Machine

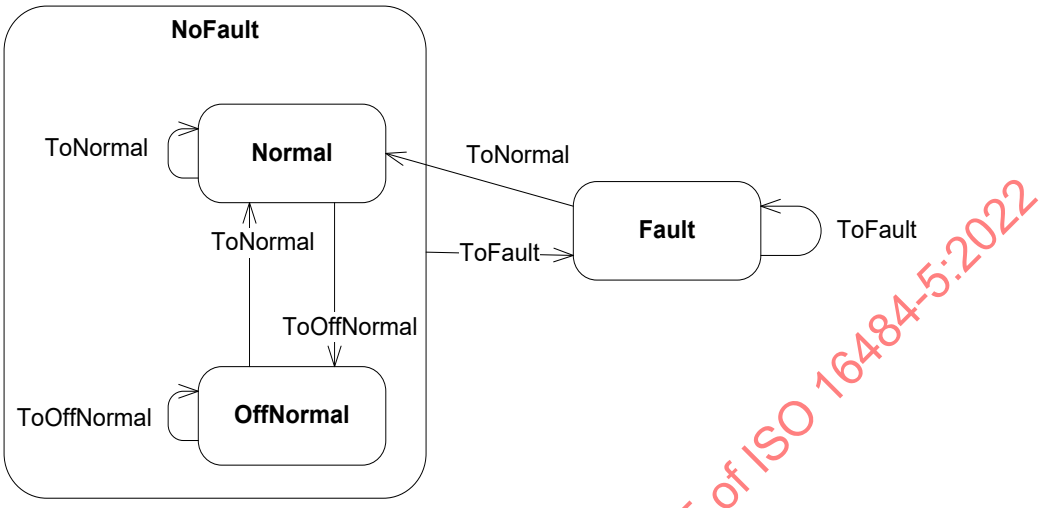


Figure 13-3. Event-State-Detection State Machine

The state machine in Figure 13-3 shows all possible transitions of the general event-state-detection model. Not all ToNormal or ToOffNormal transitions are supported by every event algorithm.

If the Event_Detection_Enable property is FALSE, then this state machine is not evaluated. In this case, no transitions shall occur, Event_State shall be set to NORMAL, and Event_Time_Stamps, Event_Message_Texts and Acked_Transitions shall be set to their respective initial conditions.

It is important to note that, in the following clauses, transitions from one Event_State to the same Event_State value are indicated by the event algorithm or by reliability-evaluation.

13.2.2.1.1 Normal

In the Normal state reliability-evaluation indicates a value of NO_FAULT_DETECTED and either the event algorithm indicates a normal event state or the Event_Algorithm_Inhibit is TRUE.

ToOffNormal
: If reliability-evaluation indicates a value of NO_FAULT_DETECTED and the event algorithm indicates an offnormal event state and Event_Algorithm_Inhibit is FALSE,

  then perform the corresponding transition actions (see Clause 13.2.2.1.4) and enter the OffNormal state.

ToFault
: If reliability-evaluation indicates a value other than NO_FAULT_DETECTED,

  then perform the corresponding transition actions and enter the Fault state.

ToNormal
: If reliability-evaluation indicates a value of NO_FAULT_DETECTED and the event algorithm indicates a transition to the Normal state and Event_Algorithm_Inhibit is FALSE,

  then perform the corresponding transition actions and re-enter the Normal state.

13.2.2.1.2 OffNormal

In the OffNormal state, reliability-evaluation indicates a value of NO_FAULT_DETECTED and the event algorithm indicates an offnormal event state and Event_Algorithm_Inhibit is FALSE. (Note that the OffNormal state includes all event states other than NORMAL and FAULT).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ToOffNormal
If reliability-evaluation indicates a value of NO_FAULT_DETECTED and the event algorithm indicates a transition to the OffNormal state and Event_Algorithm_Inhibit is FALSE,

then perform the corresponding transition actions and re-enter the OffNormal state.

ToFault
If reliability-evaluation indicates a value other than NO_FAULT_DETECTED,

then perform the corresponding transition actions and enter the Fault state.

ToNormal
If reliability-evaluation indicates a value of NO_FAULT_DETECTED and the event algorithm indicates a normal event state,
or
if reliability-evaluation indicates a value of NO_FAULT_DETECTED and Event_Algorithm_Inhibit is TRUE,

then perform the corresponding transition actions and enter the Normal state.

13.2.2.1.3 Fault

In the Fault state reliability-evaluation indicates a value other than NO_FAULT_DETECTED.

ToNormal
If reliability-evaluation indicates a value of NO_FAULT_DETECTED,

then perform the corresponding transition actions and enter the Normal state.

ToFault
If reliability-evaluation indicates a different Reliability value and the new Reliability value is not NO_FAULT_DETECTED or reliability-evaluation indicates a transition to the Fault state with the same Reliability value,

then perform the corresponding transition actions and re-enter the Fault state.

13.2.2.1.4 Transition Actions

This clause describes the actions to be taken when a transition of the event-state-detection state machine occurs. The actions are the same for all transitions and they shall be executed even if the transition does not change the event state (e.g., a transition from the OFFNORMAL event state to the OFFNORMAL event state).

Store the new event state in the event-initiating object's Event_State property. Note that the Event_State property shall reflect the specific BACnetEventState returned by the event algorithm (i.e. it is not acceptable to set Event_State to OFFNORMAL when the returned value is HIGH_LIMIT).

Store the time of the transition in the corresponding entry of the Event_Time_Stamps property.

Store the message text that is generated for distribution with the notification in the corresponding entry of the Event_Message_Texts property, if present.

Indicate the transition to the Alarm-Acknowledgment process (see Clause 13.2.3) and the event-notification-distribution process (see Clause 13.2.5).

13.2.2.1.5 Inhibiting Detection of Offnormal Conditions

The Event_Algorithm_Inhibit property temporarily overrides the event algorithm thus maintaining a normal Event_State regardless of the existence of offnormal conditions. The effect of this property on the Event_State property is shown in Figure 13-4.

Upon Event_Algorithm_Inhibit changing to TRUE, the event shall transition to the NORMAL state if not already there. While Event_Algorithm_Inhibit remains TRUE, no transitions shall occur except those into and out of FAULT. Upon

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Event_Algorithm_Inhibit changing to FALSE, any condition shall hold for its regular time delay after the change to FALSE before a transition is generated.

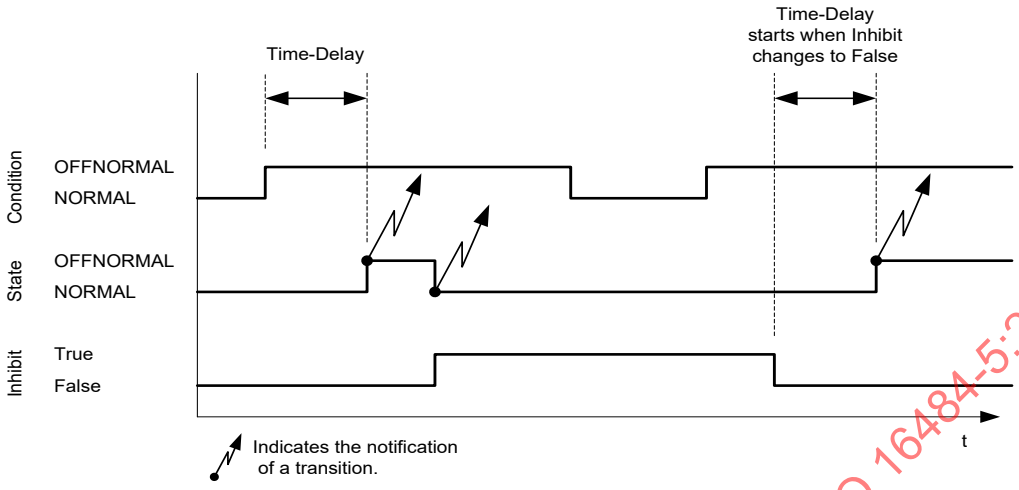


Figure 13-4. Effect of Event_Algorithm_Inhibit on Event_State

13.2.3 Alarm-Acknowledgment

The alarm-acknowledgment process, shown in Figure 13-5, is responsible for maintaining the acknowledgment state for each of the transition types (TO_NORMAL, TO_FAULT, TO_OFFNORMAL) in the Acked_Transitions property and for indicating acknowledgment transitions to the event-notification-distribution process.

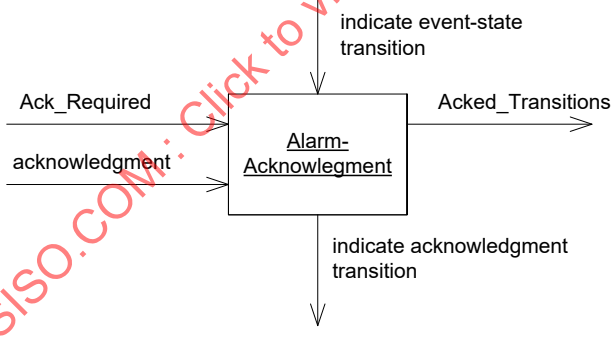


Figure 13-5. Alarm-Acknowledgment

Event state transitions are received from the event-state-detection state machine. Acknowledgment indications are received from the AcknowledgeAlarm service procedure and from a means local to the device (e.g., reset button) and acknowledgment transitions are indicated to the event-notification-distribution process (see Clause 13.2.5). Every device that supports acknowledgeable transitions shall support execution of the AcknowledgeAlarm service.

Whether or not an acknowledgment is required is determined by the Ack_Required property from the referenced Notification Class object, except in the specific cases where the Ack_Required property is ignored (see Clauses 12.31.40 and 12.52.8).

When an event state transition is received, the corresponding bit in Acked_Transitions is either set or cleared. If the corresponding bit in Ack_Required is set, then the bit in Acked_Transitions is cleared, otherwise it is set.

When an acknowledgment indication is received, the corresponding bit in Acked_Transitions is set and an Acknowledgment transition is indicated to the event-notification-distribution process.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Modification of Ack_Required does not change the value of Acked_Transitions properties of associated objects.

13.2.4 Event-Summarization

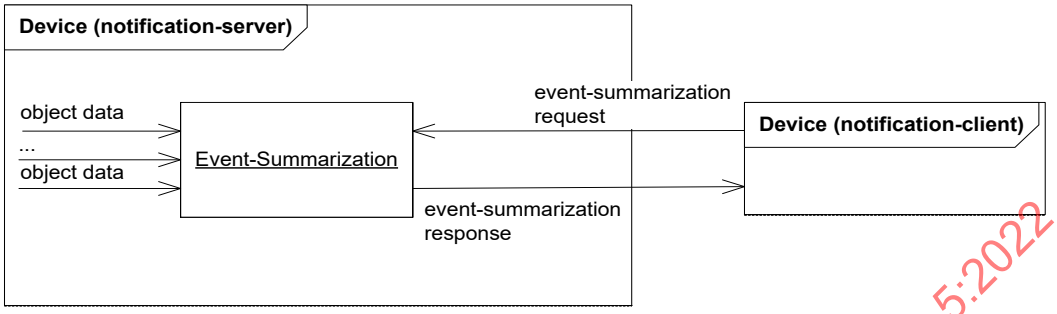


Figure 13-6. Event-Summarization

Event-summarization (see Figure 13-6) provides the means by which a notification client can efficiently determine the current state of multiple event-initiating objects in a device when it has not received all of the event notifications (either because it was offline, the network is down, or the client is not in the distribution list for the event-initiating objects). The different BACnet services that can be used for event-summarization are compared in Table 13-2.

Table 13-2. Event-Summarization Services

Service	Selected objects	Response
GetAlarmSummary (see Clause 13.10)	All event-initiating objects where: Event_Detection_Enable is TRUE and Event_State is not NORMAL and Notify_Type equal to ALARM	List of Object_Identifier Event_State Acked_Transitions
GetEnrollmentSummary (see Clause 13.11)	All event-initiating objects where: Event_Detection_Enable is TRUE and Acknowledgment Filter matches and Enrollment Filter matches (optional) and Event State Filter matches (optional) and Event Type Filter matches (optional) and Priority Filter matches (optional) and Notification Class Filter matches(optional)	List of Object_Identifier Event Type Event_State Priority Notification_Class
GetEventInformation (see Clause 13.12)	All event-initiating objects where: Event_Detection_Enable is TRUE and Event_State is not NORMAL or any bit in the Acked_Transitions property is not set	List of Object_Identifier Event_State Acked_Transitions Event_Time_Stamps Notify_Type Event_Enable Event Priorities

Notification-servers are required to support execution of the GetEventInformation service. Support for the execution of the GetAlarmSummary and GetEnrollmentSummary services is recommended to not be implemented in devices. The specification of these services is retained for historical reference so that implementations of client devices have guidance on how to interoperate with older server devices.

It is important to note that the results returned by the summarization services ignore the value of the event-initiating objects' Event_Enable properties and the notification filtering fields in the Recipient_List property of related Notification

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Class objects. This results in the set of objects returned by the services being different than the set of objects a notification-client would be made aware of via the event notification services.

13.2.5 Event-Notification-Distribution

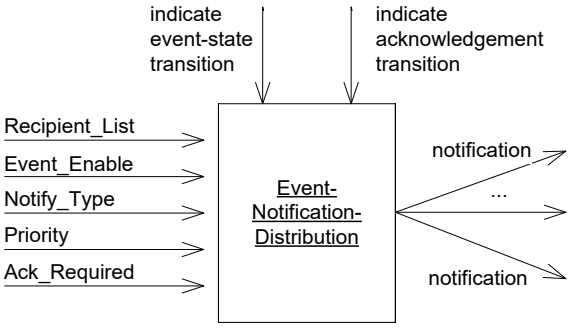


Figure 13-7. Event-Notification-Distribution

The event-notification-distribution process (see Figure 13-7) is responsible for distributing event transition notifications to all local objects and to notification-clients. Event transition notifications and acknowledgment notifications are distributed to notification-clients via the ConfirmedEventNotification and UnconfirmedEventNotification services.

The Event_Enable and Notify_Type inputs are provided by the event-initiating object, and the Recipient_List, Priority, and Ack_Required inputs are provided by the Notification Class object referenced by the event-initiating object.

When an event state transition or an acknowledgment transition is indicated, notifications are distributed to the notification-clients specified by the Recipient_List input. Additional information provided by the Recipient_List input controls the distribution of the notification. For acknowledgment notifications, the transition type of the notification (TO_FAULT, TO_NORMAL, or TO_OFFNORMAL) is the same as the transition type of the event transition that is being acknowledged.

The external distribution of notifications can be disabled by the Event_Enable property. The notifications are disabled if the bit corresponding to the transition is set to FALSE. While the Event_Enable property can take on any combination of three bits, the two most used values are (T, T, T), indicating that all transitions are to be distributed, or (F, F, F), indicating that no transitions are to be distributed.

Distribution of notifications to local objects (e.g., Event Log objects) in the device is not controlled by the Recipient_List information unless otherwise indicated (e.g., Notification Forwarder objects). It is a local matter whether or not the distribution of notifications to local objects is affected by the Event_Enable property (e.g., whether or not Event Log objects record the notifications).

13.2.5.1 Notification Forwarding

Notification forwarding is related to, but separate from, the distribution of event notifications by the Notification Class object.

The forwarding of notifications, whether by a local or remote Notification Forwarder object, modifies the notification distribution as setup by a Notification Class. The Notification Class distributes the notification to the device that contains the Notification Forwarder object, and the Notification Forwarder object re-sends the notification with recipient information from the Notification Forwarder object. See Clause 12.51 for a detailed description of notification forwarding as implemented by Notification Forwarding objects.

13.2.5.2 Service Parameters of Event Notification Service Requests

The event-notification-distribution process generates event notifications and the service parameters for those notifications are generated according to Table 13-3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 13-3. Event Notification Service Parameter Values

Service Parameter	Event State Transition (all transitions)	Acknowledgment Transition
Process Identifier	From Recipient_List entry	From Recipient_List entry
Initiating Device Identifier	Device object identifier of the device which contains the event-initiating object	Device object identifier of the device which contains the event-initiating object
Event Object Identifier	Object identifier of the event-initiating object	Object identifier of the event-initiating object
Time Stamp	Value of the Event_Time_Stamps array entry that corresponds to the 'To State'.	The time at which the AcknowledgeAlarm service is executed or If acknowledged locally, the time that the acknowledgment is performed.
Notification Class	Value of the event-initiating object's Notification_Class property.	Value of the event-initiating object's Notification_Class property.
Priority	Value of the Priority property entry that corresponds to the 'To State'	Value of the Priority property entry that corresponds to the 'To State'.
Event Type	When 'To State' or 'From State' is FAULT, set to CHANGE_OF_RELIABILITY, Otherwise the value associated with the event-initiating object's event algorithm.	When 'To State' is FAULT, set to CHANGE_OF_RELIABILITY. When 'To State' is NORMAL, and the device can determine reporting acknowledgement of a transition from FAULT, set to CHANGE_OF_RELIABILITY. Otherwise the value associated with the event-initiating object's configured event algorithm.
Message Text	Optional The value is derived from the value in the Event_Message_Texts_Config property configured in the entry corresponding to the transition, if the property exists. Otherwise the value is a local matter. The Message Text transmitted is reflected in the Event_Message_Texts array, if the property exists.	Optional The value is a local matter.
Notify Type	Value of Notify_Type	ACK_NOTIFICATION
AckRequired	Value of the Ack_Required bit that corresponds to 'To State'.	Not present
From State	Value of Event_State before this transition	Not present
To State	Value of property Event_State after this transition	The 'To State' parameter from the transition being acknowledged
Event Values	As defined for the Event_Type	Not present

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

13.2.5.3 Fault Event Notifications

For all transitions to, or from, the FAULT state, the corresponding event notification shall use the Event Type CHANGE_OF_RELIABILITY.

© ISO 2022 – All rights reserved

Table 13-4. CHANGE_OF_RELIABILITY Notification Parameters

Parameter	Value
reliability	The value of the Reliability property of the event-initiating object.
Status-flags	The value of the Status_Flags property of the event-initiating object.
Property-values	As specified in Table 13-5.

The content of the property-values parameter of CHANGE_OF_RELIABILITY event notifications depends on the type of the event-initiating object. The property values required to be conveyed are specified in Table 13-5, and shall be included in the order shown in the table. Object types that are not included in Table 13-5 are not required to include any extra properties in CHANGE_OF_RELIABILITY notifications.

Additional properties of the event-initiating object may be conveyed in the property-values parameter following the properties which are required by this standard. The selection of additional properties to include in the event notification is a local matter.

In the case of the Event Enrollment object, the first property in the property-values parameter shall be the Event Enrollment object's Object_Property_Reference property. The second property is the property that is referenced by the Event Enrollment object's Object_Property_Reference property. All other properties (required and additional properties) shall be from the monitored object. This is the only case where the properties conveyed in the CHANGE_OF_RELIABILITY are not from the event-initiating object.

Table 13-5. Properties Reported in CHANGE_OF_RELIABILITY Notifications

Object Type	Properties
Access Door	Door_Alarm_State Present_Value
Access Point	Access_Event Access_Event_Tag Access_Event_Time Access_Event_Credential
Access Zone	Occupancy_State
Accumulator	Pulse_Rate Present_Value
Analog Input, Analog Output, Analog Value, Binary Input, Binary Value, BitString Value, Channel, CharacterString Value, Global Group, Integer Value, Large Analog Value, Lighting Output, Multi-state Input, Multi-state Value, Positive Integer Value, Pulse Converter	Present_Value
Binary Output, Binary Lighting Output, Multi-state Output	Present_Value Feedback_Value[3]
Credential Data Input	Update_Time Present_Value[1]
Escalator Lift	Fault_Signals[2]

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Object Type	Properties
Event Enrollment	Object_Property_Reference Value of property referenced by Object_Property_Reference[2] Reliability[2] Status_Flags[2]
Life Safety Point, Life Safety Zone	Present_Value Mode Operation_Expected
Load Control	Present_Value Requested_Shed_Level Actual_Shed_Level
Loop	Present_Value Controlled_Variable_Value[2] Setpoint[2]
Program	Program_State Reason_For_Halt[2,3] Description_Of_Halt[2,3]
Staging	Present_Value Present_Stage
Timer	Present_Value Timer_State Update_Time[3] Last_State_Change[3] Initial_Timeout[3] Expiration_Time[3]

[1] This value may be excluded from the property-value parameter due to security requirements.
[2] This property is, or may be, from a referenced object. If the value is not known by the event-initiating object, then it shall not be included in the property-value parameter.
[3] These properties are optional and are included only if present in the object.

13.2.5.4 Alarm and Event Priority Classification

Alarms and events traversing the BACnet network need prioritization to assure that important information reaches its destination and is acted upon quickly. To assure alarm prioritization at the network level, the Network Priority as defined in Clause 6.2.2 shall be set as a function of the alarm and event priority as defined in Table 13-6. Annex M provides additional clarity and examples of specific messages and priorities.

Table 13-6. Alarm and Event Priority - Network Priority Association

Alarm and Event Priority	Network Priority
00 - 63	Life Safety message
64 - 127	Critical Equipment message
128 - 191	Urgent message
192 - 255	Normal message

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.3 Event Algorithms

Table 13-7 lists the event algorithms that are specified in this standard. The event algorithms are indicated by the BACnetEventType value of the same name.

Table 13-7. Standardized Event Algorithms

Event Algorithm	Clause
NONE	13.3.17
ACCESS_EVENT	13.3.12
BUFFER_READY	13.3.7
CHANGE_OF_BITSTRING	13.3.1
CHANGE_OF_CHARACTERSTRING	13.3.16
CHANGE_OF_DISCRETE_VALUE	13.3.18
CHANGE_OF_LIFE_SAFETY	13.3.8
CHANGE_OF_STATE	13.3.2
CHANGE_OF_STATUS_FLAGS	13.3.11
CHANGE_OF_TIMER	13.3.19
CHANGE_OF_VALUE	13.3.3
COMMAND_FAILURE	13.3.4
DOUBLE_OUT_OF_RANGE	13.3.13
EXTENDED	13.3.10
FLOATING_LIMIT	13.3.5
OUT_OF_RANGE	13.3.6
SIGNED_OUT_OF_RANGE	13.3.14
UNSIGNED_OUT_OF_RANGE	13.3.15
UNSIGNED_RANGE	13.3.9

Event algorithms monitor a value and evaluate whether the condition for a transition of event state exists. The result of the evaluation, indicated to the Event-State-Detection process, may be a transition to a new event state, a transition to the same event state, or no transition. The final determination of the Event_State property value is the responsibility of the Event-State-Detection process and is subject to additional conditions. See Clause 13.2.

Each of the event algorithms defines its input parameters, the allowable normal and offnormal states, the conditions for transitions between those states, and the notification parameters conveyed in event notifications for the algorithm.

When executing an event algorithm, all conditions defined for the algorithm shall be evaluated in the order as presented for the algorithm. Some algorithms specify optional conditions, marked as "Optional:" Whether or not an implementation uses these conditions is a local matter. If no condition evaluates to true, then no transition shall be indicated to the Event-State-Detection process.

13.3.1 CHANGE_OF_BITSTRING Event Algorithm

The CHANGE_OF_BITSTRING event algorithm detects whether the monitored value of type BIT STRING equals a value that is listed as an alarm value, after applying a bitmask.

The parameters of this event algorithm are:

pCurrentState — This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

pMonitoredValue — This parameter, of type BIT STRING, represents the current value of the monitored property.

pStatusFlags — This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pAlarmValues	This parameter, of type list of BIT STRING, represents a list of values that are considered offnormal values.
pBitmask	This parameter, of type BIT STRING, represents the bitmask that defines the bits of pMonitoredValue that are significant for comparison with values of pAlarmValues. This value is bit-wise ANDed with the pMonitoredValue before comparison with pAlarmValues.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue, after applying pBitmask, is equal to any of the values contained in pAlarmValues for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue, after applying pBitmask, is not equal to any of the values contained in pAlarmValues for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(c) Optional: If pCurrentState is OFFNORMAL, and pMonitoredValue, after applying pBitmask, is equal to one of the values contained in pAlarmValues that is different from the value that caused the last transition to OFFNORMAL and remains equal to that value for pTimeDelay, then indicate a transition to the OFFNORMAL event state.

Figure 13-8 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

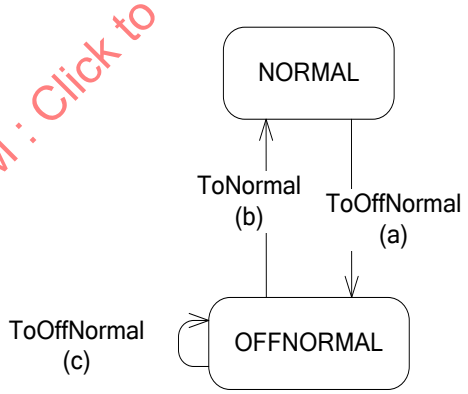


Figure 13-8. Transitions indicated by CHANGE_OF_BITSTRING algorithm

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The notification parameters of this algorithm are:

Referenced_Bitstring	This notification parameter, of type BIT STRING, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

13.3.2 CHANGE_OF_STATE Event Algorithm

The CHANGE_OF_STATE event algorithm detects whether the monitored value equals a value that is listed as an alarm value. The monitored value may be of any discrete or enumerated data type, including Boolean.

© ISO 2022 – All rights reserved

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter is a discrete value that represents the current value of the monitored property. The datatype of the value of this parameter shall be one of the options of BACnetPropertyStates.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pAlarmValues	This parameter is a list of discrete values that represent the offnormal values. The datatype of the values of this parameter and of pMonitoredValue shall be the same.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is equal to any of the values contained in pAlarmValues for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue is not equal to any of the values contained in pAlarmValues for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(c) Optional: If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to one of the values contained in pAlarmValues that is different from the value that caused the last transition to OFFNORMAL, and remains equal to that value for pTimeDelay, then indicate a transition to the OFFNORMAL event state.

Figure 13-9 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

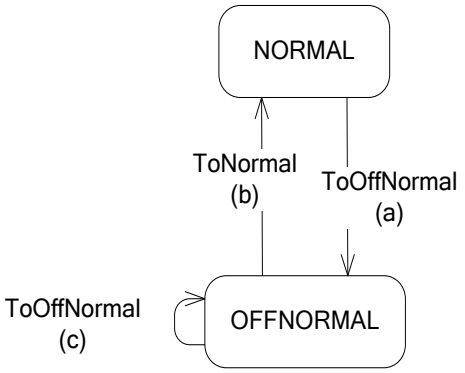


Figure 13-9. Transitions indicated by CHANGE_OF_STATE algorithm

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The notification parameters of this algorithm are:

New_State	This notification parameter, of type BACnetPropertyStates, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

13.3.3 CHANGE_OF_VALUE Event Algorithm

The CHANGE_OF_VALUE event algorithm, for monitored values of datatype REAL, detects whether the absolute value of the monitored value changes by an amount equal to or greater than a positive REAL increment.

The CHANGE_OF_VALUE event algorithm, for monitored values of datatype BIT STRING, detects whether the monitored value changes in any of the bits specified by a bitmask.

For detection of change, the value of the monitored value when a transition to NORMAL is indicated shall be used in evaluation of the conditions until the next transition to NORMAL is indicated. The initialization of the value used in evaluation before the first transition to NORMAL is indicated is a local matter.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type REAL or BIT STRING, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pIncrement	This parameter, of type REAL, shall provide a value if and only if pMonitoredValue is of type REAL. It represents the positive increment by which the monitored value of type REAL must change for a new transition.
pBitmask	This parameter, of type BIT STRING, shall provide a value if and only if pMonitoredValue is of type BIT STRING. It represents the bitmask that defines the bits of pMonitoredValue that are significant for detecting a change of value. This value is bit-wise ANDed with the pMonitoredValue before comparison with the value that has caused the last transition to NORMAL.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm, for a monitored value of type REAL, are:

(a) If pCurrentState is NORMAL, and the absolute value of pMonitoredValue changes by an amount equal to or greater than pIncrement for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The conditions evaluated by this event algorithm, for a monitored value of type BIT STRING, are:

(a) If pCurrentState is NORMAL, and any of the significant bits of pMonitoredValue change state and remain changed for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

Figure 13-10 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

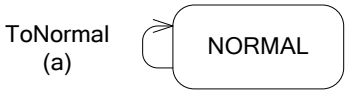


Figure 13-10. Transitions indicated by CHANGE_OF_VALUE algorithm

The notification parameters of this algorithm are:

New_Value	This notification parameter, of type REAL or BIT STRING, conveys the value of pMonitoredValue. If pMonitoredValue is of type BIT STRING, then the Changed_Bits option is used. If pMonitoredValue is of type REAL, then the Changed_Value option is used.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

13.3.4 COMMAND_FAILURE Event Algorithm

The COMMAND_FAILURE event algorithm detects whether the monitored value and the feedback value disagree for a time period. It may be used, for example, to verify that a process change has occurred after writing a property.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type ABSTRACT-SYNTAX.&Type, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pFeedbackValue	This parameter, of type ABSTRACT-SYNTAX.&Type, represents the feedback value used for comparison with the pMonitoredValue. The datatype of the value of this parameter shall be the same as the datatype of the value of pMonitoredValue.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pFeedbackValue is not equal to pMonitoredValue for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to pMonitoredValue for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

Figure 13-11 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

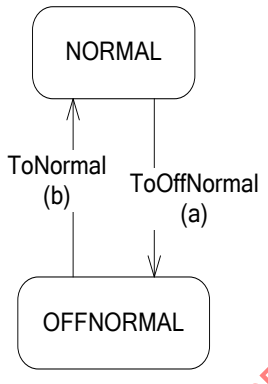


Figure 13-11. Transitions indicated by COMMAND_FAILURE algorithm

The notification parameters of this algorithm are:

Command_Value	This notification parameter, of type ABSTRACT-SYNTAX.&Type, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Feedback_Value	This notification parameter, of type ABSTRACT-SYNTAX.&Type, conveys the value of pFeedbackValue.

13.3.5 FLOATING_LIMIT Event Algorithm

The FLOATING_LIMIT event algorithm detects whether the monitored value exceeds a range defined by a setpoint, a high difference limit, a low difference limit and a deadband.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type REAL, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pSetpoint	This parameter, of type REAL, represents the value that defines the reference interval.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pLowDiffLimit	This parameter, of type REAL, represents, when subtracted from pSetpoint, the lower limit of the range considered normal.
pHighDiffLimit	This parameter, of type REAL, represents, when added to pSetpoint, the higher limit of the range considered normal.
pDeadband	This parameter, of type REAL, represents the deadband that is applied to the respective limit before a return to Normal event state is indicated.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

Figure 13-12 shows the relationship of the various parameters used in the FLOATING_LIMIT algorithm. In this figure, pTimeDelay is assumed to have a value of zero.

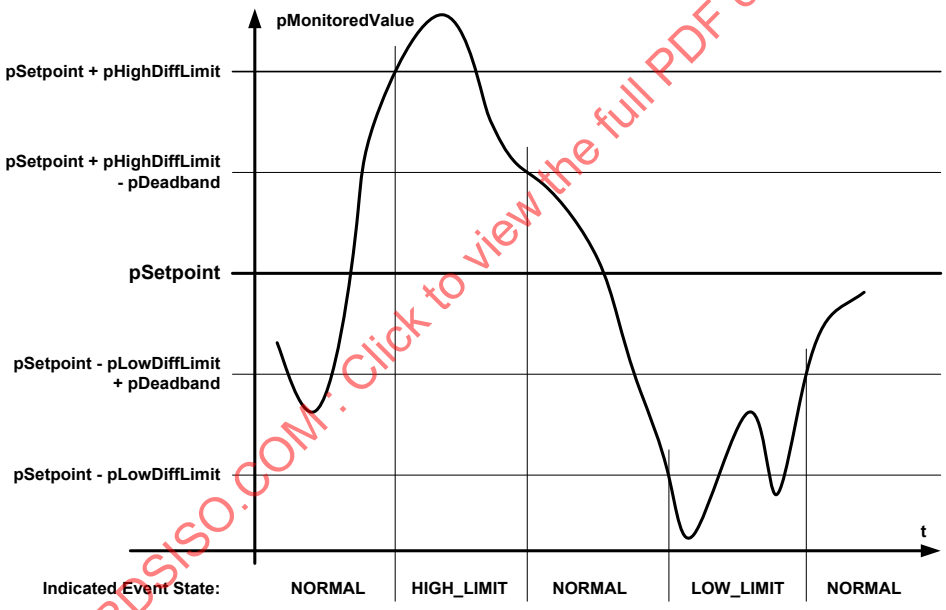


Figure 13-12. FLOATING_LIMIT parameter relationship

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is greater than (pSetpoint + pHighDiffLimit) for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(b) If pCurrentState is NORMAL, and pMonitoredValue is less than (pSetpoint - pLowDiffLimit) for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.

(c) Optional: If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pSetpoint - pLowDiffLimit) for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(d) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pSetpoint + pHighDiffLimit - pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

(e) Optional: If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pSetpoint + pHighDiffLimit) for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.

© ISO 2022 – All rights reserved

(f) If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pSetpoint - pLowDiffLimit + pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-13 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

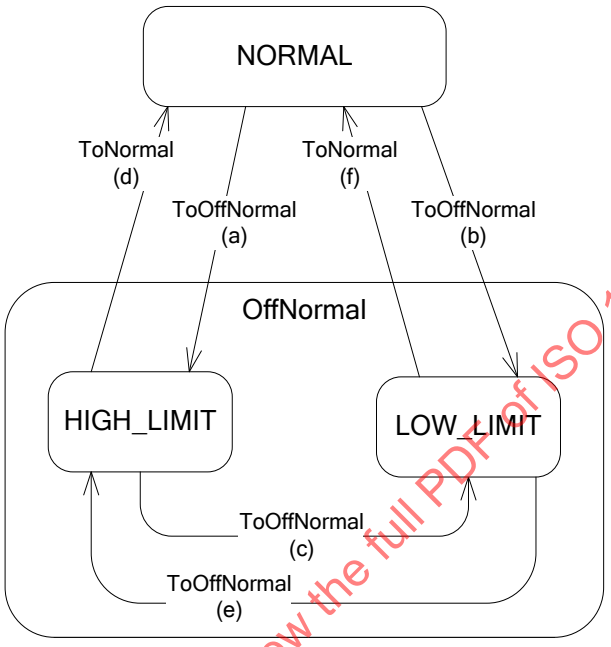


Figure 13-13. Transitions indicated by FLOATING_LIMIT algorithm

The notification parameters of this algorithm are:

Reference_Value	This notification parameter, of type REAL, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Setpoint_Value	This notification parameter, of type REAL, conveys the value of pSetpoint.
Error_Limit	This notification parameter, of type REAL, conveys the value of pLowDiffLimit if (a) the new state is LOW_LIMIT, or (b) pCurrentState is LOW_LIMIT and the new state is NORMAL This notification parameter conveys the value of pHighDiffLimit if (a) the new state is HIGH_LIMIT, or (b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

13.3.6 OUT_OF_RANGE Event Algorithm

The OUT_OF_RANGE event algorithm detects whether the monitored value exceeds a range defined by a high limit and a low limit. Each of these limits may be enabled or disabled. If disabled, the normal range has no higher limit or no lower limit. In order to reduce jitter of the resulting event state, a deadband is applied when the value is in the process of returning to the normal range.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type REAL, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pLowLimit	This parameter, of type REAL, represents the lower limit of the range considered normal.
pHighLimit	This parameter, of type REAL, represents the higher limit of the range considered normal.
pDeadband	This parameter, of type REAL, represents the deadband that is applied to the respective limit before a return to Normal event state is indicated.
pLimitEnable	This parameter, of type BACnetLimitEnable, represents two flags, HighLimitEnable and LowLimitEnable, that separately enable (TRUE) or disable (FALSE) the respective limits applied by the event algorithm. If the value of this parameter is not provided, then both flags shall be set to TRUE (1).
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(b) If pCurrentState is NORMAL, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.

(c) If pCurrentState is HIGH_LIMIT, and the HighLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(d) Optional: If pCurrentState is HIGH_LIMIT, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(e) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pHighLimit – pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

(f) If pCurrentState is LOW_LIMIT, and the LowLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(g) Optional: If pCurrentState is LOW_LIMIT, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.

(h) If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pLowLimit + pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-14 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

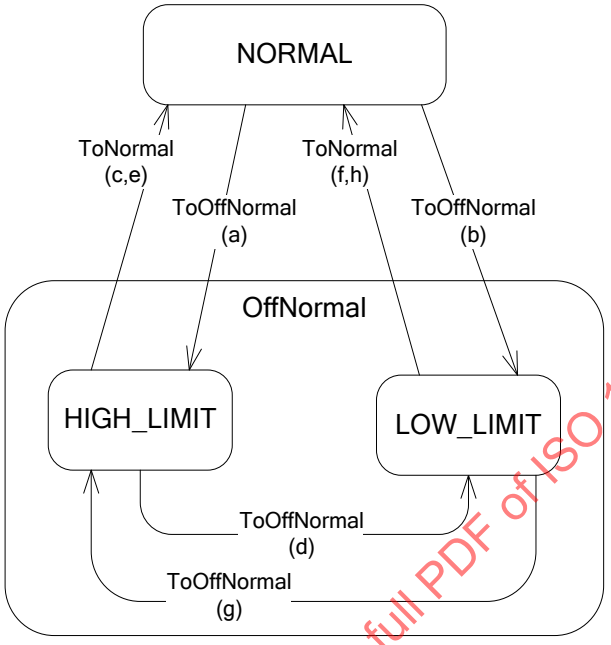


Figure 13-14. Transitions indicated by OUT_OF_RANGE algorithm

The notification parameters of this algorithm are:

Exceeding_Value — This notification parameter, of type REAL, conveys the value of pMonitoredValue.

Status_Flags — This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

Deadband — This notification parameter, of type REAL, conveys the value of pDeadband.

Exceeded_Limit — This notification parameter, of type REAL, conveys the value of pLowLimit if
(a) the new state is LOW_LIMIT, or
(b) pCurrentState is LOW_LIMIT and the new state is NORMAL
This notification parameter conveys the value of pHighLimit if
(a) the new state is HIGH_LIMIT, or
(b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

13.3.7 BUFFER_READY Event Algorithm

The BUFFER_READY event algorithm detects whether a defined number of records have been added to a log buffer since start of operation or the previous notification, whichever is most recent.

The parameters of this event algorithm are:

pCurrentState — This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

pMonitoredValue — This parameter, of type Unsigned32, represents the current total count of records in the log buffer referenced by pLogBuffer.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pLogBuffer	This parameter, of type BACnetDeviceObjectPropertyReference, represents the reference to the log buffer property for which this algorithm is applied.
pThreshold	This parameter, of type Unsigned32, represents the number of records that, when added to the log buffer, will result in a transition to NORMAL. If this parameter has a value of 0, then no transitions will be indicated by the algorithm.
pPreviousCount	This parameter, of type Unsigned32, represents the value of pMonitoredValue at the time the most recent transition to NORMAL was indicated. Upon initialization of the event algorithm, this parameter shall be set to the value of pMonitoredValue. When a transition to NORMAL is indicated, this parameter shall be updated to the value of pMonitoredValue.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is greater than or equal to pPreviousCount, and (pMonitoredValue – pPreviousCount) is greater than or equal to pThreshold and pThreshold is greater than 0, then indicate a transition to the NORMAL event state.
(b) If pCurrentState is NORMAL, and pMonitoredValue is less than pPreviousCount, and (pMonitoredValue – pPreviousCount + 2^{32} – 1) is greater than or equal to pThreshold and pThreshold is greater than 0, then indicate a transition to the NORMAL event state.

Figure 13-15 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

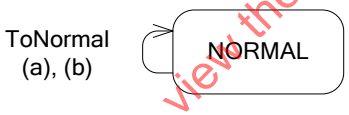


Figure 13-15. Transitions indicated by BUFFER_READY algorithm

The notification parameters of this algorithm are:

Buffer_Property	This notification parameter, of type BACnetDeviceObjectPropertyReference, conveys the value of pLogBuffer.
Previous_Notification	This notification parameter, of type Unsigned32, conveys the value of pPreviousCount.
Current_Notification	This notification parameter, of type Unsigned32, conveys the value of pMonitoredValue.

13.3.8 CHANGE_OF_LIFE_SAFETY Event Algorithm

The CHANGE_OF_LIFE_SAFETY event algorithm detects whether the monitored value equals a value that is listed as an alarm value or life safety alarm value. Event state transitions are also indicated if the value of the mode parameter changed since the last transition indicated. In this case, any time delays are overridden and the transition is indicated immediately.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type BACnetLifeSafetyState, represents the current value of the monitored property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pMode	This parameter, of type BACnetLifeSafetyMode, represents the current life safety mode of operation.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pOperationExpected	This parameter, of type BACnetLifeSafetyOperation, represents the currently expected life safety operation.
pAlarmValues	This parameter, of type list of BACnetLifeSafetyState, represents a list of values that are considered offnormal values.
pLifeSafetyAlarmValues	This parameter, of type list of BACnetLifeSafetyState, represents a list of values that are considered life safety alarm values.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is equal to any of the values contained in pAlarmValues, and remains within the set of values of pAlarmValues either for pTimeDelay or for pMode changes, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is NORMAL, and pMonitoredValue is equal to any of the values contained in pLifeSafetyAlarmValues, and remains within the set of values of pLifeSafetyAlarmValues either for pTimeDelay or for pMode changes, then indicate a transition to the LIFE_SAFETY_ALARM event state.
(c) If pCurrentState is NORMAL, and pMode changes, then indicate a transition to the NORMAL event state.
(d) If pCurrentState is OFFNORMAL, and pMonitoredValue is not equal to any of the values contained in pAlarmValues and pLifeSafetyAlarmValues either for pTimeDelayNormal or for pMode changes, then indicate a transition to the NORMAL event state.
(e) If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to any of the values contained in pLifeSafetyAlarmValues, and remains within the set of values of pLifeSafetyAlarmValues either for pTimeDelay or for pMode changes, then indicate a transition to the LIFE_SAFETY_ALARM event state.
(f) Optional: If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to one of the values contained in pAlarmValues that is different from the value causing the last transition to OFFNORMAL, and remains equal to that value for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(g) If pCurrentState is OFFNORMAL, and pMode changes, then indicate a transition to the OFFNORMAL event state.
(h) If pCurrentState is LIFE_SAFETY_ALARM, and pMonitoredValue is not equal to any of the values contained in pAlarmValues and pLifeSafetyAlarmValues either for pTimeDelayNormal or for pMode changes, then indicate a transition to the NORMAL event state.
(i) If pCurrentState is LIFE_SAFETY_ALARM, and pMonitoredValue is equal to any of the values contained in pAlarmValues, and remains within the set of values of pAlarmValues either for pTimeDelay or for pMode changes, then indicate a transition to the OFFNORMAL event state.
(j) Optional: If pCurrentState is LIFE_SAFETY_ALARM, and pMonitoredValue is equal to one of the values contained in pLifeSafetyAlarmValues that is different from the value causing the last transition to LIFE_SAFETY_ALARM, and remains equal to that value for pTimeDelay, then indicate a transition to the LIFE_SAFETY_ALARM event state.
(k) If pCurrentState is LIFE_SAFETY_ALARM, and pMode changes, then indicate a transition to the LIFE_SAFETY_ALARM event state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(l) If pCurrentState is OFFNORMAL, and current conditions require a re-alert, then indicate a transition to the OFFNORMAL event state. The conditions requiring a re-alert are a local matter. For example, the authority having jurisdiction requires re-alert for an OFFNORMAL condition.
(m) If pCurrentState is LIFE_SAFETY_ALARM, and current conditions require a re-alert, then indicate a transition to the LIFE_SAFETY_ALARM event state. The conditions requiring a re-alert are a local matter. For example, the authority having jurisdiction requires re-alert for a LIFE_SAFETY_ALARM condition.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-16 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

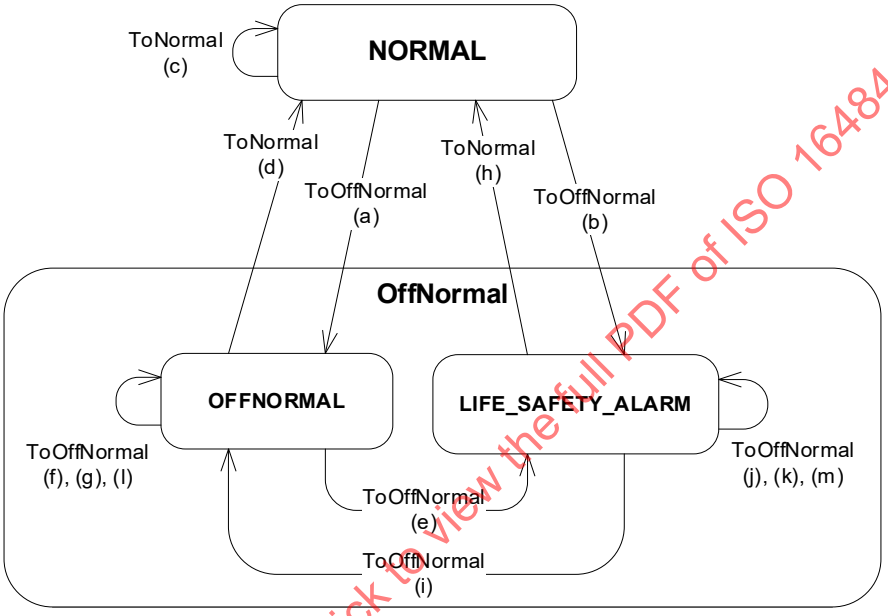


Figure 13-16. Transitions indicated by CHANGE_OF_LIFE_SAFETY algorithm

The notification parameters of this algorithm are:

New_State	This notification parameter, of type BACnetLifeSafetyState, conveys the value of pMonitoredValue.
New_Mode	This notification parameter, of type BACnetLifeSafetyMode, conveys the value of pMode.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Operation_Expected	This notification parameter, of type BACnetLifeSafetyOperation, conveys the value of pOperationExpected.

13.3.9 UNSIGNED_RANGE Event Algorithm

The UNSIGNED_RANGE event algorithm detects whether the monitored value exceeds a range defined by a high limit and a low limit.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pMonitoredValue	This parameter, of type Unsigned, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pLowLimit	This parameter, of type Unsigned, represents the lower limit of the range considered normal.
pHighLimit	This parameter, of type Unsigned, represents the higher limit of the range considered normal.
pLimitEnable	This parameter, of type BACnetLimitEnable, represents two flags, HighLimitEnable and LowLimitEnable, that separately enable (TRUE) or disable (FALSE) the respective limits applied by the event algorithm. If the value of this parameter is not provided, then both flags shall be set to TRUE (1).
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.

(b) If pCurrentState is NORMAL, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.

(c) If pCurrentState is HIGH_LIMIT, and the HighLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.

(d) Optional: If pCurrentState is HIGH_LIMIT, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.

(e) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is equal to or less than pHighLimit for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

(f) If pCurrentState is LOW_LIMIT, and the LowLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.

(g) Optional: If pCurrentState is LOW_LIMIT, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.

(h) If pCurrentState is LOW_LIMIT, and pMonitoredValue is equal to or greater than pLowLimit, for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-17 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

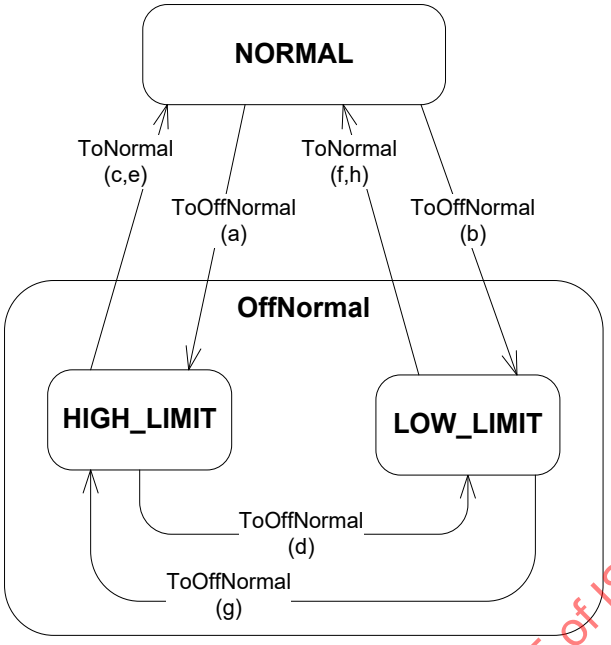


Figure 13-17. Transitions indicated by UNSIGNED_RANGE algorithm

The notification parameters of this algorithm are:

Exceeding_Value	This notification parameter, of type Unsigned, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Exceeded_Limit	This notification parameter, of type Unsigned, conveys the value of pLowLimit if (a) the new state is LOW_LIMIT, or (b) pCurrentState is LOW_LIMIT and the new state is NORMAL This notification parameter conveys the value of pHighLimit if (a) the new state is HIGH_LIMIT, or (b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

13.3.10 EXTENDED Event Algorithm

The EXTENDED event algorithm detects event conditions based on a proprietary event algorithm. The proprietary event algorithm uses parameters and conditions defined by the vendor. The algorithm is identified by a vendor-specific event type that is in the scope of the vendor's vendor identification code. The algorithm may, at the vendor's discretion, indicate a new event state, a transition to the same event state, or no transition to the Event-State-Detection. The indicated new event states may be NORMAL, and any offnormal event state. FAULT event state may not be indicated by this algorithm. For the purpose of proprietary evaluation of unreliability conditions that may result in FAULT event state, a FAULT_EXTENDED fault algorithm shall be used.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pVendorId	This parameter, of type Unsigned16, represents the vendor identification code for the event type of the vendor-proprietary event algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pEventType	This parameter, of type Unsigned, represents the vendor-proprietary event algorithm.
pParameters	This parameter is a sequence of primitive or constructed values that represent algorithm parameters whose interpretation is specific to the proprietary event algorithm. Values of this parameter may be used to provide values of the Parameters notification parameter.

The conditions evaluated by this event algorithm are vendor-specific, and are identified by pVendorId and pEventType. This algorithm may support multiple offnormal event states, including proprietary offnormal event states.

Figure 13-18 depicts those transitions of Figure 13-3 that this event algorithm may indicate. The particular offnormal states shown are for illustration only.

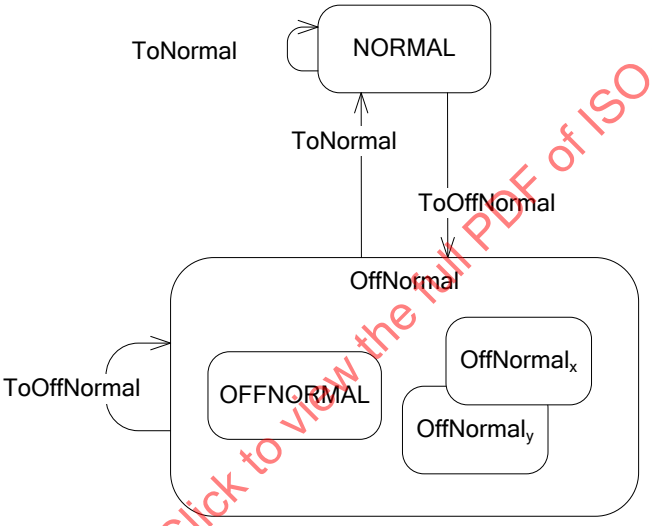


Figure 13-18. Transitions indicated by EXTENDED algorithm

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The notification parameters of this algorithm are:

Vendor_Id	This notification parameter, of type Unsigned16, conveys the value of pVendorId.
Extended_Event_Type	This notification parameter, of type Unsigned, conveys the value of pEventType.
Parameters	This notification parameter is a sequence of primitive or constructed values whose content and interpretation is specific to the proprietary event algorithm.

13.3.11 CHANGE_OF_STATUS_FLAGS Event Algorithm

The CHANGE_OF_STATUS_FLAGS event algorithm detects whether a significant flag of the monitored value of type BACnetStatusFlags has the value TRUE.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

© ISO 2022 – All rights reserved

pMonitoredValue	This parameter, of type BACnetStatusFlags, represents the current value of the monitored property.
pSelectedFlags	This parameter, of type BACnetStatusFlags, represents the flags of pMonitoredValue that are selected to be significant for evaluation. A value of TRUE in a flag indicates that the corresponding flag in pMonitoredValue is significant for evaluation. A value of FALSE in a flag indicates that the corresponding flag in pMonitoredValue is not significant for evaluation.
pPresentValue	This optional parameter, of type ABSTRACT-SYNTAX.&Type, represents the current value of the Present_Value property of the object containing the property that provides the value of the pMonitoredValue parameter. This parameter may be omitted if it is determined to be too large. The method of such determination is a local matter.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue has a value of TRUE in any of its flags that also has a value of TRUE in the corresponding flag in pSelectedFlags for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue has none of its flags set to TRUE that also has a value of TRUE in the corresponding flag in the pSelectedFlags event parameter for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(c) If pCurrentState is OFFNORMAL, and the set of selected flags of pMonitoredValue that have a value of TRUE changes, then indicate a transition to the OFFNORMAL event state.

Figure 13-19 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

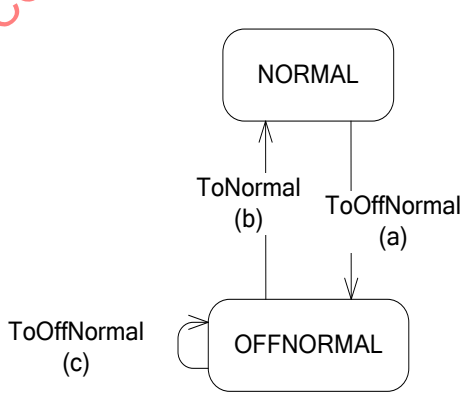


Figure 13-19. Transitions indicated by CHANGE_OF_STATUS_FLAGS algorithm

The notification parameters of this algorithm are:

Present_Value	This optional notification parameter, of type ABSTRACT-SYNTAX.&Type, conveys the value of pPresentValue. This parameter is optional and may be omitted if it is not provided to the algorithm, or if it is determined to be too large. The method of such determination is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Referenced_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pMonitoredValue.

13.3.12 ACCESS_EVENT Event Algorithm

The ACCESS_EVENT event algorithm detects whether the access event time has changed and the new access event value equals a value that is listed to cause a transition to NORMAL.

For detection of change, the access event time when a transition to NORMAL is indicated shall be used in evaluation of the conditions until the next transition to NORMAL is indicated. The initialization of the access event time used in evaluation before the first transition to NORMAL is indicated is a local matter.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type BACnetAccessEvent, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pAccessEvents	This parameter, of type list of BACnetAccessEvent, represents a list of values that are considered access event values that shall cause a transition indication.
pAccessEventTag	This parameter, of type Unsigned, represents the current value of the Access_Event_Tag property of the object containing the property that provides the value of the pMonitoredValue parameter.
pAccessEventTime	This parameter, of type BACnetTimeStamp, represents the update time of the monitored access event value.
pAccessCredential	This parameter, of type BACnetDeviceObjectReference, represents the current value of the Access_Event_Credential property of the object containing the property that provides the value of the pMonitoredValue parameter.
pAccessFactor	This optional parameter, of type BACnetDeviceObjectReference, represents the current value of the Access_Event_Authentication_Factor property of the object containing the property that provides the value of the pMonitoredValue parameter. This parameter may be omitted if it is determined to be too large, or omitted for security reasons. The method of such determination is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pAccessEventTime changes, and pMonitoredValue is equal to any of the values contained in pAccessEvents, then indicate a transition to the NORMAL event state.

Figure 13-20 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

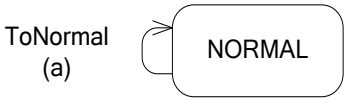


Figure 13-20. Transitions indicated by ACCESS_EVENT algorithm

© ISO 2022 – All rights reserved

The notification parameters of this algorithm are:

Access_Event	This notification parameter, of type BACnetAccessEvent, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Access_Event_Tag	This notification parameter, of type Unsigned, conveys the value of pAccessEventTag.
Access_Event_Time	This notification parameter, of type BACnetTimeStamp, conveys the value of pAccessEventTime.
Access_Credential	This notification parameter, of type BACnetDeviceObjectReference, conveys the value of pAccessCredential.
Authentication_Factor	This optional notification parameter, of type BACnetAuthenticationFactor, conveys the value of pAccessFactor. This parameter may be omitted if it is not provided to the algorithm, is determined to be too large for the event notification, or omitted for security reasons. The method of such determination is a local matter.

13.3.13 DOUBLE_OUT_OF_RANGE Event Algorithm

The DOUBLE_OUT_OF_RANGE event algorithm detects whether the monitored value exceeds a range defined by a high limit and a low limit. Each of these limits may be enabled or disabled. If disabled, the normal range has no lower limit or no higher limit respectively. In order to reduce jitter of the resulting event state, a deadband is applied when the value is in the process of returning to the normal range.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type Double, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pLowLimit	This parameter, of type Double, represents the lower limit of the range considered normal
pHighLimit	This parameter, of type Double, represents the higher limit of the range considered normal.
pDeadband	This parameter, of type Double, represents the deadband that is applied to the respective limit before a return to Normal event state is indicated.
pLimitEnable	This parameter, of type BACnetLimitEnable, represents two flags, HighLimitEnable and LowLimitEnable, that separately enable (TRUE) or disable (FALSE) the respective limits applied by the event algorithm. If the value of this parameter is not provided, then both flags shall be set to TRUE (1).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(b) If pCurrentState is NORMAL, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.

(c) If pCurrentState is HIGH_LIMIT, and the HighLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.

(d) Optional: If pCurrentState is HIGH_LIMIT, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(e) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pHighLimit - pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

(f) If pCurrentState is LOW_LIMIT, and the LowLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(g) Optional: If pCurrentState is LOW_LIMIT, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(h) If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pLowLimit + pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-21 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

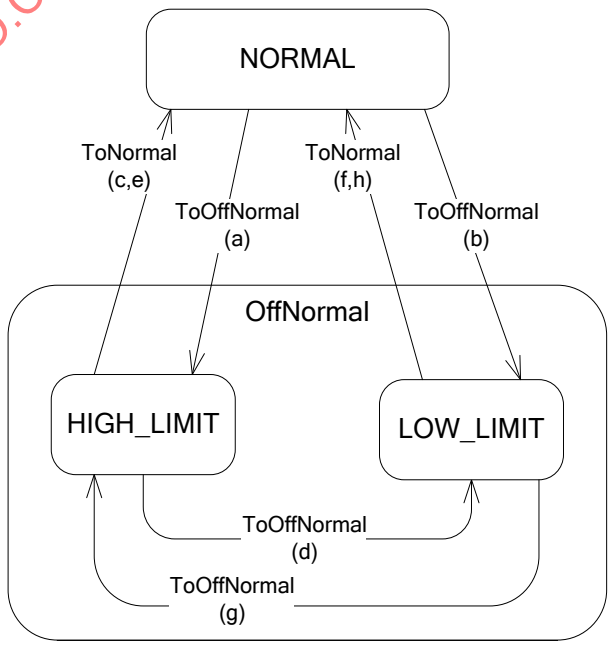


Figure 13-21. Transitions indicated by DOUBLE_OUT_OF_RANGE algorithm

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The notification parameters of this algorithm are:

Exceeding_Value	This notification parameter, of type Double, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Deadband	This notification parameter, of type Double, conveys the value of pDeadband.
Exceeded_Limit	This notification parameter, of type Double, conveys the value of pLowLimit if (a) the new state is LOW_LIMIT, or (b) pCurrentState is LOW_LIMIT and the new state is NORMAL This notification parameter conveys the value of pHighLimit if (a) the new state is HIGH_LIMIT, or (b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

13.3.14 SIGNED_OUT_OF_RANGE Event Algorithm

The SIGNED_OUT_OF_RANGE event algorithm detects whether the monitored value exceeds a range defined by a high limit and a low limit. Each of these limits may be enabled or disabled. If disabled, the normal range has no lower limit or no higher limit respectively. In order to reduce jitter of the resulting event state, a deadband is applied when the value is in the process of returning to the normal range.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type INTEGER, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pLowLimit	This parameter, of type INTEGER, represents the lower limit of the range considered normal.
pHighLimit	This parameter, of type INTEGER, represents the higher limit of the range considered normal.
pDeadband	This parameter, of type Unsigned, represents the deadband that is applied to the respective limit before a return to Normal event state is indicated.
pLimitEnable	This parameter, of type BACnetLimitEnable, represents two flags, HighLimitEnable and LowLimitEnable, that separately enable (TRUE) or disable (FALSE) the respective limits applied by the event algorithm. If the value of this parameter is not provided, then both flags shall be set to TRUE (1).
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(b) If pCurrentState is NORMAL, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(c) If pCurrentState is HIGH_LIMIT, and the HighLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(d) Optional: If pCurrentState is HIGH_LIMIT, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(e) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pHighLimit - pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(f) If pCurrentState is LOW_LIMIT, and the LowLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(g) Optional: If pCurrentState is LOW_LIMIT, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(h) If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pLowLimit + pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-22 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

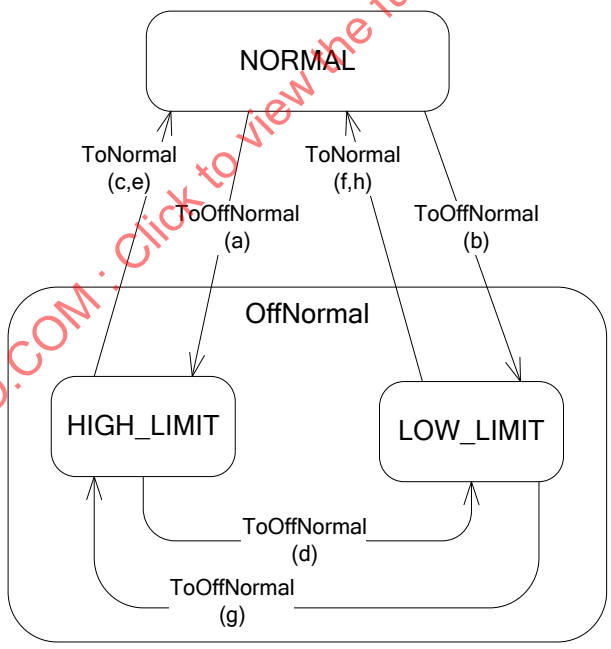


Figure 13-22. Transitions indicated by SIGNED_OUT_OF_RANGE algorithm

The notification parameters of this algorithm are:

Exceeding_Value	This notification parameter, of type INTEGER, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Deadband	This notification parameter, of type Unsigned, conveys the value of pDeadband.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Exceeded_Limit	This notification parameter, of type INTEGER, conveys the value of pLowLimit if (a) the new state is LOW_LIMIT, or (b) pCurrentState is LOW_LIMIT and the new state is NORMAL This notification parameter conveys the value of pHighLimit if (a) the new state is HIGH_LIMIT, or (b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

13.3.15 UNSIGNED_OUT_OF_RANGE Event Algorithm

The UNSIGNED_OUT_OF_RANGE event algorithm detects whether the monitored value exceeds a range defined by a high limit and a low limit. Each of these limits may be enabled or disabled. If disabled, the normal range has no lower limit or no higher limit respectively. In order to reduce jitter of the resulting event state, a deadband is applied when the value is in the process of returning to the normal range.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type Unsigned, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pLowLimit	This parameter, of type Unsigned, represents the lower limit of the range considered normal.
pHighLimit	This parameter, of type Unsigned, represents the higher limit of the range considered normal.
pDeadband	This parameter, of type Unsigned, represents the deadband that is applied to the respective limit before a return to Normal event state is indicated.
pLimitEnable	This parameter, of type BACnetLimitEnable, represents two flags, HighLimitEnable and LowLimitEnable, that separately enable (TRUE) or disable (FALSE) the respective limits applied by the event algorithm. If the value of this parameter is not provided, then both flags shall be set to TRUE (1).
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is provided for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(b) If pCurrentState is NORMAL, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(c) If pCurrentState is HIGH_LIMIT, and the HighLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(d) Optional: If pCurrentState is HIGH_LIMIT, and the LowLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is less than pLowLimit for pTimeDelay, then indicate a transition to the LOW_LIMIT event state.
(e) If pCurrentState is HIGH_LIMIT, and pMonitoredValue is less than (pHighLimit – pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(f) If pCurrentState is LOW_LIMIT, and the LowLimitEnable flag of pLimitEnable is FALSE, then indicate a transition to the NORMAL event state.
(g) Optional: If pCurrentState is LOW_LIMIT, and the HighLimitEnable flag of pLimitEnable is TRUE, and pMonitoredValue is greater than pHighLimit for pTimeDelay, then indicate a transition to the HIGH_LIMIT event state.
(h) If pCurrentState is LOW_LIMIT, and pMonitoredValue is greater than (pLowLimit + pDeadband) for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

If any of the optional conditions are supported, then all optional conditions shall be supported.

Figure 13-23 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

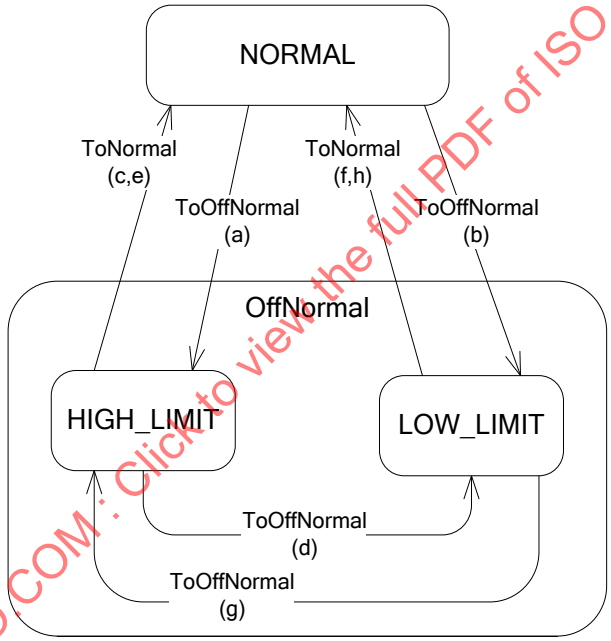


Figure 13-23. Transitions indicated by UNSIGNED_OUT_OF_RANGE algorithm

The notification parameters of this algorithm are:

Exceeding_Value	This notification parameter, of type Unsigned, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Deadband	This notification parameter, of type Unsigned, conveys the value of pDeadband.
Exceeded_Limit	This notification parameter, of type Unsigned, conveys the value of pLowLimit if (a) the new state is LOW_LIMIT, or (b) pCurrentState is LOW_LIMIT and the new state is NORMAL This notification parameter conveys the value of pHighLimit if (a) the new state is HIGH_LIMIT, or (b) pCurrentState is HIGH_LIMIT and the new state is NORMAL

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.3.16 CHANGE_OF_CHARACTERSTRING Event Algorithm

The CHANGE_OF_CHARACTERSTRING event algorithm detects whether the monitored value matches a character string that is listed as an alarm value. Alarm values are of type BACnetOptionalCharacterString, and thus may be a non-empty string, an empty string (of zero length), or a NULL.

A "match" of the monitored value with an alarm value is defined as follows:

(a) If the alarm value is NULL, then it is not considered a match.
(b) If the alarm value string is empty (of zero length), then it is considered a match if and only if the monitored value is also an empty string.
(c) If the alarm value string is not empty, then it is considered a match if the alarm value string appears in any position within the monitored value string. For character-matching purposes, character case shall be significant, and so a match must be an exact match character by character.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type CharacterString, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pAlarmValues	This parameter, of type list of BACnetOptionalCharacterString, represents a list of character strings that are considered alarm values.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue matches any of the values contained in pAlarmValues for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue does not match any of the values contained in pAlarmValues for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(c) If pCurrentState is OFFNORMAL, and pMonitoredValue matches one of the values contained in pAlarmValues that is different from the value that caused the last transition to OFFNORMAL, and remains equal to that value for pTimeDelay, then indicate a transition to the OFFNORMAL event state.

Figure 13-24 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

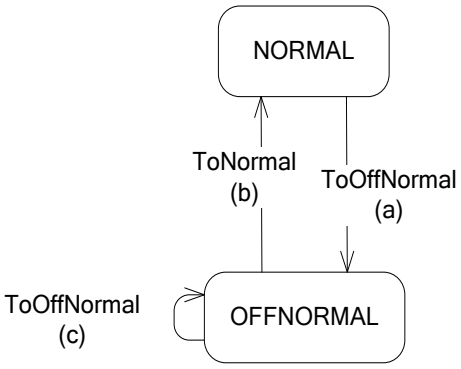


Figure 13-24. Transitions indicated by CHANGE_OF_CHARACTERSTRING algorithm

The notification parameters for this algorithm are:

Changed_Value — This notification parameter, of type CharacterString, conveys the value of pMonitoredValue.

Status_Flags — This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

Alarm_Value — This notification parameter, of type CharacterString, conveys the character string of pAlarmValues related to the event state transition reported:

(a) for transitions to OFFNORMAL, the character string of pAlarmValues that matches pMonitoredValue,

(b) for transitions to NORMAL, the character string of pAlarmValues that match pMonitoredValue at the time of the most recent transition to OFFNORMAL.

13.3.17 NONE Event Algorithm

This event algorithm has no parameters, no conditions, and does not indicate any transitions of event state.

The NONE algorithm is used when only fault detection is in use by an object.

13.3.18 CHANGE_OF_DISCRETE_VALUE Event Algorithm

The CHANGE_OF_DISCRETE_VALUE event algorithm, for monitored discrete values of datatype BOOLEAN, Unsigned, INTEGER, ENUMERATED, CharacterString, OCTET STRING, Date, Time, BACnetObjectIdentifier and BACnetDateTime, detects changes to the monitored value.

For detection of change, the value of the monitored value when a transition to NORMAL is indicated shall be used in evaluation of the conditions until the next transition to NORMAL is indicated. The initialization of the value used in evaluation before the first transition to NORMAL is indicated is a local matter.

The parameters of this event algorithm are:

pCurrentState — This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

pMonitoredValue — This parameter, of type BOOLEAN, Unsigned, INTEGER, ENUMERATED, CharacterString, OCTET STRING, Date, Time, BACnetObjectIdentifier or BACnetDateTime, represents the current value of the monitored property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm, for a monitored value of type BOOLEAN, Unsigned, INTEGER, ENUMERATED, CharacterString, OCTET STRING, Date, Time, BACnetObjectIdentifier or BACnetDateTime, are:

(a) If pCurrentState is NORMAL, and the value of pMonitoredValue changes and remains changed for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

Figure 13-25 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

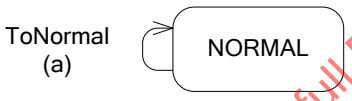


Figure 13-25. Transitions indicated by CHANGE_OF_DISCRETE_VALUE algorithm

The notification parameters of this algorithm are:

New_Value	This notification parameter, of type BOOLEAN, Unsigned, INTEGER, ENUMERATED, CharacterString, OCTET STRING, Date, Time, BACnetObjectIdentifier or BACnetDateTime, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.

13.3.19 CHANGE_OF_TIMER Event Algorithm

The CHANGE_OF_TIMER event algorithm detects whether the monitored value equals a value that is listed as an alarm value. The monitored value shall be of type BACnetTimerState.

The parameters of this event algorithm are:

pCurrentState	This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.
pMonitoredValue	This parameter, of type BACnetTimerState, represents the current value of the monitored property.
pStatusFlags	This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.
pUpdateTime	This parameter, of type BACnetDateTime, represents the date and time of update of the monitored timer state value.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pLastStateChange	This parameter, of type BACnetTimerTransition, represents the last transition of the monitored timer state value.
pInitialTimeout	This parameter, of type Unsigned, represents the initial timeout value in milliseconds at the last transition to the RUNNING timer state.
pExpirationTime	This parameter, of type BACnetDateTime, represents the expiration time that was calculated at the time of application of the algorithm.
pAlarmValues	This parameter, of type List Of BACnetTimerState, represents a list of timer state values that are considered offnormal values.
pTimeDelay	This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.
pTimeDelayNormal	This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is equal to any of the values contained in pAlarmValues for pTimeDelay, then indicate a transition to the OFFNORMAL event state.
(b) If pCurrentState is OFFNORMAL, and pMonitoredValue is not equal to any of the values contained in pAlarmValues for pTimeDelayNormal, then indicate a transition to the NORMAL event state.
(c) If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to one of the values contained in pAlarmValues, and pUpdateTime contains a value that is different from the value present at the last transition to OFFNORMAL, then indicate a transition to the OFFNORMAL event state.

Figure 13-26 depicts those transitions of Figure 13-3 that this event algorithm may indicate:

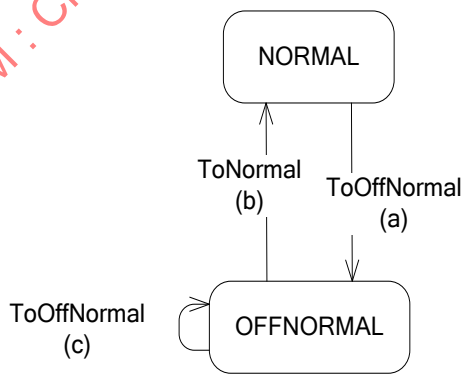


Figure 13-26. Transitions indicated by CHANGE_OF_TIMER algorithm

The notification parameters of this algorithm are:

New_State	This notification parameter, of type BACnetTimerState, conveys the value of pMonitoredValue.
Status_Flags	This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
Update_Time	This notification parameter, of type BACnetDateTime, conveys the value of pUpdateTime.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Last_State_Change	This optional notification parameter, of type BACnetTimerTransition, conveys the value of pLastStateChange, if one is available.
Initial_Timeout	This optional notification parameter, of type Unsigned, conveys the value of pInitialTimeout, if one is available.
Expiration_Time	This optional notification parameter, of type BACnetDateTime, conveys the value of pExpirationTime, if one is available.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.4 Fault Algorithms

Certain object types may optionally support a fault algorithm which has externally visible inputs and is performed as part of the object's reliability-evaluation process. This clause defines the standard fault algorithms. To determine which algorithm is applied by which object type, see the object type definitions in Clause 12.

Table 13-8 lists the fault algorithms that are specified in this standard. The fault algorithms are indicated by the BACnetFaultType value of the same name.

Table 13-8. Standardized Fault Algorithms

Fault Algorithm	Clause
NONE	13.4.1
FAULT_CHARACTERSTRING	13.4.2
FAULT_EXTENDED	13.4.3
FAULT_LIFE_SAFETY	13.4.4
FAULT_LISTED	13.4.8
FAULT_OUT_OF_RANGE	13.4.7
FAULT_STATE	13.4.5
FAULT_STATUS_FLAGS	13.4.6

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Fault algorithms monitor a value and evaluate whether the condition for transition of reliability exists. The result of the evaluation, indicated to the reliability-evaluation process, may be a transition to a new reliability, a transition to the same reliability, or no transition. The final determination of the Reliability property value is the responsibility of the reliability-evaluation process and is subject to additional conditions. See Clause 13.2.

Each of the fault algorithms defines its input parameters, the allowable reliability values, and the conditions for transitions between those values.

When evaluating the monitored value, all conditions defined for the algorithm shall be evaluated in the order as presented for the algorithm. Some algorithms specify optional conditions, marked as "Optional:" Whether or not an implementation uses these conditions is a local matter. If no condition evaluates to true, then no transition shall be indicated to the reliability-evaluation process.

13.4.1 NONE Fault Algorithm

The NONE fault algorithm is a placeholder for the case where no fault algorithm is applied by the object.

This fault algorithm has no parameters, no conditions, and does not indicate any transitions of reliability.

13.4.2 FAULT_CHARACTERSTRING Fault Algorithm

The FAULT_CHRACTERSTRING fault algorithm detects whether the monitored value matches a character string that is listed as a fault value. Fault values are of type BACnetOptionalCharacterString and thus may be a non-empty string, an empty string (of zero length), or a NULL.

A "match" of the monitored value with a fault value is defined as follows:

(a) If the fault value is NULL, then it is not considered a match.
(b) If the fault value string is empty (of zero length), then it is considered a match if and only if the monitored value is also an empty string.
(c) If the fault value string is not empty, then it is considered a match if the fault value string appears in any position within the monitored value string. For character-matching purposes, character case shall be significant, and so a match must be an exact match character by character.

© ISO 2022 – All rights reserved

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pMonitoredValue	This parameter, of type CharacterString, represents the value monitored by this algorithm.
pFaultValues	This parameter, of type list of BACnetOptionalCharacterString, represents a list of character strings that are considered fault values. This parameter shall not contain string values that are present in the pAlarmValues parameter of the CHANGE_OF_CHARACTERSTRING algorithm performed by the same object. NULL values may be present in this parameter regardless of the content of pAlarmValues.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredValue matches one of the values in pFaultValues, then indicate a transition to the MULTI_STATE_FAULT reliability.
(b) If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue does not match any of the values contained in pFaultValues, then indicate a transition to the NO_FAULT_DETECTED reliability.
(c) Optional: If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue matches one of the values contained in pFaultValues that is different from the value that caused the last transition to MULTI_STATE_FAULT, then indicate a transition to the MULTI_STATE_FAULT reliability.

Figure 13-27 depicts the reliability transitions that this fault algorithm may indicate:

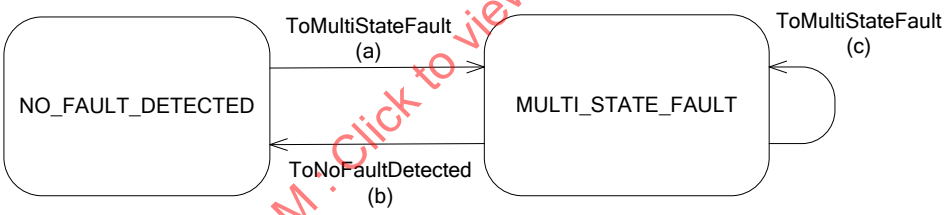


Figure 13-27. Transitions indicated by FAULT_CHARACTERSTRING algorithm

13.4.3 FAULT_EXTENDED Fault Algorithm

The FAULT_EXTENDED fault algorithm detects fault conditions based on a proprietary fault algorithm. The proprietary fault algorithm uses parameters and conditions defined by the vendor. The algorithm is identified by a vendor-specific fault type that is in the scope of the vendor's vendor identification code. The algorithm may, at the vendor's discretion, indicate a new reliability, a transition to the same reliability, or no transition to the reliability-evaluation process.

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pVendorId	This parameter, of type Unsigned16, represents the vendor identification code for the fault type of the vendor-proprietary fault algorithm.
pFaultType	This parameter, of type Unsigned, represents the vendor-proprietary fault algorithm.
pParameters	This parameter represents a sequence of primitive or constructed values whose interpretation is specific to the proprietary fault algorithm.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The conditions evaluated and transitions indicated by this fault algorithm are vendor-specific and are identified by pVendorId and pFaultType.

Figure 13-28 depicts the reliability transitions that this fault algorithm may indicate. The particular unreliable values shown are for illustration only.

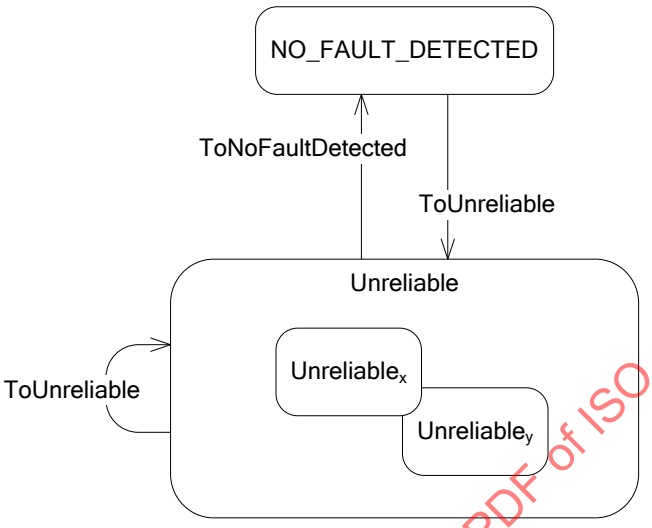


Figure 13-28. Transitions indicated by FAULT_EXTENDED algorithm

13.4.4 FAULT_LIFE_SAFETY Fault Algorithm

The FAULT_LIFE_SAFETY fault algorithm detects whether the monitored value equals a value that is listed as a fault value. The monitored value is of type BACnetLifeSafetyState. If internal operational reliability is unreliable, then the internal reliability takes precedence over evaluation of the monitored value.

In addition, this algorithm monitors a life safety mode value. If reliability is MULTI_STATE_FAULT, then new transitions to MULTI_STATE_FAULT are indicated upon change of the mode value.

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pMonitoredValue	This parameter, of type BACnetLifeSafetyState, represents the value monitored by this algorithm.
pMode	This parameter, of type BACnetLifeSafetyMode, represents the life safety mode value monitored by this algorithm.
pFaultValues	This parameter, of type list of BACnetLifeSafetyState, represents a list of values that are considered fault values. This parameter shall not contain values that are present in the pAlarmValues or pLifeSafetyAlarmValues parameters of the associated CHANGE_OF_LIFE_SAFETY algorithm performed by the same object.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredValue is equal to any of the values in pFaultValues, then indicate a transition to the MULTI_STATE_FAULT reliability.
(b) If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue is not equal to any of the values contained in pFaultValues, then indicate a transition to the NO_FAULT_DETECTED reliability

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

(c) If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue is equal to any of the values contained in pFaultValues, and pMode has changed since the last transition to MULTI_STATE_FAULT, then indicate a transition to the MULTI_STATE_FAULT reliability.
(d) Optional: If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue is equal to one of the values contained in pFaultValues that is different from the value causing the last transition to MULTI_STATE_FAULT, then indicate a transition to the MULTI_STATE_FAULT reliability.

Figure 13-29 depicts the reliability transitions that this fault algorithm may indicate:

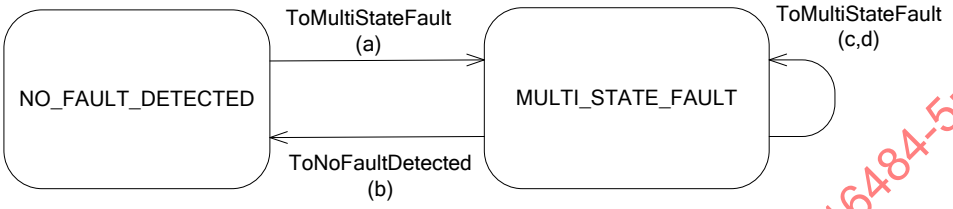


Figure 13-29. Transitions indicated by FAULT_LIFE_SAFETY algorithm

13.4.5 FAULT_STATE Fault Algorithm

The FAULT_STATE fault algorithm detects whether the monitored value equals a value that is listed as a fault value. The monitored value may be of any discrete or enumerated data type, including Boolean. If internal operational reliability is unreliable, then the internal reliability takes precedence over evaluation of the monitored value.

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pMonitoredValue	This parameter is a discrete value that represents the current value of the monitored property. The datatype of the value of this parameter shall be one of the options of BACnetPropertyStates.
pFaultValues	This parameter is a list of discrete values that represent the fault values. The datatype of the values of this parameter and of pMonitoredValue shall be the same.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredValue is equal to any of the values in pFaultValues, then indicate a transition to the MULTI_STATE_FAULT reliability.
(b) If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue is not equal to any of the values contained in pFaultValues, then indicate a transition to the NO_FAULT_DETECTED reliability.
(c) Optional: If pCurrentReliability is MULTI_STATE_FAULT, and pMonitoredValue is equal one of the values contained in pFaultValues that is different from the value that caused the last transition to MULTI_STATE_FAULT, then indicate a transition to the MULTI_STATE_FAULT reliability.

Figure 13-30 depicts the reliability transitions that this fault algorithm may indicate:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

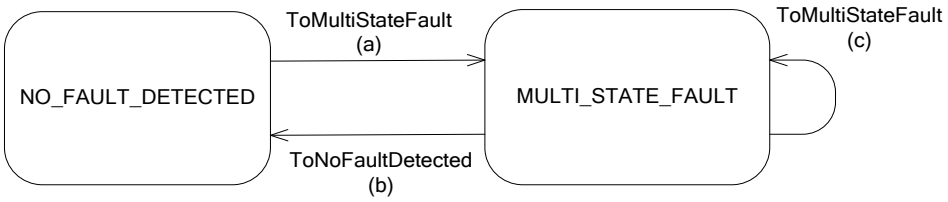


Figure 13-30. Transitions indicated by FAULT_STATE algorithm

13.4.6 FAULT_STATUS_FLAGS Fault Algorithm

The FAULT_STATUS_FLAGS fault algorithm detects whether the monitored status flags are indicating a fault condition.

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pMonitoredValue	This parameter, of type BACnetStatusFlags, is the status flags value monitored by this algorithm.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and the FAULT bit in pMonitoredValue is TRUE, then indicate a transition to the MEMBER_FAULT reliability.
(b) If pCurrentReliability is MEMBER_FAULT, and the FAULT bit in pMonitoredValue is FALSE, then indicate a transition to the NO_FAULT_DETECTED reliability.

Figure 13-31 depicts the reliability transitions that this fault algorithm may indicate:

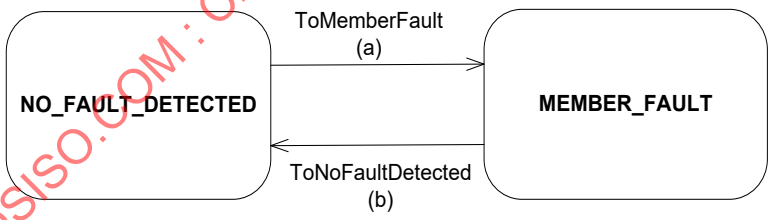


Figure 13-31. Transitions indicated by FAULT_STATUS_FLAGS algorithm

13.4.7 FAULT_OUT_OF_RANGE Fault Algorithm

The FAULT_OUT_OF_RANGE fault algorithm detects whether the monitored value is outside the range of values considered to be normal for the object. At the vendor's discretion, evaluation of the algorithm may be delayed by verification or filtering mechanisms local to the object that applies the fault algorithm which are used to insure that the pMonitoredValue is correct.

The parameters of this fault algorithm are:

pMinimumNormalValue	This parameter, of type REAL, Unsigned, Double or INTEGER, represents the minimum normal value.
pMaximumNormalValue	This parameter, of type REAL, Unsigned, Double or INTEGER, represents the maximum normal value.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

pMonitoredValue	This parameter, of type REAL, Unsigned, Double or INTEGER, is the value monitored by this algorithm.
pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.

For any single instance of the FAULT_OUT_OF_RANGE fault algorithm, all of the numeric parameters, pMinimumNormalValue, pMaximumNormalValue, and pMonitoredValue shall have the same data type and shall match the data type of the object properties assigned to the parameters.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredValue is less than pMinimumNormalValue, then indicate a transition to the UNDER_RANGE reliability.
(b) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredValue is greater than pMaximumNormalValue, then indicate a transition to the OVER_RANGE reliability.
(c) If pCurrentReliability is UNDER_RANGE, and pMonitoredValue is greater than pMaximumNormalValue, then indicate a transition to the OVER_RANGE reliability.
(d) If pCurrentReliability is OVER_RANGE, and pMonitoredValue is less than pMinimumNormalValue, then indicate a transition to the UNDER_RANGE reliability.
(e) If pCurrentReliability is UNDER_RANGE, and pMonitoredValue is greater than or equal to pMinimumNormalValue, and pMonitoredValue is less than or equal to pMaximumNormalValue, then indicate a transition to the NO_FAULT_DETECTED reliability.
(f) If pCurrentReliability is OVER_RANGE, and pMonitoredValue is greater than or equal to pMinimumNormalValue, and pMonitoredValue is less than or equal to pMaximumNormalValue, then indicate a transition to the NO_FAULT_DETECTED reliability.

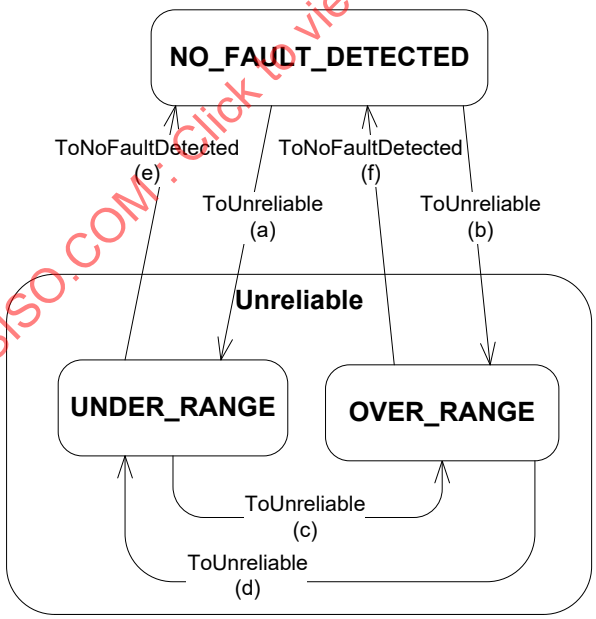


Figure 13-32. Transitions indicated by the FAULT_OUT_OF_RANGE algorithm

13.4.8 FAULT_LISTED Fault Algorithm

The FAULT_LISTED fault algorithm monitors a list of values. A fault is indicated whenever there are values in the list, or the set of values changes. At the vendor's discretion, evaluation of the algorithm may be delayed by verification or filtering mechanisms local to the object that applies the fault algorithm which are used to insure that the pMonitoredList is correct.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The parameters of this fault algorithm are:

pCurrentReliability	This parameter, of type BACnetReliability, represents the current value of the Reliability property of the object that applies the fault algorithm.
pMonitoredList	This parameter, of type BACnetLIST, is the list monitored by this algorithm.

The conditions evaluated by this fault algorithm are:

(a) If pCurrentReliability is NO_FAULT_DETECTED, and pMonitoredList contains at least one element, then indicate a transition to the FAULTS_LISTED reliability.
(b) If pCurrentReliability is FAULTS_LISTED, and pMonitoredList is empty, i.e. contains no element, then indicate a transition to the NO_FAULT_DETECTED reliability.
(c) If pCurrentReliability is FAULTS_LISTED, and pMonitoredList contains a set of values that is different from the set of values that caused the last transition to FAULTS_LISTED, then indicate a transition to the FAULTS_LISTED reliability.

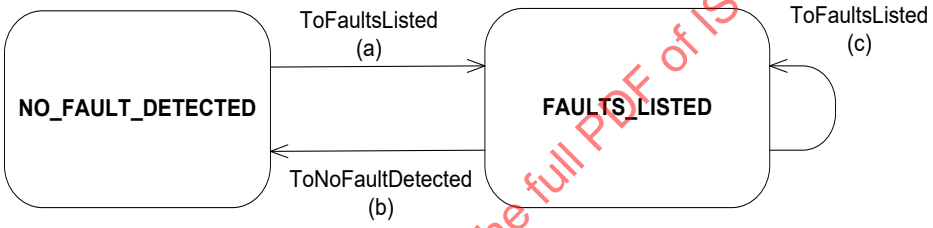


Figure 13-33. Transitions indicated by the FAULT_LISTED algorithm

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

13.5 AcknowledgeAlarm Service

In some systems a device may need to know that an operator has seen the alarm notification. The AcknowledgeAlarm service is used by a notification-client to acknowledge that a human operator has seen and responded to an event notification with 'AckRequired' = TRUE. Ensuring that the acknowledgment actually comes from a person with appropriate authority is a local matter. This service may be used in conjunction with either the ConfirmedEventNotification service or the UnconfirmedEventNotification service.

13.5.1 Structure

The structure of the AcknowledgeAlarm service primitives is shown in Table 13-9. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-9. Structure of AcknowledgeAlarm Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Acknowledging Process Identifier	M	M(=)		
Event Object Identifier	M	M(=)		
Event State Acknowledged	M	M(=)		
Time Stamp	M	M(=)		
Acknowledgment Source	M	M(=)		
Time Of Acknowledgment	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.5.1.1 Argument

This parameter shall convey the parameters for the AcknowledgeAlarm confirmed service request.

13.5.1.1.1 Acknowledging Process Identifier

This parameter, of type Unsigned32, shall specify the 'Process Identifier' parameter that identifies the acknowledging process. The assignment of acknowledging process identifiers is a local matter.

13.5.1.1.2 Event Object Identifier

This parameter, of type BACnetObjectIdentifier, shall specify the 'Event Object Identifier' parameter of the event notification to which this acknowledgment is a response. This is the same object that initiated the event notification that is being acknowledged.

13.5.1.1.3 Event State Acknowledged

This parameter, of type BACnetEventState, shall match the value of the 'To State' from the event notification that is being acknowledged. The 'Event State Acknowledged' matches the 'To State' if they are equal, or if 'Event State Acknowledged' is OFFNORMAL and 'To State' is any "offnormal" state (such as HIGH_LIMIT). This parameter is included so that the remote device that initiated the event notification can ensure that the state being acknowledged is recorded in the Acked_Transitions property of the initiating object.

An 'Event State Acknowledged' of OFFNORMAL shall match any off-normal event state.

13.5.1.1.4 Time Stamp

This parameter, of type BACnetTimeStamp, shall convey the same 'Time Stamp' that was received in the event notification that is being acknowledged by this service. The 'Time Stamp' is used by the recipient of this service request to identify the event notification that is being acknowledged in the case when more than one has been issued with the same 'To State'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.5.1.1.5 Acknowledgment Source

This parameter, of type CharacterString, shall specify the identity of the operator or process that is acknowledging the event notification.

13.5.1.1.6 Time Of Acknowledgment

This parameter, of type BACnetTimeStamp, shall specify the time that the operator or process acknowledged the event notification.

13.5.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded and the alarm is marked as acknowledged.

13.5.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

13.5.1.3.1 Error Type

This parameter consists of two components: (1) 'Error Class' and (2) 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The object does not exist.	OBJECT	UNKNOWN_OBJECT
The object exists but does not support or is not configured for event generation.	OBJECT	NO_ALARM_CONFIGURED
The requesting BACnet device does not have appropriate authorization to Acknowledge this alarm.	SERVICES	SERVICE_REQUEST_DENIED
The timestamp provided in the AcknowledgeAlarm message does not match with the latest timestamp for the transition being acknowledged.	SERVICES	INVALID_TIMESTAMP
The 'Event State Acknowledged' does not match the 'To State' parameter of the original Event Notification message. An 'Event State Acknowledged' of OFFNORMAL shall match any off-normal event state.	SERVICES	INVALID_EVENT_STATE

13.5.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to locate the specified object. If the object exists and if the 'Time Stamp' parameter matches the most recent time for the event being acknowledged, then the bit in the Acked_Transitions property of the object that corresponds to the value of the 'Event State Acknowledged' parameter shall be set to 1, a 'Result(+)' primitive shall be issued, and an event notification with a 'Notify Type' parameter equal to ACK_NOTIFICATION shall be issued. Otherwise, a 'Result(-)' primitive shall be issued. An acknowledgment notification shall use the same type of service (confirmed or unconfirmed) directed to the same recipients to which a confirmed or unconfirmed event notification for the same transition type would be sent. The Time Stamp conveyed in the acknowledgment notification shall not be derived from the Time Stamp of the original event notification, but rather the time at which the acknowledgment notification is generated.

A device shall neither fail to process, nor issue a Result(-), due to an AcknowledgeAlarm service request containing an 'Acknowledgment Source' parameter in an unsupported character set or having invalid characters. In this case, it is a local matter whether the 'Acknowledgment Source' parameter is modified, used as provided, or whether a character string, in a supported character set, of length 0 is used in its place.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.6 ConfirmedCOVNotification Service

The ConfirmedCOVNotification service is used to notify subscribers about changes that may have occurred to the properties of a particular object. Subscriptions for COV notifications are made using the SubscribeCOV service or the SubscribeCOVProperty service.

13.6.1 Structure

The structure of the ConfirmedCOVNotification service primitives is shown in Table 13-10. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-10. Structure of ConfirmedCOVNotification Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Subscriber Process Identifier	M	M(=)		
Initiating Device Identifier	M	M(=)		
Monitored Object Identifier	M	M(=)		
Time Remaining	M	M(=)		
List of Values	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.6.1.1 Argument

This parameter shall convey the parameters for the ConfirmedCOVNotification service request.

13.6.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to identify the process within the COV client that should receive the notification.

13.6.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the ConfirmedCOVNotification service request.

13.6.1.1.3 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of the object that has changed.

13.6.1.1.4 Time Remaining

This parameter, of type Unsigned, shall convey the remaining lifetime of the subscription in seconds. A value of zero shall indicate an indefinite lifetime without automatic cancellation.

13.6.1.1.5 List of Values

This parameter shall convey a list of one or more property values whose contents depend on the type of object being monitored. Table 13-1 summarizes the BACnet standard objects and those property values that shall be returned in the 'List of Values' parameter when those objects are enabled for COV reporting. The property values are returned in the order shown in Table 13-1.

13.6.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.6.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.6.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
No subscription exists for the specified object, property, and process identifier. Devices may ignore this condition and return a BACnet-SimpleACK-PDU.	SERVICES	UNKNOWN_SUBSCRIPTION
The encoding of a value specified in a service parameter is not valid for the datatype of the parameter.	SERVICES	INVALID_DATA_ENCODING

13.6.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall take whatever local actions have been assigned to the indicated COV and issue a 'Result(+)' service primitive. If the service request cannot be executed, a 'Result(-)' service primitive shall be issued indicating the error encountered.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.7 UnconfirmedCOVNotification Service

The UnconfirmedCOVNotification Service is used to notify subscribers about changes that may have occurred to the properties of a particular object, or to distribute object properties of wide interest (such as outside air conditions) to many devices simultaneously without a subscription. Subscriptions for COV notifications are made using the SubscribeCOV service. For unsubscribed notifications, the algorithm for determining when to issue this service is a local matter and may be based on a change of value, periodic updating, or some other criteria.

When issuing COV notifications as a result of a subscription or resubscription, the notification shall always be unicast to the subscriber (See Clause 13.1.2).

13.7.1 Structure

The structure of the UnconfirmedCOVNotification service primitive is shown in Table 13-11. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-11. Structure of UnconfirmedCOVNotification Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Subscriber Process Identifier	M	M(=)
Initiating Device Identifier	M	M(=)
Monitored Object Identifier	M	M(=)
Time Remaining	M	M(=)
List of Values	M	M(=)

13.7.1.1 Argument

This parameter shall convey the parameters for the UnconfirmedCOVNotification service request.

13.7.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to identify the process within the COV client that should receive the notification. The value of zero is reserved for unsubscribed COV.

13.7.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the UnconfirmedCOVNotification service request.

13.7.1.1.3 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of the object that has changed.

13.7.1.1.4 Time Remaining

This parameter, of type Unsigned, shall convey the remaining lifetime of the subscription in seconds. A value of zero shall indicate an indefinite lifetime, without automatic cancellation, or an unsubscribed notification.

13.7.1.1.5 List of Values

This parameter shall convey a list of one or more property values whose contents depend on the type of object being monitored. Table 13-1 summarizes the BACnet standard objects and those property values that shall be returned in the 'List of Values' parameter when those objects are enabled for COV reporting.

13.7.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. Actions taken in response to this notification are a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.8 ConfirmedEventNotification Service

The ConfirmedEventNotification service is used by a notification-server to notify a remote device that an event has occurred and that the notification-server needs a confirmation that the notification has been received. This confirmation means only that the device received the message. It does not imply that a human operator has been notified. A separate AcknowledgeAlarm service is used to indicate that an operator has acknowledged the receipt of the notification if the notification specifies that acknowledgment is required. If multiple recipients are to be notified, a separate invocation of this service shall be used to notify each intended recipient. If a confirmation that a notification was received is not needed, then the UnconfirmedEventNotification may be used.

13.8.1 Structure

The structure of the ConfirmedEventNotification service primitives is shown in Table 13-12. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-12. Structure of ConfirmedEventNotification Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Process Identifier	M	M(=)		
Initiating Device Identifier	M	M(=)		
Event Object Identifier	M	M(=)		
Time Stamp	M	M(=)		
Notification Class	M	M(=)		
Priority	M	M(=)		
Event Type	M	M(=)		
Message Text	U	U(=)		
Notify Type	M	M(=)		
AckRequired	C	C(=)		
From State	C	C(=)		
To State	M	M(=)		
Event Values	C	C(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.8.1.1 Argument

This parameter shall convey the parameters for the ConfirmedEventNotification service request.

13.8.1.1.1 Process Identifier

This parameter, of type Unsigned32, shall convey the identification of the process in the receiving device for which the notification is intended.

13.8.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the ConfirmedEventNotification service request.

13.8.1.1.3 Event Object Identifier

This parameter, of type BACnetObjectIdentifier, shall specify the Object_Identifier of the object that is initiating the notification. This parameter is used by the AcknowledgeAlarm service to identify the object whose notification is being acknowledged.

13.8.1.1.4 Time Stamp

This parameter, of type BACnetTimeStamp, shall convey the current time as determined by the clock in the device issuing the service request. If this device has no clock, then this parameter shall convey a sequence number, of type Unsigned, which indicates the relative ordering of this event notification to all other event notifications issued by this

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

device without regard to their intended recipient. The sequence numbers shall increase monotonically (they may be implemented using modulo arithmetic). A device may have a single sequence number for all event-initiating objects or a separate sequence number for each object.

13.8.1.1.5 Notification Class

This parameter, of type Unsigned, designates the notification class of the event. Definition of the various notification classes is a local matter (see Clauses 13.2, 13.4, and 12.21 for discussion of Notification Class objects).

13.8.1.1.6 Priority

This parameter, of type Unsigned8, shall specify the priority of the event that has occurred. The priority is specified by the Priority property of the Notification Class object associated with this event. The possible range of priorities is 0-255. A lower number indicates a higher priority. The priority and the Network Priority (see Clause 6.2.2) are associated as defined in Table 13-6.

13.8.1.1.7 Event Type

This parameter, of type BACnetEventType, shall specify the type of event that has occurred.

If 'Notify Type' is ALARM or EVENT when the 'To State' parameter is FAULT, then this parameter shall have a value of CHANGE_OF_RELIABILITY. The Event Type CHANGE_OF_RELIABILITY shall be used for reporting a transition from FAULT. Otherwise, this parameter shall have the value associated with the event-initiating object's configured event algorithm.

If 'Notify Type' is ACK_NOTIFICATION when 'To State' is FAULT, the Event Type shall be CHANGE_OF_RELIABILITY. When 'To State' is NORMAL, and the device can determine reporting acknowledgement of a transition from FAULT, the Event Type shall be CHANGE_OF_RELIABILITY. Otherwise, the Event Type shall be the value associated with the event-initiating object's configured event algorithm.

13.8.1.1.8 Message Text

This optional parameter, of type CharacterString, shall convey a string of printable characters. This parameter may be used to convey a message to be logged or displayed, which pertains to the occurrence of the event.

If 'Notify Type' is ALARM or EVENT, the content of the message text shall be derived from the value in the Event_Message_Texts_Config entry corresponding to the transition, if the property exists. Otherwise, the value is a local matter. If the optional property Event_Message_Texts is present in the event-initiating object, the text conveyed in this Message Text parameter shall be stored in the respective field of the Event_Message_Texts array.

If 'Notify Type' is ACK_NOTIFICATION, the value is a local matter.

13.8.1.1.9 Notify Type

This parameter, of type BACnetNotifyType, shall convey whether this notification is an event or an alarm or a notification that someone has acknowledged a previous event notification:

{ALARM, EVENT, ACK_NOTIFICATION}.

13.8.1.1.10 AckRequired

This parameter, of type BOOLEAN, shall convey whether this notification requires acknowledgment (TRUE) or not (FALSE). This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

13.8.1.1.11 From State

This parameter, of type BACnetEventState, shall indicate the Event_State of the object prior to the occurrence of the event that initiated this notification. This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

13.8.1.1.12 To State

This parameter, of type BACnetEventState, shall indicate the Event_State of the object after the occurrence of the event that initiated this notification.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.8.1.1.13 Event Values

This parameter, of type BACnetNotificationParameters, shall convey a set of values relevant to the particular event and whose content depends on the event type. This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

13.8.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.8.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.8.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

13.8.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall take whatever local actions have been assigned to the indicated event occurrence and issue a 'Result(+)' service primitive. If the service request cannot be executed, a 'Result(-)' service primitive shall be issued indicating the encountered error. A device shall neither fail to process, nor issue a Result(-), due to a ConfirmedEventNotification service request containing a 'Message Text' parameter in an unsupported character set or having invalid characters. In this case, the consumption of the 'Message Text' parameter is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.9 UnconfirmedEventNotification Service

The UnconfirmedEventNotification service is used by a notification-server to notify a remote device that an event has occurred. Its purpose is to notify recipients that an event has occurred, but confirmation that the notification was received is not required. Applications that require confirmation that the notification was received by the remote device should use the ConfirmedEventNotification service. The fact that this is an unconfirmed service does not mean it is inappropriate for notification of alarms. Events of type Alarm may require a human acknowledgment that is conveyed using the AcknowledgeAlarm service. Thus, using an unconfirmed service to announce the alarm has no effect on the ability to confirm that an operator has been notified. Any device that executes this service shall support programmable process identifiers to allow broadcast and multicast 'Process Identifier' parameters to be assigned on a per installation basis.

13.9.1 Structure

The structure of the UnconfirmedEventNotification service primitives is shown in Table 13-13. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-13. Structure of UnconfirmedEventNotification Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Process Identifier	M	M(=)
Initiating Device Identifier	M	M(=)
Event Object Identifier	M	M(=)
Time Stamp	M	M(=)
Notification Class	M	M(=)
Priority	M	M(=)
Event Type	M	M(=)
Message Text	U	U(=)
Notify Type	M	M(=)
AckRequired	C	C(=)
From State	C	C(=)
To State	M	M(=)
Event Values	C	C(=)

13.9.1.1 Argument

The 'Argument' parameter shall convey the parameters for the UnconfirmedEventNotification service request.

13.9.1.1.1 Process Identifier

This parameter, of type Unsigned32, shall convey the identification of the process in the receiving device for which the notification is intended.

13.9.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the UnconfirmedEventNotification service request.

13.9.1.1.3 Event Object Identifier

This parameter, of type BACnetObjectIdentifier, shall specify the identifier of the object that is initiating the notification. This parameter is used by the AcknowledgeAlarm service to identify the object whose notification is being acknowledged.

13.9.1.1.4 Time Stamp

This parameter, of type BACnetTimeStamp, shall convey the current time as determined by the clock in the device issuing the service request. If this device has no clock, then this parameter shall convey a sequence number that indicates the relative ordering of this event notification to all other event notifications issued by this device without regard to their intended recipient. The sequence numbers shall increase monotonically (they may be implemented using modulo arithmetic). A device may have a single sequence number for all event-initiating objects or a separate sequence number for each object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.9.1.1.5 Notification Class

This parameter, of type Unsigned, designates the notification class of the event. Definition of the various notification classes is a local matter (see Clause 13.2, 13.4, and 12.21 for discussion of Notification Class objects).

13.9.1.1.6 Priority

This parameter, of type Unsigned8, shall specify the priority of the event that has occurred. The priority is specified by the Priority property of the Notification Class object associated with the event. The possible range of priorities is 0-255. A lower number indicates a higher priority. The priority and the Network Priority (see Clause 6.2.2) are associated as defined in Table 13-6.

13.9.1.1.7 Event Type

This parameter, of type BACnetEventType, shall specify the type of event that has occurred.

If 'Notify Type' is ALARM or EVENT when the 'To State' parameter is FAULT, then this parameter shall have a value of CHANGE_OF_RELIABILITY. The Event Type CHANGE_OF_RELIABILITY shall be used for reporting a transition from FAULT. Otherwise, this parameter shall have the value associated with the event-initiating object's configured event algorithm.

If 'Notify Type' is ACK_NOTIFICATION when 'To State' is FAULT, the Event Type shall be CHANGE_OF_RELIABILITY. When 'To State' is NORMAL, and the device can determine reporting acknowledgement of a transition from FAULT, the Event Type shall be CHANGE_OF_RELIABILITY. Otherwise, the Event Type shall be the value associated with the event-initiating object's configured event algorithm.

13.9.1.1.8 Message Text

This optional parameter, of type CharacterString, shall convey a string of printable characters. This parameter may be used to convey a message to be logged or displayed, which pertains to the occurrence of the event.

If 'Notify Type' is ALARM or EVENT, the content of the message text shall be derived from the value in the Event_Message_Texts_Config property entry corresponding to the transition, if the property exists. Otherwise, the value is a local matter. If the optional property Event_Message_Texts is present in the event-initiating object, the text conveyed in this Message Text parameter shall be stored in the respective field of the Event_Message_Texts array.

If 'Notify Type' is ACK_NOTIFICATION, the value is a local matter.

13.9.1.1.9 Notify Type

This parameter, of type BACnetNotifyType, shall convey whether this notification is an event or an alarm or a notification that someone has acknowledged a previous event notification:

{EVENT, ALARM, ACK_NOTIFICATION}.

13.9.1.1.10 AckRequired

This parameter, of type BOOLEAN, shall convey whether this notification requires acknowledgment (TRUE) or not (FALSE). This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

13.9.1.1.11 From State

This parameter, of type BACnetEventState, shall indicate the state of the object prior to the occurrence of the event that initiated this notification. This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

13.9.1.1.12 To State

This parameter, of type BACnetEventState, shall indicate the state of the object after the occurrence of the event that initiated this notification.

13.9.1.1.13 Event Values

This parameter, of type BACnetNotificationParameters, shall convey a set of values relevant to the particular event and whose content depends on the event type. This parameter shall only be present if the 'Notify Type' parameter is EVENT or ALARM.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.9.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. Actions taken in response to this notification are a local matter. A device shall not fail to process a request due to a 'Message Text' parameter in an unsupported character set or having invalid characters. In this case, the consumption of the 'Message Text' parameter is a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.10 GetAlarmSummary Service

The specification of this service is retained for historical reference so that implementations of client devices have guidance on how to interoperate with older server devices. Otherwise, it is deprecated. It is recommended that execution of this service not be implemented in server devices.

The GetAlarmSummary service is used by a client BACnet-user to obtain a summary of "active alarms." The term "active alarm" refers to BACnet standard objects that have an Event_State property whose value is not equal to NORMAL and a Notify_Type property whose value is ALARM.

13.10.1 Structure

The structure of the GetAlarmSummary service primitives is shown in Table 13-14. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-14. Structure of GetAlarmSummary Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Result(+)			S	S(=)
List of Alarm Summaries			M	M(=)
Object Identifier			M	M(=)
Alarm State			M	M(=)
Acknowledged Transitions			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.10.1.1 Argument

This parameter indicates the GetAlarmSummary confirmed service request.

13.10.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded. A successful result includes the following parameters.

13.10.1.2.1 List of Alarm Summaries

The 'List of Alarm Summaries' shall contain zero or more Alarm Summaries. Each Alarm Summary shall consist of three parameters: 'Object Identifier', 'Alarm State', and 'Acknowledged Transitions'. If the list is of length zero, then this shall be interpreted to mean that there are no active alarms for this device.

13.10.1.2.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall identify the event-initiating object whose Event_State property is not equal to NORMAL and that has a Notify_Type property whose value is ALARM.

13.10.1.2.1.2 Alarm State

This parameter, of type BACnetEventState, indicates the value of the Event_State property of the object.

13.10.1.2.1.3 Acknowledged Transitions

This parameter, of type BACnetEventTransitionBits, indicates the value of the Acked_Transitions property of the object.

13.10.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.10.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

13.10.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall search all event-initiating objects that have an Event_State property not equal to NORMAL and a Notify_Type property whose value is ALARM. Any object that has an Event_Detection_Enable property with a value of FALSE shall be ignored. A positive response containing the alarm summaries for objects found in this search shall be constructed. If no objects are found that meet these criteria, then a list of length zero shall be returned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.11 GetEnrollmentSummary Service

The specification of this service is retained for historical reference so that implementations of client devices have guidance on how to interoperate with older server devices. Otherwise, it is deprecated. It is recommended that execution of this service not be implemented in server devices.

The GetEnrollmentSummary service is used by a client BACnet-user to obtain a summary of event-initiating objects. Several different filters may be applied to define the search criteria.

13.11.1 Structure

The structure of the GetEnrollmentSummary service primitives is shown in Table 13-15. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-15. Structure of GetEnrollmentSummary Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Acknowledgment Filter	M	M(=)		
Enrollment Filter	U	U(=)		
Event State Filter	U	U(=)		
Event Type Filter	U	U(=)		
Priority Filter	U	U(=)		
Notification Class Filter	U	U(=)		
Result(+)			S	S(=)
List of Enrollment Summaries			M	M(=)
Object Identifier			M	M(=)
Event Type			M	M(=)
Event State			M	M(=)
Priority			M	M(=)
Notification Class			U	U(=)
Result(-)			S	S(=)
Error type			M	M(=)

13.11.1.1 Argument

This parameter shall convey the parameters for the GetEnrollmentSummary confirmed service request.

13.11.1.1.1 Acknowledgment Filter

This parameter, of type ENUMERATED, shall provide a means of restricting the event-initiating objects that are to be summarized. The 'Acknowledgment Filter' may take any of three values:

ALL - Shall request that the returned summary contain all event-initiating objects without regard to whether the objects have acknowledgments or not.

ACKED - Shall request that the returned summary contain only those objects for which the Acked_Transitions property has a value of one in every bit position.

NOT-ACKED - Shall request that the returned summary contain only reports for those objects for which the Acked_Transitions property has a value of zero in one or more bit positions.

13.11.1.1.2 Enrollment Filter

This parameter, of type BACnetRecipientProcess, shall provide a means of restricting the set of objects that are to be summarized. Only those objects for which the specified BACnetRecipient and Process Identifier are enrolled to receive notifications, either confirmed or unconfirmed, shall be summarized. In this case, "enrolled" shall mean that an event-initiating object references a Notification Class object containing one or more BACnetDestinations containing the indicated Process Identifier and BACnetRecipient.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If this parameter is omitted, it shall mean that event-initiating objects shall be summarized without regard to enrollment status.

13.11.1.1.3 Event State Filter

This parameter shall provide a means of restricting the set of event-initiating objects that are to be summarized. It may have any of the following values:

{OFFNORMAL, FAULT, NORMAL, ALL, ACTIVE}.

Only those event-initiating objects whose Event_State property matches the value specified in this parameter shall be included. If the value ALL is specified, then all of the event-initiating objects shall be summarized without regard to the value of the Event_State property. If the value ACTIVE is specified, then only those event-initiating objects whose Event_State property has a value other than NORMAL shall be summarized. If this parameter is omitted, a default value of ALL shall be assumed.

13.11.1.1.4 Event Type Filter

This parameter is provided as a means of restricting the summary to only those event-initiating objects that can generate event notifications with an Event_Type equal to the value of this parameter. This parameter may have any legal value of Event_Type as defined in the Event Enrollment object specification. If this parameter is omitted, all event-initiating objects shall be included in the summary without regard to which event types they generate.

13.11.1.1.5 Priority Filter

This parameter consists of two parts, MinPriority and MaxPriority, each of datatype Unsigned8. It provides a means of restricting the summary to only those event-initiating objects that can generate event notifications with a Priority as specified by this parameter. The 'Priority Filter' parameter consists of two parts, MinPriority and MaxPriority. All event-initiating objects, such that MinPriority ≤ Priority ≤ MaxPriority, shall be included in the summary. For the purpose of this filter, the Priority checked by the filter is the Priority associated with the most recent transition. If 'Priority Filter' is omitted, all event-initiating objects shall be summarized without regard to their Priority.

13.11.1.1.6 Notification Class Filter

This parameter, of type Unsigned, provides a means of restricting the summary to only those event-initiating objects that can generate event notifications with a Notification Class equal to the value of this parameter. If 'Notification Class Filter' is omitted, it shall mean that all event-initiating objects shall be summarized without regard to their Notification Class.

13.11.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded. A successful result includes the following parameters.

13.11.1.2.1 List of Enrollment Summaries

The 'List of Enrollment Summaries' shall contain zero or more Enrollment Summaries. Each Enrollment Summary shall consist of up to five parameters: 'Object Identifier', 'Event Type', 'Event State', 'Priority', and, optionally, 'Notification Class'. If the list is of length zero, then this shall be interpreted to mean that there are no event-initiating objects that meet the search criteria specified in the request primitive.

13.11.1.2.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall identify an object meeting the search criteria.

13.11.1.2.1.2 Event Type

This parameter, of type BACnetEventType, indicates the Event_Type that the object can generate.

13.11.1.2.1.3 Event State

This parameter, of type BACnetEventState, indicates the value of the Event_State property of the object.

13.11.1.2.1.4 Priority

This parameter, of type Unsigned8, indicates the priority of notifications generated by the object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.11.1.2.1.5 Notification Class

This optional parameter, of type Unsigned, indicates the class of notifications generated by the object and refers to a Notification Class object that has an object instance number of the same value.

13.11.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.11.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

13.11.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall search for all event-initiating objects that meet the search criteria specified in the request primitive. The search criteria are the logical conjunctions of all of the explicitly stated filters and the default values for any filters that were omitted in the request primitive. Any object that has an Event_Detection_Enable property with a value of FALSE shall be ignored. A positive response containing the enrollment summaries for objects found in this search shall be constructed. If no objects are found that meet these criteria, then a list of length zero shall be returned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.12 GetEventInformation Service

The GetEventInformation service is used by a client BACnet-user to obtain a summary of all "active event states". The term "active event states" refers to all event-initiating objects that

(a) have an Event_State property whose value is not equal to NORMAL, or
(b) have an Acked_Transitions property, which has at least one of the bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

This service is intended to be implemented in all devices that generate event notifications.

13.12.1 Structure

The structure of the GetEventInformation service primitives is shown in Table 13-16. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-16. Structure of GetEventInformation Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Last Received Object Identifier	U	U(=)		
Result(+)				
List of Event Summaries			M	M(=)
Object Identifier			M	M(=)
Event State			M	M(=)
Acknowledged Transitions			M	M(=)
Event Timestamps			M	M(=)
Notify Type			M	M(=)
Event Enable			M	M(=)
Event Priorities			M	M(=)
More Events			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.12.1.1 Argument

This parameter indicates the GetEventInformation confirmed service request.

13.12.1.1.1 Last Received Object Identifier

This optional parameter, of type BACnetObjectIdentifier, shall specify the last Object Identifier received in a preceding GetEventInformation-ACK, if its 'More Events' parameter was TRUE. If this parameter is omitted, the returned summary shall start with the first object meeting the "active event states" criteria. A fixed object processing order is assumed, however the particular order is a local matter. If the Last Received Object Identifier has become invalid in the responding device (i.e. the object is no longer present), the service shall resume if it is possible to determine the object that would have been the successor of the object that is no longer present. Otherwise a Result(-) shall be returned with an 'Error Class' of OBJECT and an 'Error Code' of UNKNOWN_OBJECT.

13.12.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded. A successful result includes the following parameters.

13.12.1.2.1 List of Event Summaries

The 'List of Event Summaries' shall contain zero or more Event Summaries. Each Event Summary shall consist of seven parameters: 'Object Identifier', 'Event State', 'Acknowledged Transitions', 'Event Time Stamps', 'Notify Type', 'Event Enable' and 'Event Priorities'. If the list is of length zero, then this shall be interpreted to mean that there are no event-initiating objects that have active event states in this device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.12.1.2.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall identify the event-initiating object that has an Event_State property whose value is not equal to NORMAL or has an Acked_Transitions property that has at least one of the following bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

13.12.1.2.1.2 Event State

This parameter, of type BACnetEventState, indicates the value of the Event_State property of the object.

13.12.1.2.1.3 Acknowledged Transitions

This parameter, of type BACnetEventTransitionBits, indicates the value of the Acked_Transitions property of the object.

13.12.1.2.1.4 Event Timestamps

This parameter, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the timestamps of the last event notifications for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events.

13.12.1.2.1.5 Notify Type

This parameter, of type BACnetNotifyType, shall convey the value of the Notify_Type property of this object.

13.12.1.2.1.6 Event Enable

This parameter, of type BACnetEventTransitionBits, shall convey the value of the Event_Enable property of the object.

13.12.1.2.1.7 Event Priorities

This parameter, of type BACnetARRAY[3] of Unsigned, shall convey the priorities specified in the Priority property of the associated Notification Class object.

13.12.1.2.2 More Events

This parameter, of type BOOLEAN, shall indicate whether (TRUE) or not (FALSE) more objects exist that meet the active event state criteria of the service request, but that could not be returned in the reply.

13.12.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.12.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

13.12.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall search for all event-initiating objects that do not have an Event_Detection_Enable property with a value of FALSE and that meet the following conditions, beginning with the object following (in whatever internal ordering of objects is used by the responding device) the object specified by the 'Last Received Object Identifier' parameter, if present:

(a) have an Event_State property whose value is not equal to NORMAL, or
(b) have an Acked_Transitions property that has at least one of the following bits (TO_OFFNORMAL, TO_FAULT, TO_NORMAL) set to FALSE.

A positive response containing the event summaries for objects found in this search shall be constructed. If no objects are found that meet these criteria, then a list of length zero shall be returned. As many of the included objects as can be returned within the APDU shall be returned. If more objects exist that meet the criteria but cannot be returned in the APDU, the 'More Events' parameter shall be set to TRUE, otherwise it shall be set to FALSE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.13 LifeSafetyOperation Service

The LifeSafetyOperation service is intended for use in fire, life safety and security systems to provide a mechanism for conveying specific instructions from a human operator to accomplish any of the following objectives:

(a) silence audible or visual notification appliances,
(b) reset latched notification appliances, or
(c) unsilence previously silenced audible or visual notification appliances.

Ensuring that the LifeSafetyOperation request actually comes from a person with appropriate authority is a local matter.

13.13.1 Structure

The structure of the LifeSafetyOperation primitive is shown in Table 13-17. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-17. Structure of LifeSafetyOperation Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Requesting Process Identifier	M	M(=)		
Requesting Source	M	M(=)		
Request	M	M(=)		
Object Identifier	U	U(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.13.1.1 Argument

This parameter shall convey the parameters for the LifeSafetyOperation confirmed service request.

13.13.1.1.1 Requesting Process Identifier

This parameter, of type Unsigned32, specifies an identifying number of significance to the sending device that uniquely identifies the process which initiated the service request. The assignment and meaning of process identifiers shall be a local matter.

13.13.1.1.2 Requesting Source

This parameter, of type CharacterString, specifies the identity of the human operator that initiated the LifeSafetyOperation service request.

13.13.1.1.3 Request

This parameter, of type BACnetLifeSafetyOperation, shall convey the requested operation:

{SILENCE, SILENCE_AUDIBLE, SILENCE_VISUAL, RESET, RESET_ALARM, RESET_FAULT, UNSILENCE, UNSILENCE_AUDIBLE, UNSILENCE_VISUAL}

13.13.1.1.4 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the specific BACnet object to which the life safety request is directed. If this parameter is not present, then all applicable objects within the receiving BACnet device shall be silenced or reset accordingly based on the 'Request' provided.

13.13.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.13.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for the failure shall be specified by the 'Error Type' parameter.

13.13.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
The object does not support LifeSafetyOperation service requests.	OBJECT	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
The object does not support the operation specified in the 'Request' parameter.	OBJECT	VALUE_OUT_OF_RANGE
The requesting BACnet device does not have appropriate authorization for the operation specified in the 'Request' parameter.	SERVICES	SERVICE_REQUEST_DENIED
The operation specified in the 'Request' parameter is invalid in the current state of the selected object.	OBJECT	INVALID_OPERATION_IN_THIS_STATE

13.13.2 Service Procedure

The responding BACnet-user shall first verify the validity of the 'Object Identifier' parameter and return a 'Result(-)' response with the appropriate error class and error code if the 'Request' is invalid or if the 'Object Identifier' parameter is present and specifies an object that is either unknown or does not represent an appropriate request for this object.

If the 'Object Identifier' parameter is not present, then the responding BACnet-user shall attempt to operate all applicable objects in the device based on the 'Request' parameter, and a Result(+) primitive shall be issued.

If the 'Object Identifier' parameter is present, the responding BACnet-user shall attempt to execute the requested operation on the object specified in the 'Object Identifier' parameter based on the 'Request' parameter.If the operation is executed successfully, the responding BACnet-user shall issue a Result(+) primitive.

If the 'Request' parameter conveys a value which is inappropriate for the current state of the object specified in the 'Object Identifier' parameter, e.g, a RESET operation is requested but the object is not ready to be reset or is already reset, then a Result(-) shall be issued.

A device shall not fail to process, or issue a Result(-), upon receiving a LifeSafetyOperation service request containing a 'Requesting Source' parameter in an unsupported character set. In this case, it is a local matter whether the 'Requesting Source' parameter is used as provided or whether a character string, in a supported character set, of length 0 is used in its place.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.14 SubscribeCOV Service

The SubscribeCOV service is used by a COV-client to subscribe for the receipt of notifications of changes that may occur to the properties of a particular object. Certain BACnet standard objects may optionally support COV reporting. If a standard object provides COV reporting, then changes of value of specific properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to one or more subscriber clients. Typically, COV notifications are sent to supervisory programs in BACnet client devices or to operators or logging devices. Proprietary objects may support COV reporting at the implementor's option. The standardized objects that may optionally provide COV support and the change of value algorithms they shall employ are summarized in Table 13-1.

The subscription establishes a connection between the change of value detection and reporting mechanism within the COV-server device and a "process" within the COV-client device. Notifications of changes are issued by the COV-server device when changes occur after the subscription has been established. The ConfirmedCOVNotification and UnconfirmedCOVNotification services are used by the COV-server device to convey change notifications. The choice of confirmed or unconfirmed service is made at the time the subscription is established.

When issuing COV notifications as a result of a subscription or resubscription, the notification shall always be unicast to the subscriber (See Clause 13.1.2).

13.14.1 Structure

The structure of the SubscribeCOV service primitives is shown in Table 13-18. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-18. Structure of SubscribeCOV Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Subscriber Process Identifier	M	M(=)		
Monitored Object Identifier	M	M(=)		
Issue Confirmed Notifications	U	U(=)		
Lifetime	U	U(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.14.1.1 Argument

This parameter shall convey the parameters for the SubscribeCOV confirmed service request.

13.14.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to match future re-subscriptions and cancellations from the subscriber with the COV context that exists within the COV-server device and with confirmed or unconfirmed COV notifications to identify the process within the COV-client that should receive them. The value zero is reserved for unsubscribed COV notifications as described in Clause 13.7.

13.14.1.1.2 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the identifier of the object within the receiving device for which a subscription is desired.

13.14.1.1.3 Issue Confirmed Notifications

This parameter, of type BOOLEAN, shall convey whether the COV-server device shall issue ConfirmedCOVNotifications (TRUE) or UnconfirmedCOVNotifications (FALSE) when changes occur. This parameter, if present, shall indicate a subscription or re-subscription is to occur and that the lifetime shall be refreshed to its initial state. If both the 'Issue Confirmed Notifications' and 'Lifetime' parameters are absent, then this shall indicate a cancellation request. If the 'Lifetime' parameter is present then the 'Issue Confirmed Notifications' parameter shall be present.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.14.1.1.4 Lifetime

This parameter, of type Unsigned, shall convey the desired lifetime of the subscription in seconds. A value of zero shall indicate an indefinite lifetime, without automatic cancellation. A non-zero value shall indicate the number of seconds that may elapse before the subscription shall be automatically cancelled. If both the 'Issue Confirmed Notifications' and 'Lifetime' parameters are absent, then this shall indicate a cancellation request. If the 'Lifetime' parameter is present then the 'Issue Confirmed Notifications' parameter shall be present.

Devices that execute this service shall accept, at a minimum, lifetime values up to and including 28800 seconds (8 hours). Devices may optionally support lifetime values larger than 28800. Devices that initiate this service shall be capable of providing Lifetime values less than or equal to 28800. Support for subscriptions of indefinite lifetime is optional.

13.14.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.14.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.14.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist	OBJECT	UNKNOWN_OBJECT
Specified object does not support COV notifications	OBJECT	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
No context can be created due to resource limitations	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT
The Lifetime parameter is out of the range supported by the device	SERVICES	VALUE_OUT_OF_RANGE

13.14.2 Service Procedure

If neither 'Lifetime' nor 'Issue Confirmed Notifications' are present, then the request shall be considered to be a cancellation. Any COV context that already exists for the same BACnet address contained in the PDU that carries the SubscribeCOV request and has the same 'Subscriber Process Identifier' and 'Monitored Object Identifier' shall be disabled and a 'Result(+)' returned. Cancellations that are issued for which no matching COV context can be found shall succeed as if a context had existed, returning 'Result(+)'.

If the 'Lifetime' parameter is not present but the 'Issue Confirmed Notifications' parameter is present, then a value of zero (indefinite lifetime) shall be assumed for the lifetime. If the 'Issue Confirmed Notifications' parameter is present but the object to be monitored does not support COV reporting, then a 'Result(-)' shall be returned. If the object to be monitored does support COV reporting, then a check shall be made to locate an existing COV context for the same BACnet address contained in the PDU that carries the SubscribeCOV request and has the same 'Subscriber Process Identifier' and 'Monitored Object Identifier'. If an existing COV context is found, then the request shall be considered a re-subscription and shall succeed as if the subscription had been newly created.

If no COV context can be found that matches the request, then a new COV context shall be established that contains the BACnet address from the PDU that carries the SubscribeCOV request and the same 'Subscriber Process Identifier' and 'Monitored Object Identifier'. If no context can be created, then a 'Result(-)' shall be returned.

If a new context is created, or a re-subscription is received, then the COV context shall be initialized and given a lifetime as specified by the 'Lifetime' parameter, if present, or zero if the 'Lifetime' parameter is not present. The subscription shall be automatically cancelled after that many seconds have elapsed unless a re-subscription is received. A lifetime of zero shall indicate that the subscription is indefinite and no automatic cancellation shall occur. In either case, a

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'Result(+)' shall be returned. A ConfirmedCOVNotification or UnconfirmedCOVNotification shall be issued as soon as possible after the successful completion of a subscription or re-subscription request, as specified by the 'Issue Confirmed Notifications' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.15 SubscribeCOVProperty Service

The SubscribeCOVProperty service is used by a COV-client to subscribe for the receipt of notifications of changes that may occur to the properties of a particular object. Any object may optionally support COV reporting. If a standard object provides COV reporting, then changes of value of subscribed-to properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to one or more subscriber clients. Typically, COV notifications are sent to supervisory programs in BACnet client devices or to operators or logging devices.

The subscription establishes a connection between the change of value detection and reporting mechanism within the COV-server device and a "process" within the COV-client device. Notifications of changes are issued by the COV-server device when changes occur after the subscription has been established. The ConfirmedCOVNotification and UnconfirmedCOVNotification services are used by the COV-server device to convey change notifications. The choice of confirmed or unconfirmed service is made at the time the subscription is established. Any object, proprietary or standard, may support COV reporting for any property at the implementor's option.

The SubscribeCOVProperty service differs from the SubscribeCOV service in that it allows monitoring of properties other than those listed in Table 13-1.

When issuing COV notifications as a result of a subscription or resubscription, the notification shall always be unicast to the subscriber (See Clause 13.1.2).

13.15.1 Structure

The structure of the SubscribeCOVProperty service primitives is shown in Table 13-19. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-19. Structure of SubscribeCOVProperty Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Subscriber Process Identifier	M	M(=)		
Monitored Object Identifier	M	M(=)		
Issue Confirmed Notifications	U	U(=)		
Lifetime	U	U(=)		
Monitored Property Identifier	M	M(=)		
COV Increment	U	U(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.15.1.1 Argument

This parameter shall convey the parameters for the SubscribeCOVProperty confirmed service request.

13.15.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to match future re-subscriptions and cancellations from the subscriber with the COV context that exists within the COV-server device and with confirmed or unconfirmed COV notifications to identify the process within the COV-client that should receive them.

13.15.1.1.2 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the identifier of the object within the receiving device that contains the property for which a subscription is desired.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.15.1.1.3 Issue Confirmed Notifications

This parameter, of type BOOLEAN, shall convey whether the COV-server device shall issue ConfirmedCOVNotifications (TRUE) or UnconfirmedCOVNotifications (FALSE) when changes occur. This parameter, if present, shall indicate that a subscription or re-subscription is to occur and that the lifetime shall be refreshed to its initial state. This parameter shall be present if the request is a subscription, or re-subscription, and absent if the request is a cancellation.

13.15.1.1.4 Lifetime

This parameter, of type Unsigned, shall convey the desired lifetime of the subscription in seconds. A value of zero shall not be allowed. A non-zero value shall indicate the number of seconds that may elapse before the subscription shall be automatically cancelled. This parameter shall be present if the request is a subscription, or re-subscription, and absent if the request is a cancellation.

Devices that execute this service shall accept, at a minimum, lifetime values up to and including 28800 seconds (8 hours). Devices may optionally support lifetime values larger than 28800. Devices that initiate this service shall be capable of providing Lifetime values less than or equal to 28800.

13.15.1.1.5 Monitored Property Identifier

This parameter, of type BACnetPropertyReference, shall convey the property identifier and optional array index for which a subscription is desired. If COV reporting is supported for a property that has an array datatype, it is a local matter to determine whether to support COV subscriptions for all elements of the array or only for particular elements in the array.

13.15.1.1.6 COV Increment

This parameter, of type REAL, shall specify the minimum change in the monitored property that will cause a ConfirmedCOVNotification or UnconfirmedCOVNotification to be issued to subscriber COV-clients. This parameter is ignored if the datatype of the monitored property is not numeric. If the monitored property is Present_Value, its datatype is numeric, this parameter is not present, and the monitored object has a COV_Increment property, then the COV increment to use is taken from the COV_Increment property of the monitored object. Otherwise, the COV increment is a local matter. The intent is to allow the subscriber to use a previously established COV increment from another subscription or to allow use of the COV_Increment property in the monitored object.

13.15.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.15.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.15.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist	OBJECT	UNKNOWN_OBJECT
Specified property does not exist	PROPERTY	UNKNOWN_PROPERTY
Specified object does not support COV notifications	OBJECT	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
Specified property does not support COV notifications	PROPERTY	NOT_COV_PROPERTY
No context can be created due to resource limitations	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT
The Lifetime parameter is outside the range supported by the device	SERVICES	VALUE_OUT_OF_RANGE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.15.2 Service Procedure

The absence of the 'Lifetime' and 'Issue Confirmed Notifications' indicates that the request is a cancellation. Any COV context that already exists for the same BACnet address contained in the PDU that carries the SubscribeCOVProperty request and has the same 'Subscriber Process Identifier', 'Monitored Object Identifier' and 'Monitored Property Identifier' shall be disabled and a 'Result(+)' returned. Cancellations that are issued for which no matching COV context can be found shall succeed as if a context had existed, returning 'Result(+)'. If an existing COV context is found, it shall be removed from the Active_COV_Subscriptions property in the Device object.

If the 'Issue Confirmed Notifications' parameter is present but the property to be monitored does not support COV reporting, then a 'Result(-)' shall be returned. If the property to be monitored does support COV reporting, then a check shall be made to locate an existing COV context for the same BACnet address contained in the PDU that carries the SubscribeCOVProperty request and has the same 'Subscriber Process Identifier', 'Monitored Object Identifier' and 'Monitored Property Identifier'. If an existing COV context is found, then the request shall be considered a re-subscription and shall succeed as if the subscription had been newly created.

If no COV context can be found that matches the request, then a new COV context shall be established that contains the BACnet address from the PDU that carries the SubscribeCOVProperty request and the same 'Subscriber Process Identifier', 'Monitored Object Identifier' and 'Monitored Property Identifier'. The new context shall be included in the Active_COV_Subscriptions property of the Device object. If no context can be created, then a 'Result(-)' shall be returned.

If a new context is created, or a re-subscription is received, then the COV context shall be initialized and given a lifetime as specified by the 'Lifetime' parameter. The subscription shall be automatically cancelled after that many seconds have elapsed unless a re-subscription is received. A 'Result(+)' shall be returned and a ConfirmedCOVNotification or UnconfirmedCOVNotification shall be issued as soon as possible after the successful completion of a subscription or re-subscription request, as specified by the 'Issue Confirmed Notifications' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.16 SubscribeCOVPropertyMultiple Service

The SubscribeCOVPropertyMultiple service is used by a COV-client to subscribe for the receipt of notifications of changes that may occur to multiple properties of multiple objects. Any object may optionally support COV reporting. If an object provides COV reporting, then changes of value of subscribed-to properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to one or more subscriber clients. Typically, COV notifications are sent to supervisory programs in BACnet client devices or to operators or logging devices.

The subscription establishes a connection between the change of value detection and reporting mechanism within the COV-server device and a "process" within the COV-client device. Notifications of changes are issued by the COV-server device when changes occur after the subscription has been established. The ConfirmedCOVNotificationMultiple and UnconfirmedCOVNotificationMultiple services are used by the COV-server device to convey change notifications. The choice of confirmed or unconfirmed service is made by the COV-client at the time the subscription is established.

The SubscribeCOVPropertyMultiple service differs from the SubscribeCOVProperty service in that the former allows multiple properties to be monitored via a single subscription request. For establishing or cancelling subscriptions, the effect of the SubscribeCOVPropertyMultiple service is similar to a series of independent SubscribeCOVProperty requests, one difference being that subscription via SubscribeCOVPropertyMultiple results in the use of the ConfirmedCOVNotificationMultiple and UnconfirmedCOVNotificationMultiple services to convey COV notifications of only those properties whose values changed.

The SubscribeCOVPropertyMultiple service also supports requests for timestamped data, for situations where the client requires the time associated with the value change.

When issuing COV notifications as a result of a subscription or resubscription, the notification shall always be unicast to the subscriber (See Clause 13.1.2).

13.16.1 Structure

The structure of the SubscribeCOVPropertyMultiple service primitives is shown in Table 13-20. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-20. Structure of SubscribeCOVPropertyMultiple Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Subscriber Process Identifier	M	M(=)		
Issue Confirmed Notifications	M	M(=)		
Lifetime	U	U(=)		
Max Notification Delay	U	U(=)		
List of COV Subscription Specifications	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			S	S(=)
First Failed Subscription			S	S(=)

13.16.1.1 Argument

This parameter shall convey the parameters for the SubscribeCOVPropertyMultiple confirmed service request.

13.16.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to match future re-subscriptions and cancellations from the subscriber with the COV-multiple context that exists within the COV-server device and with confirmed or unconfirmed COV notifications to identify the process within the COV-client that should receive them.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.16.1.1.2 Issue Confirmed Notifications

This parameter, of type BOOLEAN, shall convey the form of notification, i.e. whether the COV-server device shall issue ConfirmedCOVNotificationMultiple requests (TRUE) or UnconfirmedCOVNotificationMultiple requests (FALSE) when changes occur. This parameter shall be used to match future re-subscriptions and cancellations from the subscriber with the COV-multiple context that exists within the COV-server device.

13.16.1.1.3 Lifetime

This parameter, of type Unsigned, shall convey the desired lifetime of the subscription in seconds. A value of zero shall not be allowed. A non-zero value shall indicate the number of seconds that may elapse before all subscriptions shall be automatically canceled for the recipient and form of notification.

Devices that execute this service shall accept, at a minimum, lifetime values up to and including 28800 seconds (8 hours). Devices may optionally support lifetime values larger than 28800. Devices that initiate this service shall be capable of providing Lifetime values less than or equal to 28800.

This parameter shall be present if the request is a subscription, or re-subscription, and absent if the request is a cancellation.

13.16.1.1.4 Max Notification Delay

This parameter, of type Unsigned, shall convey the maximum notification delay for the subscription in seconds. This parameter indicates the maximum number of seconds that may elapse before a notification is issued if changes of properties occurred for which the 'Timestamped' parameter is TRUE and which were queued up for notification. The value of this parameter shall be less than the value of the Lifetime parameter.

The range for this parameter shall be 0 to 3600 seconds.

This parameter shall be present if the request is a subscription, or re-subscription, and absent if the request is a cancellation.

13.16.1.1.5 List of COV Subscription Specifications

This parameter shall consist of a list of zero or more 'COV Subscription Specifications'. Each specification shall consist of two parameters: (1) an 'Object Identifier' and (2) a 'List of COV References'. See Clause 13.16.3.1.

This parameter may be an empty list for re-subscriptions that only update the lifetime of the COV-multiple subscription or for a cancellation of a subscription.

13.16.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.16.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter in the case of a general error in executing the service, or by the 'First Failed Subscription' parameter if an error occurred in processing the 'List of COV Subscription Specifications' parameter.

13.16.1.3.1 Error Type

This parameter shall be used to report general service execution errors not related to a specific COV subscription specification of the request. See Clause 13.16.3.2.1.

13.16.1.3.2 First Failed Subscription

This parameter shall consist of three parameters: (1) an 'Object Identifier', (2) a 'Property Reference', and (3) an 'Error' parameter, indicating the reason for failure of the subscription. See Clause 13.16.3.2.2.

13.16.2 Service Procedure

If the parameters 'Lifetime' and 'Max Notification Delay' are not present, then the request shall be considered to be a cancellation. All COV subscription specifications that already exist for the same recipient and form of notification contained in the PDU that carries the SubscribeCOVPropertyMultiple request (i.e. a COV-multiple context exists) and that appear in the list of 'COV Subscription Specifications' shall be removed from the respective COV-multiple context

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

and a 'Result(+)' returned. If no COV subscription specifications remain in a COV-multiple context, the COV-multiple context is removed in its entirety. If the list of COV subscription specifications conveyed in a cancellation is empty, the COV-multiple context shall be removed. Cancellations that are issued for which no matching COV-multiple context, or no COV subscription specifications in an existing COV-multiple context can be found shall succeed as if a COV-multiple context or COV subscription specification had existed and was removed.

If the 'Lifetime' and 'Max Notification Delay' parameters are present, the COV subscriptions and the COV references present in the request shall be processed in the order specified in the request. If any of the properties to be monitored do not support COV-multiple reporting, then a 'Result(-)' shall be returned. If the properties to be monitored do support COV-multiple reporting, then a check shall be made to locate an existing COV-multiple context for the same recipient and form of notification contained in the PDU that carries the SubscribeCOVPropertyMultiple request. If an existing COV-multiple context is found, then this request shall be considered a re-subscription and shall succeed as if the subscription had been newly created. For COV subscription specifications conveyed in the request for which no COV subscription specifications exist in the COV-multiple context, respective COV subscription specifications shall be added to the COV-multiple context.

If the 'Lifetime' and 'Max Notification Delay' parameters are present, and no COV-multiple context can be found that matches the source BACnet address, the recipient process and the form of notification conveyed in the subscription request, then a new COV-multiple context shall be established that contains the BACnet address from the PDU that carries the SubscribeCOVPropertyMultiple request, the 'Subscriber Process Identifier', the 'Issue Confirmed Notifications' flag, the 'Lifetime', and the 'Max Notification Delay' timeout. All COV subscription specifications conveyed in the request shall be added to the COV-multiple context. The new COV-multiple context shall be included in the Active_COV_Multiple_Subscriptions property of the Device object. If no COV-multiple context can be created, then a 'Result(-)' shall be returned.

If a new COV-multiple context is created, or re-subscribed, then it shall be initialized and given a lifetime as specified by the 'Lifetime' parameter. The subscription shall be automatically canceled and the COV-multiple context shall be removed after 'Lifetime' seconds have elapsed unless a re-subscription is received.

A 'Result(+)' shall be returned and a ConfirmedCOVNotificationMultiple or one or multiple UnconfirmedCOVNotificationMultiple notifications shall be issued for each COV subscription specification specified in the SubscribeCOVPropertyMultiple request as soon as possible after the successful completion of a subscription or re-subscription request, as specified by the 'Issue Confirmed Notifications' parameter.

A Result(-) with parameter 'Error Type' shall be returned in case of a failure in processing the request before any COV subscription specification and COV reference has been processed. The Result(-) shall convey the 'Error Class' and 'Error Code' indicating the error occurred. No COV-multiple notifications shall be issued in this case.

A 'Result(-)' with parameter 'First Failed Subscription' shall be returned in the case that an error occurred in processing a particular COV subscription specification and COV reference of the request. Remaining COV references, if any, and the remaining COV subscription specifications shall not be processed. The 'Result(-)' shall convey the object identifier of the failing COV subscription specification in parameter 'Monitored Object Identifier', the property reference of the failing COV reference in parameter 'Monitored Property Reference', and the error type of the error that occurred for this COV subscription specification in parameter 'Error Type'. COV-multiple notifications shall be issued for those COV subscription specifications and COV references already processed successfully, as defined for the 'Result(+)' case.

13.16.3 Parameters Referenced by the SubscribeCOVPropertyMultiple Service

The following parameters appear in the SubscribeCOVPropertyMultiple service primitives.

13.16.3.1 COV Subscription Specification Parameter

The 'COV Subscription Specification' parameter is shown in Table 13-21. The terminology and symbology used in this table are explained in Clause 5.6.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 13-21. Structure of 'COV Subscription Specification' Parameter

Parameter Name	Req	Ind	Datatype
Monitored Object	M	M(=)	BACnetObjectIdentifier
List of COV References	M	M(=)	
Monitored Property	M	M(=)	BACnetPropertyReference
COV Increment	U	U(=)	REAL
Timestamped	M	M(=)	BOOLEAN

13.16.3.1.1 Monitored Object

This parameter, of type BACnetObjectIdentifier, shall convey the identifier of the object within the receiving device that contains one or more properties for which a subscription is desired.

13.16.3.1.2 List of COV References

This parameter shall be a list of one or more 'COV References', each of which corresponds directly to a specific property of the object identified above.

Each 'COV Reference' shall convey the property identifier and optional array index for which a subscription is desired. If COV reporting is supported for a property that has an array datatype, it is a local matter to determine whether to support COV subscriptions for all elements of the array or only for particular elements in the array. The property identifier in a 'COV Reference' shall not be one of the special property identifiers ALL, REQUIRED, or OPTIONAL.

13.16.3.1.2.1 Monitored Property

This parameter, of type BACnetPropertyReference, shall convey the property identifier and optional array index for which a subscription is desired.

13.16.3.1.2.2 COV Increment

The 'COV Increment' parameter, of type REAL, shall specify the minimum change in the monitored property that will cause a COV notification to be queued up or issued to subscriber COV-clients. This parameter shall be ignored if the value of the monitored property does not represent a numeric quantity or a BACnetARRAY of numeric quantities.

If the monitored property is Present_Value, its value represents a numeric quantity, the 'COV_Increment' parameter is not present, and the monitored object has a COV_Increment property, then the COV increment to use is taken from the COV_Increment property of the monitored object. Otherwise, if this parameter is not present, the value used for the COV increment shall be a local matter. The intent is to allow the subscriber to use a previously established COV increment from another subscription or to allow use of the COV_Increment property in the monitored object.

13.16.3.1.2.3 Timestamped

This parameter, of type BOOLEAN, shall convey whether the COV-server device shall timestamp (TRUE) or not (FALSE) the changes that occur in the monitored property. If TRUE, the COV-server shall queue up all changes of the monitored property until a COV-multiple notification is sent to the subscriber, and shall, in this notification, provide all changes that were queued up to be sent in the COV-server.

If FALSE, the COV-server shall initiate a COV-multiple notification when the value of the property changes, by 'COV-Increment' if one is specified, and include all pending timestamped changes in the notification that are queued up for the subscriber.

13.16.3.2 Parameters Referenced by the Result(-) Error Return

The following parameters appear in the Result(-) error return, as shown in Table 13-20.

13.16.3.2.1 Error Type

This parameter shall consist of two components: (1) 'Error Class' and (2) 'Error Code'. See Clause 18.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
No COV-multiple context can be created due to resource limitations	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT
The 'Lifetime' parameter is outside of the range supported by the device	SERVICES	VALUE_OUT_OF_RANGE
The 'Max Notification Delay' parameter is outside of the range supported by the device.	SERVICES	VALUE_OUT_OF_RANGE
The 'Max Notification Delay' parameter is greater than the 'Lifetime' parameter.	SERVICES	VALUE_OUT_OF_RANGE

13.16.3.2.2 First Failed Subscription

The 'First Failed Subscription' parameter is shown in Table 13-22. This parameter identifies the first failed subscription, corresponding to a specific object and property identified in the request for which the subscription failed. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-22. Structure of 'First Failed Subscription' Parameter

Parameter Name	Rsp	Cnf	Datatype
Monitored Object Identifier	M	M(=)	BACnetObjectIdentifier
Monitored Property Reference	M	M(=)	BACnetPropertyReference
Error Type	M	M(=)	Error

13.16.3.2.2.1 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the identifier of an object containing one or more properties for which a subscription failed.

13.16.3.2.2.2 Monitored Property Reference

This parameter, of type BACnetPropertyReference, shall convey the property identifier and array index (if one appeared in the corresponding request) for which a subscription failed.

13.16.3.2.2.3 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations in processing an individual subscription for a property are as follows:

Situation	Error Class	Error Code
Specified object does not exist	OBJECT	UNKNOWN_OBJECT
Specified property does not exist	PROPERTY	UNKNOWN_PROPERTY
Specified object does not support COV-multiple notifications	OBJECT	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
Specified property does not support COV-multiple notifications	PROPERTY	NOT_COV_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
No COV subscription specification can be created in the COV-multiple context due to resource limitations	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.17 ConfirmedCOVNotificationMultiple Service

The ConfirmedCOVNotificationMultiple service is used to notify subscribers about changes that may have occurred to one or more properties of one or more objects. Subscriptions for ConfirmedCOVNotificationMultiple are made using the SubscribeCOVPropertyMultiple service.

13.17.1 Structure

The structure of the ConfirmedCOVNotificationMultiple service primitives is shown in Table 13-23. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-23. Structure of ConfirmedCOVNotificationMultiple Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Subscriber Process Identifier	M	M(=)		
Initiating Device Identifier	M	M(=)		
Time Remaining	M	M(=)		
Timestamp	U	U(=)		
List of COV Notifications	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.17.1.1 Argument

This parameter shall convey the parameters for the ConfirmedCOVNotificationMultiple service request.

13.17.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to identify the process within the COV client that should receive the notification.

13.17.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the ConfirmedCOVNotificationMultiple service request.

13.17.1.1.3 Time Remaining

This parameter, of type Unsigned, shall convey the remaining lifetime of the COV-multiple subscription, in seconds.

13.17.1.1.4 Timestamp

This parameter, of type BACnetDateTime, shall convey the date and time of the last change conveyed in the notification. This parameter shall be present if, and only if, any of the COV notifications conveyed in the 'List of COV Notifications' parameter contain the 'Time of Change' parameter.

13.17.1.1.5 List of COV Notifications

This parameter shall be a list of one or more 'COV Notifications', each of which corresponds directly to a specific property of the object identified and whose value has changed, resulting in the COV notification to be issued. See Clause 13.17.3.1.

The ConfirmedCOVNotificationMultiple request may convey COV notifications from any objects and properties that are subscribed to by the same BACnet address, the same Process Identifier, and requested to issue ConfirmedCOVNotificationMultiple notifications. These subscriptions may have been established through one or multiple SubscribeCOVPropertyMultiple requests. The ConfirmedCOVNotificationMultiple request shall also contain all timestamped changes of properties that were queued up since the last notification sent to the subscriber specified by the respective COV-Multiple context.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.17.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

13.17.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

13.17.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
No subscription exists for one of the specified objects, properties, and process identifier. Devices may ignore this condition and return a BACnet-SimpleACK-PDU.	SERVICES	UNKNOWN_SUBSCRIPTION
The encoding of a value specified in a service parameter is not valid for the datatype of the parameter.	SERVICES	INVALID_DATA_ENCODING

13.17.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall take whatever local actions have been assigned to the indicated COV subscriptions and issue a 'Result(+)' service primitive. If the service request fails in its entirety or for any COV notification conveyed, a 'Result(-)' service primitive may be issued indicating the error encountered.

13.17.3 Parameters Referenced by the ConfirmedCOVNotificationMultiple Service

The following parameters appear in the ConfirmedCOVNotificationMultiple service primitives.

13.17.3.1 COV Notification Parameter

The 'COV Notification' parameter is shown in Table 13-24. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-24. Structure of 'COV Notification' Parameter

Parameter Name	Req	Ind	Datatype
Monitored Object Identifier	M	M(=)	BACnetObjectIdentifier
List of Values	M	M(=)	
Property Identifier	M	M(=)	BACnetPropertyIdentifier
Property Array Index	U	U(=)	Unsigned
Property Value	M	M(=)	ANY
Time of Change	U	U(=)	Time

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

13.17.3.1.1 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of an object with one or more properties subscribed via SubscribeCOVPropertyMultiple and whose values have changed.

13.17.3.1.2 List of Values

This parameter shall convey a list of one or more property values that have changed, possibly conveyed as timestamped data. This parameter may include multiple entries for the same property if the 'Timestamped' flag is TRUE in the respective COV subscription specification and multiple changes of that property have been queued up since the last notification.

© ISO 2022 – All rights reserved

13.17.3.1.2.1 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall convey the property identifier of the property.

13.17.3.1.2.2 Property Array Index

If, and only if, the subscription corresponding to this notification specified an array index and the property is of datatype array, this parameter, of type Unsigned, shall convey that array index.

13.17.3.1.2.3 Property Value

This parameter, of type ANY, shall convey the new value of the property.

13.17.3.1.2.4 Time of Change

This parameter, of type Time, shall be present if, and only if, the SubscribeCOVPropertyMultiple request was made with its 'Timestamped' parameter set to TRUE in the COV subscription specification for the property and shall convey the local time when the data value changed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.18 UnconfirmedCOVNotificationMultiple Service

The UnconfirmedCOVNotificationMultiple service is used to notify subscribers about changes that may have occurred to one or more properties of one or more objects, or to distribute object properties of wide interest (such as outside air conditions) to many devices simultaneously without a subscription. Subscriptions for UnconfirmedCOVNotificationMultiple are made using the SubscribeCOVPropertyMultiple service. For unsubscribed notifications, the algorithm for determining when to issue this service is a local matter and may be based on a change of value, periodic updating, or some other criteria.

When issuing COV notifications as a result of a subscription or resubscription, the notification shall always be unicast to the subscriber (See Clause 13.1.2).

13.18.1 Structure

The structure of the UnconfirmedCOVNotificationMultiple service primitives is shown in Table 13-25. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-25. Structure of UnconfirmedCOVNotificationMultiple Service Primitives

Parameter Name	Req	Ind
Argument	M	M(=)
Subscriber Process Identifier	M	M(=)
Initiating Device Identifier	M	M(=)
Time Remaining	M	M(=)
Timestamp	U	U(=)
List of COV Notifications	M	M(=)

13.18.1.1 Argument

This parameter shall convey the parameters for the UnconfirmedCOVNotificationMultiple service request.

13.18.1.1.1 Subscriber Process Identifier

This parameter, of type Unsigned32, shall convey a numeric "handle" meaningful to the subscriber. This handle shall be used to identify the process within the COV client that should receive the notification. The value zero is reserved for unsubscribed COV notifications.

13.18.1.1.2 Initiating Device Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Device Object_Identifier of the device that initiated the UnconfirmedCOVNotificationMultiple service request.

13.18.1.1.3 Time Remaining

This parameter, of type Unsigned, shall convey the remaining lifetime of the COV-multiple subscription, in seconds. A value of zero shall be used in unsubscribed notifications.

13.18.1.1.4 Timestamp

This parameter, of type BACnetDateTime, shall convey the date and time of the last change conveyed in the notification. This parameter shall be present if, and only if, any of the COV notifications conveyed in the 'List of COV Notifications' contain the 'Time of Change' value.

13.18.1.1.5 List of COV Notifications

This parameter shall be a list of one or more 'COV Notifications', each of which corresponds directly to a specific property of the object identified and whose value has changed, resulting in the COV notification to be issued. See Clause 13.18.3.1

The UnconfirmedCOVNotificationMultiple request may convey COV notifications from any objects and properties that are subscribed to by the same BACnet address, the same Process Identifier, and requesting to issue UnconfirmedCOVNotificationMultiple notifications. These subscriptions may have been established through one or multiple SubscribeCOVPropertyMultiple requests. The UnconfirmedCOVNotificationMultiple request shall also contain all timestamped changes of properties that were queued up since the last notification sent to the subscriber specified by the respective COV-multiple context.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the number of changes exceeds the number of changes conveyable in a single notification request, the COV-server shall initiate as many UnconfirmedCOVNotificationMultiple requests that are required to notify all changes.

13.18.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. Actions taken in response to this notification are a local matter.

13.18.3 Parameters Referenced by the UnconfirmedCOVNotificationMultiple Service

The following parameters appear in the UnconfirmedCOVNotificationMultiple service primitives.

13.18.3.1 COV Notification Parameter

The 'COV Notification' parameter is shown in Table 13-26. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-26. Structure of 'COV Notification' Parameter

Parameter Name	Req	Ind	Datatype
Monitored Object Identifier	M	M(=)	BACnetObjectIdentifier
List of Values	M	M(=)	
Property Identifier	M	M(=)	BACnetPropertyIdentifier
Property Array Index	U	U(=)	Unsigned
Property Value	M	M(=)	ANY
Time of Change	U	U(=)	Time

13.18.3.1.1 Monitored Object Identifier

This parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of an object with one or more properties subscribed via SubscribeCOVPropertyMultiple and whose values have changed.

13.18.3.1.2 List of Values

This parameter shall convey a list of one or more property values that have changed, possibly conveyed as timestamped data. This parameter may include multiple entries for the same property if the 'Timestamped' flag is TRUE in the respective COV subscription specification and multiple changes of that property have been queued up since the last notification.

13.18.3.1.2.1 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall convey the property identifier of the property.

13.18.3.1.2.2 Property Array Index

If, and only if, the subscription corresponding to this notification specified an array index and the property is of datatype array, this parameter, of type Unsigned, shall convey that array index.

13.18.3.1.2.3 Property Value

This parameter, of type ANY, shall convey the new value of the property.

13.18.3.1.3 Time of Change

This parameter, of type Time, shall be present if, and only if, the SubscribeCOVPropertyMultiple request for the property was made with its 'Timestamped' parameter set to TRUE in the COV subscription specification for the property and shall convey the local time when the data value changed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.19 AuditLogQuery

The AuditLogQuery service provides efficient access to Audit Log records. The service allows for a few common queries on Audit Logs.

The service allows devices to find and retrieve audit information without reading and processing a complete audit log.

The queries currently facilitated by the AuditLogQuery service are:

Audit Query By Target - Retrieve the audit records for the specified target device, object, and property. Records are returned in reverse sequence number order (newest to oldest).

Audit Query By Source - Retrieve all audited actions initiated by the specified operation source. Records are returned in the reverse sequence number order (newest to oldest).

13.19.1 AuditLogQuery Service Structure

The structure of the AuditLogQuery service primitive is shown in Table 13-27. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-27. Structure of AuditLogQuery Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Audit Log	M	M(=)		
Query Parameters	M	M(=)		
Start At Sequence Number	U	U(=)		
Requested Count	M	M(=)		
Result(+)			S	S(=)
Audit Log			M	M(=)
Records			M	M(=)
No More Items			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.19.1.1 Argument

This parameter shall convey the parameters for the AuditLogQuery confirmed service request.

13.19.1.1.1 Audit Log

This parameter, of type BACnetObjectIdentifier, specifies the Audit Log object to query.

13.19.1.1.2 Query Parameters

This parameter, of type BACnetAuditLogQueryParameters, specifies the query parameters to apply. The type of parameters varies by the type of query as shown below.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.19.1.1.2.1 Audit Query By Target

Table 13-28. Structure of 'Audit Query By Property' Query Parameters

Parameter Name	Req	Ind	Datatype
Target Device Identifier	M	M(=)	BACnetObjectIdentifier
Target Device Address	U	U(=)	BACnetAddress
Target Object Identifier	U	U(=)	BACnetObjectIdentifier
Target Property Identifier	U	U(=)	BACnetPropertyIdentifier
Target Array Index	U	U(=)	Unsigned
Target Priority	U	U(=)	Unsigned (1..16)
Operations	U	U(=)	BACnetAuditOperationFlags
Result Filter	M	M(=)	BACnetSuccessFilter

The 'Target Device Identifier', 'Target Device Address', 'Target Object Identifier', 'Target Property Identifier', 'Target Array Index' identify the device, object, or property for which audit records are requested.

When both 'Target Device Identifier' and 'Target Device Address' parameters are included, a record is a match if it matches either of the two parameters.

The 'Target Priority' parameter is used to restrict records to only those for the specified priority. Audit records without a 'Priority' will match any 'Target Priority'.

The 'Operations' parameter determines the operation types for which entries are requested. If not present, all operation types will be returned.

The 'Result Filter' parameter indicates whether audit records for successful and/or failed actions will be included in the response.

13.19.1.1.2.2 Audit Query By Source

Table 13-29. Structure of 'Audit Query By Source' Query Parameters

Parameter Name	Req	Ind	Datatype
Source Device Identifier	M	M(=)	BACnetObjectIdentifier
Source Device Address	U	U(=)	BACnetAddress
Source Object Identifier	U	U(=)	BACnetObjectIdentifier
Operations	U	U(=)	BACnetAuditOperationFlags
Result Filter	M	M(=)	BACnetSuccessFilter

The 'Source Device Identifier' and 'Source Object Identifier' identify the operation source for which audit records are requested.

When both the 'Source Device Identifier' and 'Source Device Address' parameters are included, a record is a match if it matches either of the two parameters.

The 'Operations' parameter determines the operation types for which entries are requested. If not present, all operation types will be returned.

The 'Result Filter' parameter indicates whether audit records for successful and/or failed actions will be included in the response.

13.19.1.1.3 Start At Sequence Number

This optional parameter, of type Unsigned64, identifies, by sequence number, where in the log to start the query. Only those records which match the query parameters and which have a sequence number less than the value of this parameter are to be returned. If this parameter is not present, the query will start with the latest record in the log.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.19.1.1.4 Requested Count

This parameter, of type Unsigned16, is used to limit the number of records that will be returned. The responding device shall not return more records than specified by this parameter.

13.19.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters.

13.19.1.2.1 Audit Log

This parameter, of type BACnetObjectIdentifier, specifies the Audit Log object the query was performed on.

13.19.1.2.2 Records

This parameter, of type list of BACnetAuditLogRecordResult, contains the returned records. Each entry is of the form:

Table 13-30. Structure of an Entry of the 'Records' Parameter

Parameter Name	Rsp	Cnf	Parameter Type
Sequence Number	M	M(=)	Unsigned64
Record	M	M(=)	BACnetAuditLogRecord

13.19.1.2.3 No More Items

This parameter, of type BOOLEAN, specifies whether the query processed to the end of the Audit Log and found no more records (TRUE), or not (FALSE). When FALSE, there may, or may not, be more items in the log which match the query.

13.19.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

13.19.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The specified Audit Log object does not exist.	OBJECT	UNKNOWN_OBJECT
The device is not configured to perform audit logging.	SERVICES	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED

Note that there shall be no validation of object/property specification filter parameters. For example, 'Target Array Index' shall not be validated to ensure it is consistent with the 'Target Object Identifier' and 'Target Property Identifier'. The responding device shall just perform the comparison between the audit log records and the filter parameters as described below.

13.19.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to query the specified Audit Log object and construct a list of records that match the query.

Devices that support execution of this service are required to support all query types.

Records are returned latest first. The responding device shall return up to 'Requested Count' records. If the responding device reaches the end of the buffer and all records which match the query are returned, then the 'No More Items' parameter shall be set to TRUE; otherwise it shall be set to FALSE.

If there are no records in the Audit Log that match the query criteria, then a Result(+) shall be returned with no records included and the 'No More Items' parameter set to TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.19.2.1 Audit Query By Target

This query is used to get a history of audit records for objects in a specific device. It selects the most recent records with a sequence number less then 'Start At Sequence Number', and have 'Target Device', 'Target Object', 'Target Property', and 'Priority' fields which match the 'Target Device Identifier' or 'Target Device Address', 'Target Object Identifier', 'Target Property Identifier', 'Target Array Index', and 'Target Priority' query parameters.

Some of the query filter parameters ('Target Object Identifier', 'Target Property Identifier', 'Target Array Index', 'Target Priority') are optional. When one of these parameters is not provided, all values of the corresponding field in the audit log record will be considered a match.

13.19.2.2 Audit Query By Source

This query is used to get a history of audit records of actions initiated by a specific operation source. It selects the most recent records with a sequence number less then 'Start At Sequence Number', and have 'Source Device', and 'Source Object' fields which match the 'Source Device Identifier' and 'Source Object Identifier' query parameters.

The 'Source Object Identifier' and 'Operations' query filter parameter is optional. When one of these parameters is not provided, all values of the corresponding field in the audit log record will be considered a match.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.20 ConfirmedAuditNotification

The ConfirmedAuditNotification service is used by an operation source, operation target, or audit forwarder to provide audit notifications to an audit logger. See Clause 19.6.

13.20.1 ConfirmedAuditNotification Service Structure

The structure of the ConfirmedAuditNotification service primitive is shown in Table 13-31. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-31. Structure of ConfirmedAuditNotification Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Notifications	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.20.1.1 Argument

This parameter shall convey the parameters for the ConfirmedAuditNotification confirmed service request.

13.20.1.1.1 Notifications

This parameter specifies one or more audit notifications of type BACnetAuditNotification. For the content of audit notifications, see Clause 19.6.4.

13.20.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded.

13.20.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

13.20.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The device is not configured as an audit logger.	SERVICE	SERVICE_REQUEST_DENIED

13.20.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall add the audit notifications to the local Audit Log object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

13.21 UnconfirmedAuditNotification

The UnconfirmedAuditNotification service is used by an operation source, operation target, or audit forwarder to provide audit notifications to an audit logger. See Clause 19.6.

13.21.1 UnconfirmedAuditNotification Service Structure

The structure of the UnconfirmedAuditNotification service primitive is shown in Table 13-32. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-32. Structure of UnconfirmedAuditNotification Service Primitives

Parameter Name	Req	Ind
Argument	M	M(=)
Notifications	M	M(=)

13.21.1.1 Argument

This parameter shall convey the parameters for the UnconfirmedAuditNotification confirmed service request.

13.21.1.1.1 Notifications

This parameter specifies one or more audit notifications of type BACnetAuditNotification. For the content of audit notifications, see Clause 19.6.4.

13.21.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall add the audit notifications to the local Audit Log object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

14 FILE ACCESS SERVICES

This clause defines the set of services used to access and manipulate files contained in BACnet devices. The concept of files is used here as a network-visible representation for a collection of octets of arbitrary length and meaning. This is an abstract concept only and does not imply the use of disk, tape or other mass storage devices in the server devices. These services may be used to access vendor-defined files as well as specific files defined in the BACnet protocol standard.

Every file that is accessible by File Access Services shall have a corresponding File object in the BACnet device. This File object is used to identify the particular file by name. In addition, the File object provides access to "header information," such as the file's total size, creation date, and type. File Access Services may model files in two ways: as a continuous stream of octets or as a contiguous sequence of numbered records.

The File Access Services provide atomic read and write operations. In this context "atomic" means that during the execution of a read or write operation, no other AtomicReadFile or AtomicWriteFile operations are allowed for the same file. Synchronization of these services with internal operations of the BACnet device is a local matter and is not defined by this standard.

14.1 AtomicReadFile Service

The AtomicReadFile Service is used by a client BACnet-user to perform an open-read-close operation on the contents of the specified file. The file may be accessed as records or as a stream of octets.

14.1.1 Structure

The structure of the AtomicReadFile service primitives is shown in Table 14-1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 14-1. Structure of AtomicReadFile Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
File Identifier	M	M(=)		
Stream Access	S	S(=)		
File Start Position	M	M(=)		
Requested Octet Count	M	M(=)		
Record Access	S	S(=)		
File Start Record	M	M(=)		
Requested Record Count	M	M(=)		
Result(+)			S	S(=)
End Of File			M	M(=)
Stream Access			S	S(=)
File Start Position			M	M(=)
File Data			C	C(=)
Record Access			S	S(=)
File Start Record			M	M(=)
Returned Record Count			M	M(=)
File Record Data			C	C(=)
Result(-)			S	S(=)
Error Type			M	M(=)

14.1.2 Argument

This parameter shall convey the parameters for the AtomicReadFile confirmed service request.

14.1.2.1 File Identifier

This parameter is the Object_Identifier of the File object that identifies the file to be read.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

14.1.2.2 Stream Access

The 'Stream Access' parameter shall indicate that stream-oriented file access is required. Stream access includes the parameters 'File Start Position' and 'Requested Octet Count'.

14.1.2.2.1 File Start Position

This parameter, of type INTEGER, represents the number of octets from the beginning of the file at which reading shall commence. A 'File Start Position' of 0 is the first octet of the file.

14.1.2.2.2 Requested Octet Count

This parameter, of type Unsigned, represents the number of octets that shall be read from the file starting at the 'File Start Position'.

14.1.2.3 Record Access

The 'Record Access' parameter shall indicate that record-oriented file access is required. Record access includes the parameters 'File Start Record' and 'Requested Record Count'.

14.1.2.3.1 File Start Record

This parameter, of type INTEGER, represents the number of records from the beginning of the file at which reading shall commence. A 'File Start Record' of 0 is the first record of the file.

14.1.2.3.2 Requested Record Count

This parameter, of type Unsigned, represents the number of records that shall be read from the file starting at the 'File Start Record'.

14.1.3 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters.

14.1.3.1 End Of File

The 'End Of File' parameter, of type BOOLEAN, shall be equal to TRUE if this response includes the last octet of the file and FALSE otherwise. This parameter shall be used to check for the end of file since the number of octets returned could be less than the 'Requested Octet Count' or the 'Returned Record Count' could be less than the 'Requested Record Count' due to the amount of data remaining in the file. This parameter also provides a data-independent way for the client user of this service to detect an end of file.

14.1.3.2 Stream Access

The 'Stream Access' parameter shall indicate that stream-oriented file access was requested. Stream access includes the parameters 'File Start Position' and 'File Data'.

14.1.3.2.1 File Start Position

This parameter, of type INTEGER, represents the number of octets from the beginning of the file from which the start of the data was read. A 'File Start Position' of 0 is the first octet of the file.

14.1.3.2.2 File Data

This parameter consists of an OCTET STRING that contains the requested file data.

14.1.3.3 Record Access

The 'Record Access' parameter shall indicate that record-oriented file access was requested. Record access includes the parameters 'File Start Record', 'Returned Record Count', and 'File Record Data'.

14.1.3.3.1 File Start Record

This parameter, of type INTEGER, represents the number of records from the beginning of the file from which the start of the data was read. A 'File Start Record' of 0 is the first record of the file.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

14.1.3.3.2 Returned Record Count

This parameter, of type Unsigned, represents the number of records that were actually read from the file, which may be less than the 'Requested Record Count'.

14.1.3.3.3 File Record Data

This parameter consists of a list of OCTET STRINGs that contain the requested file data.

14.1.4 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

14.1.4.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The File object does not exist.	OBJECT	UNKNOWN_OBJECT
'File Start Record' is out of range.	SERVICES	INVALID_FILE_START_POSITION
Incorrect File access method.	SERVICES	INVALID_FILE_ACCESS_METHOD
A non-File Object Identifier was provided.	SERVICES	INCONSISTENT_OBJECT_TYPE

14.1.5 Service Procedure

The responding BACnet-user shall first verify the validity of the 'File Identifier' parameter and return a 'Result(-)' response with the appropriate error class and error code if the File object is unknown, if there is currently another AtomicReadFile or AtomicWriteFile service in progress, or if the File object is currently inaccessible for another reason. If the 'File Start Position' parameter or the 'File Start Record' parameter is either less than 0 or exceeds the actual file size, then the appropriate error is returned in a 'Result(-)' response. If not, then the responding BACnet-user shall read the number of octets specified by 'Requested Octet Count' or the number of records specified by 'Requested Record Count'. If the number of remaining octets or records is less than the requested amount, then the length of the 'File Data' returned or 'Returned Record Count' shall indicate the actual number read. If the returned response contains the last octet or record of the file, then the 'End Of File' parameter shall be TRUE, otherwise FALSE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

14.2 AtomicWriteFile Service

The AtomicWriteFile Service is used by a client BACnet-user to perform an open-write-close operation of an OCTET STRING into a specified position or a list of OCTET STRINGs into a specified group of records in a file. The file may be accessed as records or as a stream of octets.

14.2.1 Structure

The structure of the AtomicWriteFile service primitives is shown in Table 14-2. The terminology and symbology used in this table are explained in Clause 5.6.

Table 14-2. Structure of AtomicWriteFile Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
File Identifier	M	M(=)		
Stream Access	S	S(=)		
File Start Position	M	M(=)		
File Data	M	M(=)		
Record Access	S	S(=)		
File Start Record	M	M(=)		
Record Count	M	M(=)		
File Record Data	M	M(=)		
Result(+)			S	S(=)
Stream Access			S	S(=)
File Start Position			M	M(=)
Record Access			S	S(=)
File Start Record			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)

14.2.2 Argument

This parameter shall convey the parameters for the AtomicWriteFile confirmed service request.

14.2.2.1 File Identifier

This parameter is the Object_Identifier of the File object that identifies the file to be written.

14.2.2.2 Stream Access

The 'Stream Access' parameter shall indicate that stream-oriented file access is required. Stream access includes the parameters 'File Start Position' and 'File Data'.

14.2.2.2.1 File Start Position

This parameter, of type INTEGER, represents the number of octets from the beginning of the file at which the data shall start being written. A 'File Start Position' of 0 is the first octet of the file. A 'File Start Position' of -1 shall indicate the end of the current file, i.e. an append to file operation.

14.2.2.2.2 File Data

This parameter consists of an OCTET STRING that is to be written to the file.

14.2.2.3 Record Access

The 'Record Access' parameter shall indicate that record-oriented file access is required. Record access includes the parameters 'File Start Record', 'Record Count', and 'File Record Data'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

14.2.2.3.1 File Start Record

This parameter, of type INTEGER, represents the number of records from the beginning of the file at which the data shall start being written. A 'File Start Record' of 0 is the first record of the file. A 'File Start Record' of -1 shall indicate the end of the current file, i.e. an append to file operation.

14.2.2.3.2 Record Count

This parameter, of type Unsigned, represents the number of records that shall be written to the file starting at the 'File Start Record'.

14.2.2.3.3 File Record Data

This parameter consists of a list of OCTET STRINGs that is to be written to the file.

14.2.3 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result shall include the following parameters.

14.2.3.1 Stream Access

The 'Stream Access' parameter shall indicate that stream-oriented file access was requested. Stream access includes the 'File Start Position' parameter. The 'File Start Position' parameter, of type INTEGER, represents the number of octets from the beginning of the file where the data were actually written. A 'File Start Position' of 0 is the first octet of the file.

14.2.3.2 Record Access

The 'Record Access' parameter shall indicate that record-oriented file access was requested. Record access includes the 'File Start Record' parameter. The 'File Start Record' parameter, of type INTEGER, represents the number of records from the beginning of the file where the data were actually written. A 'File Start Record' of 0 is the first record of the file.

14.2.4 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

14.2.4.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The File object does not exist.	OBJECT	UNKNOWN_OBJECT
'File Start Record' is out of range.	SERVICES	INVALID_FILE_START_POSITION
Incorrect File access method.	SERVICES	INVALID_FILE_ACCESS_METHOD
Write to a read-only File.	SERVICES	FILE_ACCESS_DENIED
A syntax error is encountered in the message after the file has been partially modified during the execution of this service.	SERVICES	INVALID_TAG
The File object is full	OBJECT	FILE_FULL
A non-File Object Identifier was provided	SERVICES	INCONSISTENT_OBJECT_TYPE

14.2.5 Service Procedure

The responding BACnet-user shall first verify the validity of the 'File Identifier' parameter and return a 'Result(-)' response with the appropriate error class and error code if the File object is unknown, if there is currently another AtomicReadFile or AtomicWriteFile service in progress, or if the File object is currently inaccessible for another reason. If the 'File Start Position' parameter or the 'File Start Record' parameter exceeds the actual file size, then the file shall be extended to the size indicated, but the contents of any intervening octets or records shall be a local matter. If either of these parameters has the special value -1, then the write operation shall be treated as an append to the current end of file. Then the responding BACnet-user shall write the number of octets specified by 'Octet Count' or the number of records specified by 'Record Count' to the file. If the write fails for any reason, then a 'Result(-)' response with the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

appropriate error class and error code shall be returned. If the write succeeds in its entirety, then a 'Result(+)' response shall be returned. The 'File Start Position' or 'File Start Record' shall indicate the actual position or record at which data were written.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15 OBJECT ACCESS SERVICES

This clause defines application services that collectively provide the means to access and manipulate the properties of BACnet objects. A BACnet object is any object whose properties are accessible through this protocol regardless of its particular function within the device in which it resides. These services may be used to access the properties of vendor-defined objects as well as those of objects specified in this standard.

15.1 AddListElement Service

The AddListElement service is used by a client BACnet-user to add one or more list elements to an object property that is a BACnetLIST or a BACnetARRAY of BACnetLIST.

15.1.1 Structure

The structure of the AddListElement service primitives is shown in Table 15-1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-1. Structure of AddListElement Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Property Identifier	M	M(=)		
Property Array Index	C	C(=)		
List of Elements	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)
First Failed Element Number			M	M(=)

15.1.1.1 Argument

This parameter shall convey the parameters for the AddListElement confirmed service request.

15.1.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose specified property is to be modified by this service.

15.1.1.1.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall provide the means of uniquely identifying the property to be modified by this service.

15.1.1.1.3 Property Array Index

If the property identified above is of datatype BACnetARRAY of BACnetLIST, this conditional parameter of type Unsigned shall be present and shall indicate the array index of the element of the referenced property to be modified by this service. Otherwise, it shall be omitted. The index value, if this parameter is present, shall not be zero.

15.1.1.1.4 List of Elements

This parameter specifies one or more elements that shall be added to the property specified by the 'Property Identifier' parameter, or to the array element of the property if the 'Property Array Index' parameter is present. The datatype of the elements of this parameter is determined by the definition of the object type for the object specified by the 'Object Identifier' parameter.

15.1.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded and all of the specified elements were added to the BACnetLIST.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.1.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed and none of the specified elements were added to the BACnetLIST. The reason for failure is specified by the 'Error Type' parameter.

15.1.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
The element datatype does not match the property.	PROPERTY	INVALID_DATATYPE
The data being written has a datatype not supported by the property.	PROPERTY	DATATYPE_NOT_SUPPORTED
The encoding is not valid for the datatype of the property.	PROPERTY	INVALID_DATA_ENCODING
The element value is out of range for the property.	PROPERTY	VALUE_OUT_OF_RANGE
The specified property is currently not modifiable by the requester.	PROPERTY	WRITE_ACCESS_DENIED
There is not enough free memory for the element.	RESOURCES	NO_SPACE_TO_ADD_LIST_ELEMENT
The property or specified array element is not a BACnetLIST.	SERVICES	PROPERTY_IS_NOT_A_LIST
An array index is provided but the property is not a BACnetARRAY of BACnetLIST.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX

15.1.1.3.2 First Failed Element Number

This parameter, of type Unsigned, shall convey the numerical position, starting at 1, of the offending element in the 'List of Elements' parameter received in the request. If the request is considered invalid for reasons other than the 'List of Elements' parameter, the 'First Failed Element Number' parameter shall be equal to zero.

15.1.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to modify the object identified in the 'Object Identifier' parameter. If the identified object exists and has the property specified in the 'Property Identifier' parameter, and, if present, has the array element specified in the 'Property Array Index' parameter, an attempt shall be made to add all of the elements specified in the 'List of Elements' parameter to the specified property or array element of the property. If this attempt is successful, a 'Result(+)' primitive shall be issued.

When comparing elements in the 'List of Elements' parameter with elements in the specified property or array element of the property, the complete element shall be compared unless the property description specifies otherwise. If one or more of the elements is already present in the BACnetLIST, it shall be updated with the provided element, that is, the existing element is over-written with the provided element. Optionally, if the provided element is exactly the same as the existing element in every way, it can be ignored, that is, not added to the BACnetLIST. Ignoring an element that already exists shall not cause the service to fail.

If the specified object does not exist, the specified property does not exist, the specified array element does not exist, or the specified property or array element is not a BACnetLIST, then the service shall fail and a 'Result(-)' response primitive shall be issued. If one or more elements cannot be added to, or updated in, the BACnetLIST, a 'Result(-)' response primitive shall be issued and no elements shall be added to, or updated in, the BACnetLIST.

The effect of this service shall be to add to, or update in, the BACnetLIST all of the specified elements, or to neither add nor update any elements at all.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.2 RemoveListElement Service

The RemoveListElement service is used by a client BACnet-user to remove one or more elements from the property of an object that is a BACnetLIST or is a BACnetARRAY of BACnetLIST.

15.2.1 Structure

The structure of the RemoveListElement service primitives is shown in Table 15-2. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-2. Structure of RemoveListElement Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Property Identifier	M	M(=)		
Property Array Index	C	C(=)		
List of Elements	M	M(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)
First Failed Element Number			M	M(=)

15.2.1.1 Argument

This parameter shall convey the parameters for the RemoveListElement confirmed service request.

15.2.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose specified property is to be modified by this service.

15.2.1.1.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall provide the means of uniquely identifying the property to be modified by this service.

15.2.1.1.3 Property Array Index

If the property identified above is of datatype BACnetARRAY of BACnetLIST, this conditional parameter of type Unsigned shall be present and shall indicate the array index of the element of the referenced property to be modified by this service. Otherwise, it shall be omitted. The index value, if this parameter is present, shall not be zero.

15.2.1.1.4 List of Elements

This parameter specifies one or more elements that shall be removed from the property specified in the 'Property Identifier' parameter, or from the array element of the property if the 'Property Array Index' parameter is present. The datatype of the elements of this parameter is determined by the definition of the object type for the object specified by the 'Object Identifier' parameter.

15.2.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded and all of the specified elements have been removed.

15.2.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter. None of the elements of the specified BACnetLIST shall be removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.2.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
The element datatype does not match the property.	PROPERTY	INVALID_DATATYPE
The specified property is currently not modifiable by the requestor.	PROPERTY	WRITE_ACCESS_DENIED
A list element to be removed is not present.	SERVICES	LIST_ELEMENT_NOT_FOUND
The property or specified array element is not a BACnetLIST.	SERVICES	PROPERTY_IS_NOT_A_LIST
An array index is provided but the property is not a BACnetARRAY of BACnetLIST.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX

15.2.1.3.2 First Failed Element Number

This parameter, of type Unsigned, shall convey the numerical position, starting at 1, of the offending element in the 'List of Elements' parameter received in the request. If the request is considered invalid for reasons other than the 'List of Elements' parameter, the 'First Failed Element Number' parameter shall be equal to zero.

15.2.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to modify the object identified in the 'Object Identifier' parameter. If the identified object exists and it has the property specified in the 'Property Identifier' parameter, and if present has the array element specified in the 'Property Array Index' parameter, an attempt shall be made to remove the elements in the 'List of Elements' from the property or array element of the property of the object.

When comparing elements of the service with entries in the affected BACnetLIST, the complete element shall be compared unless the property description specifies otherwise. If one or more of the elements does not exist or cannot be removed because of insufficient authority, none of the elements shall be removed and a 'Result(-)' response primitive shall be issued.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.3 CreateObject Service

The CreateObject service is used by a client BACnet-user to create a new instance of an object. This service may be used to create instances of both standard and vendor specific objects. The standard object types supported by this service shall be specified in the PICS. The properties of standard objects created with this service may be initialized in two ways: initial values may be provided as part of the CreateObject service request or values may be written to the newly created object using the BACnet WriteProperty services. The initialization of non-standard objects is a local matter. The behavior of objects created by this service that are not supplied, or only partially supplied, with initial property values is dependent upon the device and is a local matter.

15.3.1 Structure

The structure of the CreateObject service primitives is shown in Table 15-3. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-3. Structure of CreateObject Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Specifier	M	M(=)		
List of Initial Values	U	U(=)		
Result(+)			S	S(=)
Object Identifier			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)
First Failed Element Number			M	M(=)

15.3.1.1 Argument

This parameter shall convey the parameters for the CreateObject confirmed service request.

15.3.1.1.1 Object Specifier

This parameter shall convey information about the type of object that is to be created. The datatype is a choice between an object type and an object identifier. If the object type choice is used, the specified object type shall become the value of the Object_Type property of the newly created object and the responding BACnet-user shall select an object identifier. If the object identifier choice is used, an object with this particular object identifier shall be created.

15.3.1.1.2 List of Initial Values

This parameter shall convey a list of BACnetPropertyValues that shall be used to initialize the values of the specified properties of the newly created object.

15.3.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A success includes successfully initializing all the properties specified in the 'List of Initial Values' parameter. The 'Result(+)' shall convey as a parameter an 'Object Identifier', which is the value of the Object_Identifier property of the newly created object. This identifier shall be unique within the server device.

15.3.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

15.3.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Situation	Error Class	Error Code
The device cannot allocate the space needed for the new object.	RESOURCES	NO_SPACE_FOR_OBJECT
The device supports the object type and may have sufficient space, but does not support the creation of the object for some other reason.	OBJECT	DYNAMIC_CREATION_NOT_SUPPORTED
The device does not support the specified object type.	OBJECT	UNSUPPORTED_OBJECT_TYPE
The object being created already exists.	OBJECT	OBJECT_IDENTIFIER_ALREADY_EXISTS
A datatype of a property value specified in the List of Initial Values does not match the datatype of the property specified by the Property_Identifier.	PROPERTY	INVALID_DATATYPE
A value used in the List of Initial Values is outside the range of values defined for the property specified by the Property_Identifier.	PROPERTY	VALUE_OUT_OF_RANGE
A Property_Identifier has been specified in the List of Initial Values that is unknown for objects of the type being created.	PROPERTY	UNKNOWN_PROPERTY
A character string value was encountered in the List of Initial Values that is not a supported character set.	PROPERTY	CHARACTER_SET_NOT_SUPPORTED
A property specified by the Property_Identifier in the List of Initial Values does not support initialization during the CreateObject service.	PROPERTY	WRITE_ACCESS_DENIED
The data being written has a datatype not supported by the property.	PROPERTY	DATATYPE_NOT_SUPPORTED
The encoding is not valid for the datatype of the property.	PROPERTY	INVALID_DATA_ENCODING

15.3.1.3.2 First Failed Element Number

This parameter, of type Unsigned, shall convey the numerical position, starting at 1, of the offending 'Initial Value' in the 'List of Initial Values' parameter received in the request. If the request is considered invalid for reasons other than the 'List of Initial Values' parameter, the 'First Failed Element Number' shall be equal to zero.

15.3.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to create a new object of the type specified in the 'Object Specifier' parameter.

If the 'Object Specifier' parameter contains an object type, the Object_Identifier property of the newly created object shall be initialized to a value that is unique within the responding BACnet-user device. The method used to generate the object identifier is a local matter. The Object_Type property shall be initialized to the value of the 'Object Specifier' parameter. If a new object of the specified type cannot be created, a 'Result(-)' primitive shall be returned and the 'First Failed Element Number' parameter shall have a value of zero.

If the 'Object Specifier' parameter contains an object identifier, the responding BACnet-user shall determine if an object with this identifier already exists. If such an object exists, then a new object shall not be created, and a 'Result(-)' primitive shall be returned and the 'First Failed Element Number' parameter shall have a value of zero. If such an object does not exist and it cannot be created, a 'Result(-)' primitive shall be returned and the 'First Failed Element Number' parameter shall have a value of zero. If such an object does not exist but it can be created, the new object shall be created. The Object_Identifier property of the new object shall have the value specified in the 'Object Specifier' parameter, and the Object_Type property shall have a value consistent with the object type field of the Object_Identifier. See Clause 20.2.14.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the optional 'List of Initial Values' parameter is included, then all properties in the list shall be initialized as indicated. The initial values of all other properties are a local matter. If this initialization cannot be done, then a 'Result(-)' primitive shall be returned. The 'First Failed Element Number' parameter shall indicate the first property in the 'List of Initial Values' that cannot be initialized, and the object shall not be created. If the attempt to create the object is successful, a 'Result(+)' response primitive shall be issued that conveys the value of the Object_Identifier property of the newly created object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.4 DeleteObject Service

The DeleteObject service is used by a client BACnet-user to delete an existing object. Although this service is general in the sense that it can be applied to any object type, it is expected that most objects in a control system cannot be deleted by this service because they are protected as a security feature. There are some objects, however, that may be created and deleted dynamically. Group objects and Event Enrollment objects are examples. This service is primarily used to delete objects of these types but may also be used to remove vendor-specific deletable objects.

15.4.1 Structure

The structure of the DeleteObject service primitives is shown in Table 15-4. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-4. Structure of DeleteObject Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

15.4.1.1 Argument

This parameter shall convey the parameters for the DeleteObject confirmed service request.

15.4.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall specify the object that is to be deleted by this service.

15.4.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded and the specified object was deleted.

15.4.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed and the specified object was not deleted. The reason for failure is specified in the 'Error type' parameter.

15.4.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The object to be deleted does not exist.	OBJECT	UNKNOWN_OBJECT
The object exists but cannot be deleted.	OBJECT	OBJECT_DELETION_NOT_PERMITTED

15.4.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to delete the object specified by the 'Object Identifier' parameter of the request/indication primitive. If the specified object exists and can be deleted, it shall be deleted and the 'Result(+)' primitive shall be issued. If the specified object does not exist or cannot be deleted, then the 'Result(-)' primitive shall be issued.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.5 ReadProperty Service

The ReadProperty service is used by a client BACnet-user to request the value of one property of one BACnet Object. This service allows read access to any property of any object, whether a BACnet-defined object or not.

15.5.1 Structure

The structure of the ReadProperty service primitives is shown in Table 15-5. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-5. Structure of ReadProperty Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Property Identifier	M	M(=)		
Property Array Index	U	U(=)		
Result (+)			S	S(=)
Object Identifier			M	M(=)
Property Identifier			M	M(=)
Property Array Index			U	U(=)
Property Value			M	M(=)
Result (-)			S	S(=)
Error Type			M	M(=)

15.5.1.1 Argument

This parameter shall convey the parameters for the ReadProperty confirmed service request.

15.5.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose property is to be read and returned to the client BACnet-user.

15.5.1.1.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall provide the means of uniquely identifying the property to be read and returned by this service. Because this service is intended to read a single property of a single object, the value of this parameter shall not be one of the special property identifiers ALL, REQUIRED, or OPTIONAL.

15.5.1.1.3 Property Array Index

If the property identified above is of datatype array, this optional parameter of type Unsigned shall indicate the array index of the element of the property referenced by this service. If the 'Property Array Index' is omitted, this shall mean that the entire array shall be referenced.

If the property identified above is not of datatype array, this parameter shall be omitted.

15.5.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters:

15.5.1.2.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall identify the object whose property has been read and is being returned to the client BACnet-user.

15.5.1.2.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall identify the property that was read and is being returned by this service. Because this service is intended to read a single property of a single object, the value of this parameter shall not be one of the special property identifiers ALL, REQUIRED, or OPTIONAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.5.1.2.3 Property Array Index

If the property identified above is of datatype array and a 'Property Array Index' was specified in the request, this parameter of type Unsigned shall be present and shall indicate the array index of the element of the property referenced by this service. Otherwise it shall be omitted.

15.5.1.2.4 Property Value

If access to the specified property of the specified object was successful, this parameter shall be returned. It shall be of the datatype appropriate to the specified property and shall contain the value of the requested property.

15.5.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

15.5.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The property is not accessible using this service.	PROPERTY	READ_ACCESS_DENIED

15.5.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to access the specified property of the specified object. If the access is successful, a 'Result(+)' primitive, which returns the accessed value, shall be generated. If the access fails, a 'Result(-)' primitive shall be generated, indicating the reason for the failure.

When the object-type in the Object Identifier parameter contains the value DEVICE and the instance in the 'Object Identifier' parameter contains the value 4194303, the responding BACnet-user shall treat the Object Identifier as if it correctly matched the local Device object. This allows the device instance of a device that does not generate I-Am messages to be determined.

When the object-type in the Object Identifier parameter contains the value NETWORK_PORT and the instance in the 'Object Identifier' parameter contains the value 4194303, the responding BACnet-user shall treat the Object Identifier as if it correctly matched the local Network Port object representing the network port through which the request was received. This allows the network port instance of the network port that was used to receive the request to be determined.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.6 Deleted Clause

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.7 ReadPropertyMultiple Service

The ReadPropertyMultiple service is used by a client BACnet-user to request the values of one or more specified properties of one or more BACnet Objects. This service allows read access to any property of any object, whether a BACnet-defined object or not. The user may read a single property of a single object, a list of properties of a single object, or any number of properties of any number of objects. A 'Read Access Specification' with the property identifier ALL can be used to learn the implemented properties of an object along with their values.

15.7.1 Structure

The structure of the ReadPropertyMultiple service primitives is shown in Table 15-10. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-10. Structure of ReadPropertyMultiple Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
List of Read Access Specifications	M	M(=)		
Result (+)			S	S(=)
List of Read Access Results			M	M(=)
Result (-)			S	S(=)
Error Type			M	M(=)

15.7.1.1 Argument

This parameter shall convey the parameters for the ReadPropertyMultiple confirmed service request.

15.7.1.1.1 List of Read Access Specifications

This parameter shall consist of a list of one or more 'Read Access Specifications'. Each specification shall consist of two parameters: (1) an 'Object Identifier' and (2) a 'List of Property References'. See Clause 15.7.3.1.

15.7.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameter.

15.7.1.2.1 List of Read Access Results

The 'List of Read Access Results' parameter shall indicate the success or failure of the access to each specified property. The contents of each Read Access Result are described in Clause 15.7.3.2.

15.7.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

15.7.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The property is not accessible using this service.	PROPERTY	READ_ACCESS_DENIED

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.7.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to access the specified properties of the specified objects and shall construct a 'List of Read Access Results' in the order specified in the request. If the 'List of Property References' portion of the 'List of Read Access Specifications' parameter contains the property identifier ALL, REQUIRED, or OPTIONAL, then the 'List of Read Access Results' shall be constructed as if each property being returned had been explicitly referenced (see Clause 15.7.3.1.2). While there is no requirement that the request be carried out "atomically," nonetheless the responding BACnet-user shall ensure that all readings are taken in the shortest possible time subject only to higher priority processing. The request shall continue to be executed until an attempt has been made to access all specified properties. If none of the specified objects is found or if none of the specified properties of the specified objects can be accessed, either a 'Result(-)' primitive or a Result(+) primitive that returns error codes for all properties shall be issued. If any of the specified properties of the specified objects can be accessed, then a 'Result(+)' primitive shall be issued, which returns all accessed values and error codes for all properties that could not be accessed.

When the object-type in the Object Identifier portion of the Read Access Specification parameter contains the value DEVICE and the instance of that 'Object Identifier' parameter contains the value 4194303, the responding BACnet-user shall treat the Object Identifier as if it correctly matched the local Device object. This allows the device instance of a device that does not generate I-Am messages to be determined.

When the object-type in the Object Identifier portion of the Read Access Specification contains the value NETWORK_PORT and the instance of that 'Object Identifier' parameter contains the value 4194303, the responding BACnet-user shall treat the Object Identifier as if it correctly matched the local Network Port object representing the network port through which the request was received. This allows the network port instance of the network port that was used to receive the request to be determined.

15.7.3 Parameters Referenced by the ReadPropertyMultiple Service

The following parameters appear in the ReadPropertyMultiple service primitives.

15.7.3.1 Read Access Specification Parameter

The 'Read Access Specification' parameter is shown in Table 15-11. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-11. Structure of 'Read Access Specification' Parameter

Parameter Name	Req Ind	Rsp Cnf	Datatype
Object Identifier	M	M(=)	BACnetObjectIdentifier
List of Property References	M	M(=)	List of BACnetPropertyReference

15.7.3.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose properties are to be read and returned to the service requester.

15.7.3.1.2 List of Property References

This parameter shall be a list of one or more BACnetPropertyReferences, each of which corresponds directly to a specific property of the object identified above. The property identifier ALL means that all defined properties of the object are to be accessed, including any proprietary properties.

The property identifier REQUIRED means that only those standard properties having a conformance code of "R" or "W" shall be returned. The property identifier OPTIONAL means that only those standard properties present in the object that have a conformance code "O" shall be returned. If no optional properties are present then no entries associated with OPTIONAL property identifier shall be in the 'List of Results' and if no other properties were requested, then the 'List of Results' shall be present and empty. The Property_List property shall not be returned when properties ALL or REQUIRED are requested. See the specification for the particular object type in Clause 12. If the property identifier ALL, REQUIRED, or OPTIONAL is specified and any of the selected properties is not readable by this service, then a Property Access Error for that property shall be returned in the List of Read Access Results as specified by Clause 15.7.3.2.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.7.3.2 Read Access Result

The 'Read Access Result' parameter is shown in Table 15-12. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-12. Structure of 'Read Access Result' Parameter

Parameter Name	Rsp	Cnf	Datatype
Object Identifier	M	M(=)	BACnetObjectIdentifier
List of Results	M	M(=)	
Property Identifier	M	M(=)	BACnetPropertyIdentifier
Property Array Index	U	U(=)	Unsigned
Property Value	S	S(=)	ANY
Property Access Error	S	S(=)	Error

15.7.3.2.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall identify the object whose properties are being returned to the service requester.

15.7.3.2.2 List of Results

The result of reading a given property is either the present value of the property or an error code indicating why the access attempt failed. Each element in the 'List of Results' contains a 'Property Identifier' and conditionally a 'Property Array Index', followed by either a 'Property Value' or a 'Property Access Error'.

15.7.3.2.2.1 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall identify the property whose value has been read.

15.7.3.2.2.2 Property Array Index

If the property identified above is of datatype array and a 'Property Array Index' was specified in the request, this parameter of type Unsigned shall be present and shall indicate the array index of the element of the property referenced by this service. Otherwise it shall be omitted.

15.7.3.2.2.3 Property Value

If access to the specified property of the specified object is successful, this parameter shall be returned. It shall be of a datatype consistent with the requested property and shall contain the value of the requested property.

15.7.3.2.2.4 Property Access Error

If the responding BACnet-user is unable to access the specified property of the specified object, then this parameter shall be returned. It shall contain a value that indicates the reason for the access failure. This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. Note that this parameter refers only to a failure of the access to a specific property of a specific object, whereas the 'Error Type' parameter returned in the 'Result(-)' primitive refers to a failure of the entire ReadPropertyMultiple service request.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.8 ReadRange Service

The ReadRange service is used by a client BACnet-user to read a specific range of data items representing a subset of data available within a specified object property. The service may be used with any BACnetLIST or BACnetARRAY of BACnetLIST property.

15.8.1 Structure

The structure of the ReadRange primitive is shown in Table 15-13. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-13. Structure of ReadRange Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Property Identifier	M	M(=)		
Property Array Index	C	C(=)		
Range	U	U(=)		
Result(+)			S	S(=)
Object Identifier			M	M(=)
Property Identifier			M	M(=)
Property Array Index			C	C(=)
Result Flags			M	M(=)
Item Count			M	M(=)
Item Data			M	M(=)
First Sequence Number			C	C(=)
Result(-)			S	S(=)
Error Type			M	M(=)

15.8.1.1 Argument

This parameter shall convey the parameters for the ReadRange confirmed service request.

15.8.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, specifies the object and property to be read.

15.8.1.1.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, specifies the property to be read by this service. Because this service is intended to read a single property of a single object, the value of this parameter shall not be one of the special property identifiers ALL, REQUIRED, or OPTIONAL.

15.8.1.1.3 Property Array Index

If the property identified above is of datatype BACnetARRAY of BACnetLIST, this optional parameter of type Unsigned shall indicate the array index of the element of the property referenced by this service. If the property identified above is not of datatype BACnetARRAY of BACnetLIST, this parameter shall be omitted. The index value, if this parameter is present, shall not be zero.

15.8.1.1.4 Range

This optional parameter shall convey criteria for the consecutive range of items within the referenced BACnetLIST that are to be returned, as described in Clause 15.8.2. The 'Range' parameter is shown in Table 15-14. The terminology and symbology used in this table are explained in Clause 5.6.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 15-14. Structure of the 'Range' Parameter

Parameter Name	Req	Ind	Datatype
By Position	S	S(=)	
Reference Index	M	M(=)	Unsigned
Count	M	M(=)	INTEGER16
By Sequence Number	S	S(=)	
Reference Sequence Number	M	M(=)	Unsigned
Count	M	M(=)	INTEGER16
By Time	S	S(=)	
Reference Time	M	M(=)	BACnetDateTime
Count	M	M(=)	INTEGER16

15.8.1.1.4.1 By Position

The 'By Position' parameter shall indicate that the particular items to be read are referenced by an index. For indexing of BACnetLIST items, see Clause 12.1.5.2.

15.8.1.1.4.1.1 Reference Index

The 'Reference Index' parameter specifies the index of the first (if 'Count' is positive) or last (if 'Count' is negative) item to be read. If the item with the index specified in this parameter does not exist, then no items match the criteria for being read and returned, regardless of the value of the 'Count' parameter.

Devices that execute ReadRange shall support Unsigned32 values for reference indexes unless the device contains object types which have Total_Record_Count properties of type Unsigned64, or which are capable of containing list properties with more than 2^{32}-1 entries, in which case the device shall support Unsigned64 reference indexes.

Devices which initiate the By Position form of ReadRange with arbitrary reference indexes shall support Unsigned32 values for reference indexes unless the device interacts with object types which have Total_Record_Count properties of type Unsigned64, in which case the device shall support Unsigned64 reference indexes.

15.8.1.1.4.1.2 Count

The absolute value of the 'Count' parameter specifies the number of items to be read. If 'Count' is positive, the item specified by 'Reference Index' shall be the first item read and returned; if 'Count' is negative the item specified by 'Reference Index' shall be the last record. 'Count' may not be zero.

15.8.1.1.4.1.3 Example - Positive Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Index' = 800 and 'Count' = 300. The resulting ReadRange service response contains 200 items from Index 800 to 999 with FIRST_ITEM = FALSE, LAST_ITEM = FALSE, and MORE_ITEMS = TRUE. The 'First Sequence Number' parameter is not included in the response.

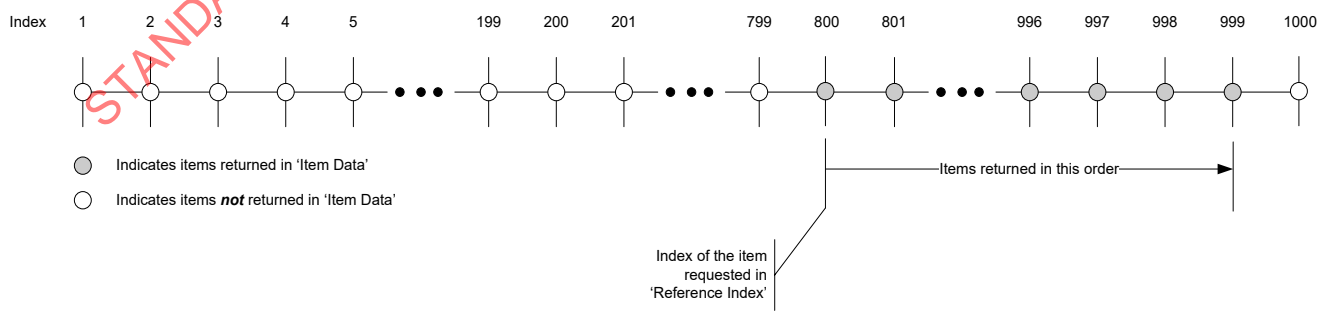


Figure 15-1. By Position with a Positive Count

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

15.8.1.1.4.1.4 Example - Negative Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Index' = 1000 and 'Count' = -1000. The resulting ReadRange service response contains 200 items from Index 801 to 1000 with FIRST_ITEM = FALSE, LAST_ITEM = TRUE and MORE_ITEMS = TRUE. The 'First Sequence Number' shall not exist in the response.

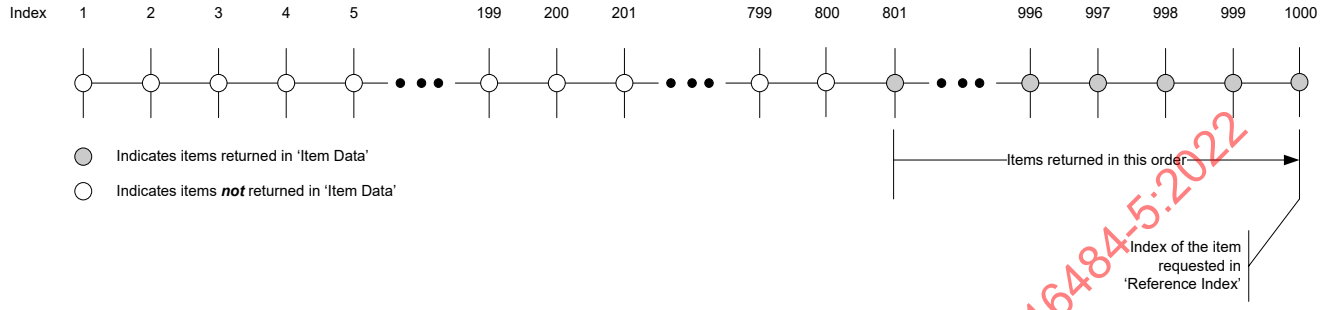


Figure 15-2. By Position with a Negative Count

15.8.1.1.4.2 By Sequence Number

The 'By Sequence Number' parameter shall indicate that the particular items to be read are referenced by a sequence number and that the response shall include the sequence number of the first returned item. This differs semantically from the 'By Position' parameter choice. The 'Reference Index' provided in the 'By Position' choice references an item by its position in the list. In contrast, the 'Reference Sequence Number' provided in the 'By Sequence Number' choice references an item by its sequence number, which it is given when the item is added to the list. Not all BACnetLIST properties or array elements that are a BACnetLIST implement the concept of a sequence number. An example of a BACnetLIST that does implement the concept of a sequence number is the Log_Buffer property of the Trend Log object.

15.8.1.1.4.2.1 Reference Sequence Number

The 'Reference Sequence Number' parameter specifies the sequence number of the first (if 'Count' is positive) or last (if 'Count' is negative) item to be read. If the item with the sequence number specified in this parameter does not exist, then no items match the criteria for being read and returned, regardless of the value of the 'Count' parameter.

Devices that execute ReadRange shall support Unsigned32 values for sequence numbers, unless the device contains object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

Devices which initiate ReadRange shall support Unsigned32 values for sequence numbers, unless the device interacts with object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

15.8.1.1.4.2.2 Count

The absolute value of the 'Count' parameter specifies the number of items to be read. If 'Count' is positive, the item specified by 'Reference Sequence Number' shall be the first and oldest item read and returned. If 'Count' is negative the item specified by 'Reference Sequence Number' shall be the last and newest item read and returned. 'Count' shall not be zero.

15.8.1.1.4.2.3 Example - Positive Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Sequence Number' = 2800 and 'Count' = 300. The resulting ReadRange service response contains 200 items from Sequence Number 2800 to 2999 with FIRST_ITEM = FALSE, LAST_ITEM = FALSE and MORE_ITEMS = TRUE. The 'First Sequence Number' = 2800.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

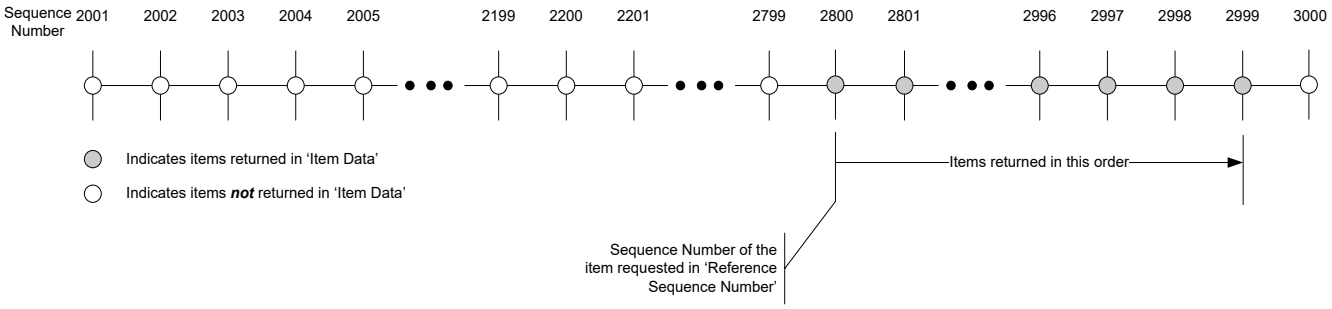


Figure 15-3. By Sequence Number with a Positive Count

15.8.1.1.4.2.4 Example - Negative Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Sequence Number' = 3000 and 'Count' = -1000. The resulting ReadRange service response contains 200 items from Sequence Number 2801 to 3000 with FIRST_ITEM = FALSE, LAST_ITEM = TRUE, and MORE_ITEMS = TRUE. The 'First Sequence Number' = 2801.

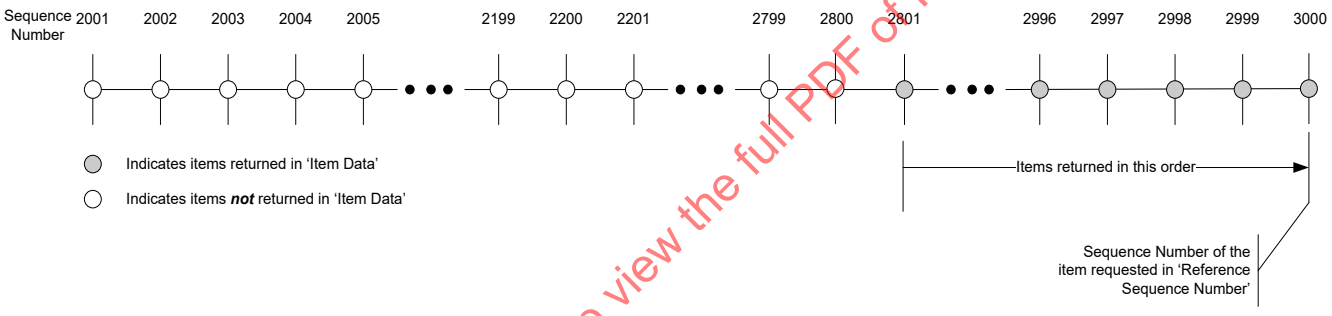


Figure 15-4. By Sequence Number with a Negative Count

15.8.1.1.4.3 By Time

The 'By Time' parameter shall indicate that the particular item to be read is referenced by timestamp and that the Sequence Number of the item shall be returned in the response. This form of the service is expected to be used when searching lists that are loosely indexed by time.

15.8.1.1.4.3.1 Reference Time

If 'Count' is positive, the first item to be read shall be the first item with a timestamp newer than the time specified by the 'Reference Time' parameter. If 'Count' is negative, the last item to be read shall be the newest item with a timestamp older than the time specified by the 'Reference Time' parameter. This parameter shall contain a specific datetime value.

15.8.1.1.4.3.2 Count

The absolute value of the 'Count' parameter specifies the number of items to be read. If 'Count' is positive, the first item with a timestamp newer than the time specified by 'Reference Time' shall be the first and oldest item read and returned; if 'Count' is negative, the newest item with a timestamp older than the time specified by 'Reference Time' shall be the last and newest item. 'Count' shall not be zero.

15.8.1.1.4.3.3 Example - Positive Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Time' = March 18, 2013, 14:19:00 and 'Count' = 300. The resulting ReadRange service response contains 200 items from March 18, 2013, 14:20:00 to March 18, 2013, 17:39:00 with FIRST_ITEM = FALSE, LAST_ITEM = FALSE, and MORE_ITEMS = TRUE. The 'First Sequence Number' = 2800.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

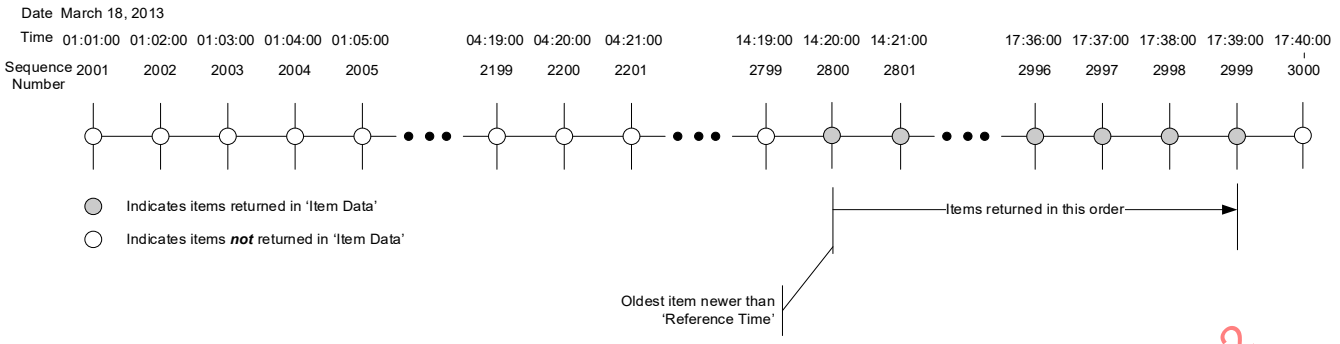


Figure 15-5. By Time with a Positive Count

15.8.1.1.4.3.4 Example - Positive Count, Outdated Reference Time

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Time' = November 17, 1991, 19:20:00 and 'Count' = 300. The resulting ReadRange service response contains 200 items from March 18, 2013, 01:01:00 to March 18, 2013, 04:20:00 with FIRST_ITEM = TRUE, LAST_ITEM = FALSE, and MORE_ITEMS = TRUE. The 'First Sequence Number' = 2001.

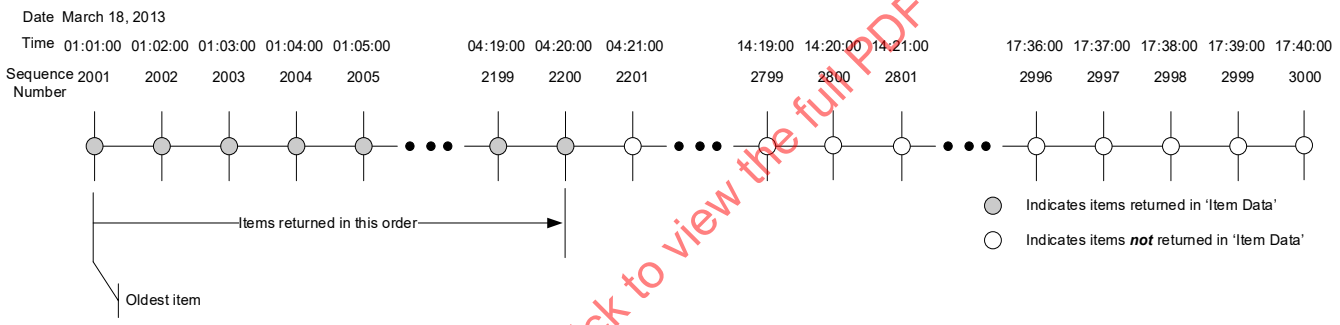


Figure 15-6. By Time with a Positive Count, Outdated Reference Time

15.8.1.1.4.3.5 Example - Negative Count

Assume a device contains a list with 1000 items and is capable of returning 200 items in a ReadRange response. The ReadRange service request contains a 'Reference Time' = March 18, 2013, 17:40:00 and 'Count' = -1000. The resulting ReadRange service response contains 200 items from March 18, 2013, 14:20:00 to March 18, 2013, 17:39:00 with FIRST_ITEM = FALSE, LAST_ITEM = FALSE, and MORE_ITEMS = TRUE. The 'First Sequence Number' = 2800.

Figure 15-7. By Time with a Negative Count

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.8.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters.

15.8.1.2.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, specifies the object that was read.

15.8.1.2.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall identify that property that was read.

15.8.1.2.3 Property Array Index

If the property identified above is of datatype BACnetARRAY if BACnetLIST, this parameter of type Unsigned shall indicate the array index of the array element of the property referenced by this service. If the property identified above is not of datatype BACnetARRAY of BACnetLIST, this parameter shall be omitted.

15.8.1.2.4 Result Flags

This parameter, of type BACnetResultFlags, shall convey several flags that describe characteristics of the response data:

{FIRST_ITEM, LAST_ITEM, MORE_ITEMS}

The FIRST_ITEM flag indicates whether this response includes the first list item (in the case of positional indexing), or the oldest timestamped item (in the case of time indexing).

The LAST_ITEM flag indicates whether this response includes the last list item (in the case of positional indexing), or the newest timestamped item (in the case of time indexing)

The MORE_ITEMS flag indicates whether more items matched the request but were not transmittable within the PDU.

15.8.1.2.5 Item Count

This parameter, of type Unsigned, represents the number of items that were returned.

15.8.1.2.6 Item Data

This parameter consists of a list of the requested data.

15.8.1.2.7 First Sequence Number

This parameter, of type Unsigned32, specifies the sequence number of the first item returned. This parameter is only included if the 'Range' parameter of the request was of the type 'By Sequence Number' or 'By Time' and 'Item Count' is greater than 0.

Devices that execute ReadRange shall support Unsigned32 values for sequence numbers, unless the device contains object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

Devices which initiate ReadRange shall support Unsigned32 values for sequence numbers, unless the device is intended to interact with object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

15.8.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for the failure shall be specified by the 'Error Type' parameter.

15.8.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Situation	Error Class	Error Code
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
The specified property is currently not readable by the requester.	PROPERTY	READ_ACCESS_DENIED
Property is not a BACnetLIST or BACnetARRAY of BACnetLIST.	SERVICES	PROPERTY_IS_NOT_A_LIST
An array index is provided but the property is not a BACnetARRAY of BACnetLIST.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The list items do not have a sequence number but a request 'By Sequence Number' was received.	PROPERTY	LIST_ITEM_NOT_NUMBERED
The list items are not timestamped but a request 'By Time' was received.	PROPERTY	LIST_ITEM_NOT_TIMESTAMPED

15.8.2 Service Procedure

The responding BACnet-user shall first verify the validity of the 'Object Identifier', 'Property Identifier' and 'Property Array Index' parameters and return a 'Result(-)' response with the appropriate error class and error code if the object or property is unknown, if the referenced property or array element is not a BACnetLIST, the list items do not support the range selection type, or if it is currently inaccessible for another reason.

If the 'Range' parameter is not present, then the responding BACnet-user shall read and attempt to return all of the available items in the BACnetLIST.

If the 'Range' parameter is present and specifies the 'By Position' parameters, then the responding BACnet-user shall read and attempt to return all of the items specified. The items specified include the item at the index specified by 'Reference Index' plus up to 'Count' - 1 items following if 'Count' is positive, or up to -1 - 'Count' items preceding if 'Count' is negative. For indexing of BACnetLIST items, see Clause 12.1.5.2.

If the 'Range' parameter is present and specifies the 'By Time' parameter, then the responding BACnet-user shall read and attempt to return all of the items specified. If 'By Time' parameters are specified and the items are not timestamped an error shall be returned. If 'Count' is positive, the items specified include the first items with a timestamp newer than 'Reference Time' plus up to 'Count'-1 items following. If 'Count' is negative, the items specified include the newest record with a timestamp older than 'Reference Time' and up to -1-'Count' items preceding. The sequence number of the first item returned shall be included in the response. The items shall be returned in chronological order.

If the 'Range' parameter is present and specifies the 'By Sequence Number' parameters, then the responding BACnet-user shall read and attempt to return all of the items specified. If 'By Sequence Number' parameters are specified and the items are not sequence numbered, an error shall be returned.The items specified are all items with a sequence number in the range 'Reference Sequence Number' to 'Reference Sequence Number' plus 'Count'-1 if 'Count' is positive, or in the range 'Reference Sequence Number' plus 'Count'+1 to 'Reference Sequence' if 'Count' is negative.

To avoid missing items when using chained time-based reads, the first item in the desired set should be found using the 'By Time' form of the 'Range' parameter. Subsequent requests to retrieve the remaining items in the desired set should use the 'By Sequence Number' form of the 'Range' parameter. The reason for this is that BACnetLIST items that include a timestamp but are ordered by time of arrival may have items with out-of-order timestamps due to negative time changes in the local device's clock. If items are read that match the request parameters but cannot be returned in the response, the 'Result Flags' parameter shall contain the MORE_ITEMS flag set to TRUE, otherwise it shall be FALSE. Remaining items may be obtained with subsequent requests specifying appropriately chosen parameters.

The returned response shall convey the number of items read and returned using the 'Item Count' parameter. The actual items shall be returned in the 'Item Data' parameter. If the returned response includes the first positional index and a 'By Position' request had been made, or the oldest sequence number and a 'By Sequence Number' or 'By Time' request had been made, then the 'Result Flags' parameter shall contain the FIRST_ITEM flag set to TRUE; otherwise it shall be FALSE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the returned response includes the last positional index and a 'By Position' request had been made, or the newest sequence number and a 'By Sequence Number' or 'By Time' request had been made, then the 'Result Flags' shall contain the LAST_ITEM flag set to TRUE; otherwise it shall be FALSE.

If there are no items in the list that match the 'Range' parameter criteria, then a Result(+) shall be returned with an 'Item Count' of 0 and no 'First Sequence Number' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.9 WriteProperty Service

The WriteProperty service is used by a client BACnet-user to modify the value of a single specified property of a BACnet object. This service potentially allows write access to any property of any object, whether a BACnet-defined object or not. Some implementors may wish to restrict write access to certain properties of certain objects. In such cases, an attempt to modify a restricted property shall result in the return of an error of 'Error Class' PROPERTY and 'Error Code' WRITE_ACCESS_DENIED. Note that these restricted properties may be accessible through the use of Virtual Terminal services or other means at the discretion of the implementor.

15.9.1 Structure

The structure of the WriteProperty service primitives is shown in Table 15-15. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-15. Structure of WriteProperty Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Object Identifier	M	M(=)		
Property Identifier	M	M(=)		
Property Array Index	U	U(=)		
Property Value	M	M(=)		
Priority	C	C(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)

15.9.1.1 Argument

This parameter shall convey the parameters for the WriteProperty confirmed service request.

15.9.1.1.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose property is to be modified.

15.9.1.1.2 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall provide the means of uniquely identifying the property to be written by this service. Because this service is intended to write a single property of a single object, the value of this parameter shall not be one of the special property identifiers ALL, REQUIRED, or, OPTIONAL.

15.9.1.1.3 Property Array Index

If the property identified above is of datatype array, this optional parameter of type Unsigned shall indicate the array index of the element of the property referenced by this service. If the 'Property Array Index' is omitted for an array, this shall mean that the entire array shall be referenced.

If the property identified above is not of datatype array, this parameter shall be omitted.

15.9.1.1.4 Property Value

If access to the specified property of the specified object is successful, this parameter shall be used to replace the value of the property at the time of access. It shall be of any datatype that is valid for the property being modified.

15.9.1.1.5 Priority

This parameter shall be an integer in the range 1..16, which indicates the priority assigned to this write operation. If an attempt is made to write to a commandable property without specifying the priority, a default priority of 16 (the lowest priority) shall be assumed. If an attempt is made to write to a property that is not commandable with a specified priority, the priority shall be ignored. See Clause 19.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.9.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded in its entirety.

15.9.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

15.9.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The specified property is currently not writable by the requestor.	PROPERTY	WRITE_ACCESS_DENIED
The datatype of the value provided is incorrect for the specified property.	PROPERTY	INVALID_DATATYPE
The property is Object_Name and the name is already in use in the device.	PROPERTY	DUPLICATE_NAME
The property is Object Identifier and the identifier is already in use in the device.	PROPERTY	DUPLICATE_OBJECT_ID
The value provided is outside the range of values that the property can take on.	PROPERTY	VALUE_OUT_OF_RANGE
There is not enough space to store the new value.	RESOURCES	NO_SPACE_TO_WRITE_PROPERTY
The data being written has a datatype not supported by the property.	PROPERTY	DATATYPE_NOT_SUPPORTED
The encoding is not valid for the datatype of the property.	PROPERTY	INVALID_DATA_ENCODING
The Priority parameter is not within the defined range of 1..16. This condition may be ignored if the property is not commandable.	SERVICES	PARAMETER_OUT_OF_RANGE

15.9.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to modify the specified property of the specified object using the value provided in the 'Property Value' parameter. If the modification attempt is successful, a 'Result(+)' primitive shall be issued. If the modification attempt fails, a 'Result(-)' primitive shall be issued indicating the reason for the failure. Interpretation of the conditional Priority parameter shall be as defined in Clause 19.2.

If an attempt is made to relinquish a property that is not commandable and for which NULL is not a supported datatype, if no other error conditions exist, the property shall not be changed, and the write shall be considered successful. See Clause 19.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.10 WritePropertyMultiple Service

The WritePropertyMultiple service is used by a client BACnet-user to modify the value of one or more specified properties of a BACnet object. This service potentially allows write access to any property of any object, whether a BACnet-defined object or not.

Properties shall be modified by the WritePropertyMultiple service in the order specified in the 'List of Write Access Specifications' parameter, and execution of the service shall continue until all of the specified properties have been written to or a property is encountered that for some reason cannot be modified as requested.

Some implementors may wish to restrict write access to certain properties of certain objects. In such cases, an attempt to modify a restricted property shall result in the return of an error of 'Error Class' PROPERTY and 'Error Code' WRITE_ACCESS_DENIED. Note that these restricted properties may be accessible through the use of Virtual Terminal services or other means at the discretion of the implementor.

15.10.1 Structure

The structure of the WritePropertyMultiple service primitives is shown in Table 15-16. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-16. Structure of WritePropertyMultiple Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
List of Write Access Specifications	M	M(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)
First Failed Write Attempt			M	M(=)

15.10.1.1 Argument

This parameter shall convey the parameters for the WritePropertyMultiple confirmed service request.

15.10.1.1.1 List of Write Access Specifications

Each 'Write Access Specification' conveys the information required to carry out the modification of a property or properties of a BACnet object. The specification consists of up to five parameters: (1) an 'Object Identifier'; a List of Properties each of which consists of (2) a 'Property Identifier'; (3) an optional 'Property Array Index'; (4) a 'Property Value'; and (5) an optional 'Priority'.

15.10.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded in its entirety and that all specified properties were correctly modified.

15.10.1.3 Result(-)

The 'Result(-)' parameter shall indicate that at least one of the specified properties could not be modified as requested. The reason for the failure shall be conveyed by the 'Error Type' parameter along with the 'Object Identifier', 'Property Identifier', and the optional 'Property Array Index' of the first encountered property that, for the reason specified by the 'Error Type' parameter, could not be properly written.

15.10.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned in a 'Result(-)' for specific situations are as follows:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The specified property is currently read-only.	PROPERTY	WRITE_ACCESS_DENIED
The datatype of the value provided is incorrect for the specified property.	PROPERTY	INVALID_DATATYPE
The property is Object_Name and the name is already in use in the device.	PROPERTY	DUPLICATE_NAME
The property is Object Identifier and the identifier is already in use in the device.	PROPERTY	DUPLICATE_OBJECT_ID
The value provided is outside the range of values that the property can take on.	PROPERTY	VALUE_OUT_OF_RANGE
There is not enough space to store the new value.	RESOURCES	NO_SPACE_TO_WRITE_PROPERTY
The data being written has a datatype not supported by the property.	PROPERTY	DATATYPE_NOT_SUPPORTED
The encoding is not valid for the datatype of the property.	PROPERTY	INVALID_DATA_ENCODING
The Priority parameter is not within the defined range of 1..16. This condition may be ignored if the property is not commandable.	SERVICES	PARAMETER_OUT_OF_RANGE
A syntax error is encountered in the message after one or more properties have been successfully written.	SERVICES	INVALID_TAG

15.10.1.3.2 First Failed Write Attempt

This parameter, of type BACnetObjectPropertyReference, shall convey the 'Object Identifier', 'Property Identifier', and the optional 'Property Array Index' of the first failed element in the 'List of Write Access Specification' for which the write attempt failed.

15.10.2 Service Procedure

For each 'Write Access Specification' contained in the 'List of Write Access Specifications', the value of each specified property shall be replaced by the property value provided in the 'Write Access Specification' and a 'Result(+)' primitive shall be issued, indicating that the service request was carried out in its entirety. Interpretation of the conditional Priority parameter shall be as specified in Clause 19.

If, in the process of carrying out the modification of the indicated properties in the order specified in the 'List of Write Access Specifications', a property is encountered that cannot be modified, the responding BACnet-user shall issue a 'Result(-)' response primitive indicating the reason for the failure. The result of this service shall be either that all of the specified properties or only the properties up to, but not including, the property specified in the 'First Failed Write Attempt' parameter were successfully modified.

A BACnet-Reject-PDU shall be issued only if no write operations have been successfully executed, indicating that the service request was rejected in its entirety. If any of the write operations contained in the 'List of Write Access Specifications' have been successfully executed, a Result(-) response indicating the reason for the failure shall be issued as described above.

In the case that the 'Object Identifier', the 'Property Identifier', or the 'Property Array Index' cannot be successfully decoded after at least one write operation has completed successfully, the object instance portion of the 'Object Identifier' specified in the 'First Failed Write Attempt' shall contain the instance value 4194303. In this case, the value of the 'Property Identifier' parameter and the 'Property Array Index' parameter is a local matter.

If an attempt is made to relinquish a property that is not commandable and for which NULL is not a supported datatype, if no other error conditions exist, the property shall not be changed, and the write to that property shall be considered successful. See Clause 19.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.10.3 Parameters Referenced by the WritePropertyMultiple Service

The 'Write Access Specification' parameter is shown in Table 15-17. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-17. Structure of 'Write Access Specification' Parameter

Parameter Name	Req Ind	Rsp Cnf	Datatype
Object Identifier	M	M(=)	BACnetObjectIdentifier
List of Properties	M	M(=)	
Property Identifier	M	M(=)	BACnetPropertyIdentifier
Property Array Index	U	U(=)	Unsigned
Property Value	M	M(=)	ANY
Priority	C	C(=)	Unsigned(1..16)

15.10.3.1 Object Identifier

This parameter, of type BACnetObjectIdentifier, shall provide the means of identifying the object whose property or properties are to be modified.

15.10.3.2 List of Properties

This parameter shall specify a list of one or more properties of the object identified above, the value to be written to each property, and the priority assigned to the write operation. Each element of the list shall specify the following parameters.

15.10.3.2.1 Property Identifier

This parameter, of type BACnetPropertyIdentifier, shall provide the means of uniquely identifying the property to be written by this service. The value of this parameter shall not be one of the special property identifiers ALL, REQUIRED, or, OPTIONAL.

15.10.3.2.2 Property Array Index

If the property identified above is of datatype array, this optional parameter of type Unsigned shall indicate the array index of the element of the property referenced by this service. If the 'Property Array Index' is omitted for an array, this shall mean that the entire array shall be referenced.

If the property identified above is not of datatype array, this parameter shall be omitted.

15.10.3.2.3 Property Value

If access to the specified property of the specified object is successful, this parameter shall be used to replace the value of the property at the time of access. It shall be of any datatype that is valid for the property being modified.

15.10.3.2.4 Priority

This parameter shall be an integer in the range 1..16, which indicates the priority assigned to this write operation. If an attempt is made to write to a commandable property without specifying the priority, a default priority of 16 (the lowest priority) shall be assumed. If an attempt is made to write to a property that is not commandable with a specified priority, the priority shall be ignored. See Clause 19.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.11 WriteGroup Service

The purpose of WriteGroup is to facilitate the efficient distribution of values to a large number of devices and objects. WriteGroup provides compact representations for data values that allow rapid transfer of many values. See Clause 12-53 and Figure 12-10.

The WriteGroup service is used by a sending BACnet-user to update arbitrary Channel objects' Present_Value properties for a particular numbered control group. The WriteGroup service is an unconfirmed service. Upon receipt of a WriteGroup service request, all devices that are members of the specified control group shall write to their corresponding Channel objects' Present_Value properties with the value applicable to the Channel Number, if any. A device shall be considered to be a member of a control group if that device has one or more Channel objects for which the 'Group Number' from the service appears in its Control_Groups property. If the receiving device does not contain one or more Channel objects with matching channel numbers, then those values shall be ignored.

The WriteGroup service may be unicast, multicast, broadcast locally, on a particular remote network, or using the global BACnet network address. Since global broadcasts are generally discouraged, the use of multiple directed broadcasts is preferred.

15.11.1 WriteGroup Service Structure

The structure of the WriteGroup service primitive is shown in Table 15-18. The terminology and symbology used in this table are explained in Clause 5.6.

Table 15-18. Structure of WriteGroup Service Primitives

Parameter Name	Req	Ind
Argument	M	M(=)
Group Number	M	M(=)
Write Priority	M	M(=)
Change List	M	M(=)
Inhibit Delay	U	U(=)

15.11.1.1 Argument

The 'Argument' parameter shall convey the parameters for the WriteGroup unconfirmed service request.

15.11.1.1.1 Group Number

This parameter is an unsigned integer in the range 1 – 4294967295 that represents the control group to be affected by this request. Control group zero shall never be used and shall be reserved. WriteGroup service requests containing a zero value for 'Group Number' shall be ignored.

15.11.1.1.2 Write Priority

This parameter is an unsigned integer in the range 1..16 that represents the priority for writing that shall apply to any channel value changes that result in writes to properties of BACnet objects.

15.11.1.1.3 Change List

This parameter shall specify a list of BACnetGroupChannelValue values containing at least one value. The list consists of (channel number, overridingPriority, value) tuples representing each channel number whose value is to be updated. Channel numbers shall range from 0 to 65535 where the channel number corresponds directly to the Channel_Number property of a Channel object. The optional overridingPriority allows specific values to be written with some priority other than that specified by Write_Priority property. BACnetGroupChannelValue values convey BACnetChannelValue values that are any primitive application datatype or BACnetLightingCommand. The NULL value represents 'relinquish control' as with commandable object properties. See Clause 19.

15.11.1.1.4 Inhibit Delay

This optional parameter shall specify whether Channel objects whose Allow_Group_Delay_Inhibit properties have a value of TRUE shall inhibit any execution delay specified in their Execution_Delay property. If the 'Inhibit Delay' parameter is absent or FALSE, then execution delay(s) shall occur according to the Execution_Delay property.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

15.11.2 WriteGroup Service Procedure

Since this is an unconfirmed service, no response primitives are expected. The sending BACnet-user shall transmit the WriteGroup unconfirmed request using a unicast, multicast or broadcast address. A broadcast may be sent locally, to a remote BACnet network number, or using the global BACnet network address.

If the 'Group Number' is non-zero, and the receiving BACnet-user has been configured to be a member of the control group 'Group Number' by virtue of having that group number in any of the array elements of the Control_Groups property of any of its Channel objects, then for each (channel number, overridingPriority, value) tuple provided in the 'Change List' parameter, the receiving BACnet-user shall attempt to write to the Channel object(s) whose Channel_Number property(s) match that channel number with the indicated value. If no Channel object's Channel_Number property matches the provided channel number, then that value shall be ignored.

If the optional field overridingPriority is provided, it shall specify the priority for writing the value. Otherwise the 'Write Priority' parameter shall specify the priority for writing.

If a BACnetGroupChannelValue specifies a NULL value, it shall serve the same function as if NULL had been used with WriteProperty.

The failure of any particular write shall not prevent the remaining writes from taking place.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16 REMOTE DEVICE MANAGEMENT SERVICES

16.1 DeviceCommunicationControl Service

The DeviceCommunicationControl service is used by a client BACnet-user to instruct a remote device to stop initiating BACnet services on the communication network or internetwork for a specified duration of time. This service is primarily used by a human operator for diagnostic purposes. A password may be required from the client BACnet-user prior to executing the service. The time duration can be set to "indefinite," meaning communication must be re-enabled by a DeviceCommunicationControl or, if supported, ReinitializeDevice service, not by time.

16.1.1 Structure

The structure of the DeviceCommunicationControl service primitives is shown in Table 16-1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-1. Structure of DeviceCommunicationControl Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Time Duration	U	U(=)		
Enable/Disable	M	M(=)		
Password	U	U(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)

16.1.1.1 Argument

This parameter shall convey the parameters for the DeviceCommunicationControl confirmed service request.

16.1.1.1.1 Time Duration

This optional parameter, of type Unsigned16, indicates the number of minutes that the remote device shall not initiate BACnet services. If the 'Time Duration' parameter is not present, then the time duration shall be considered indefinite, meaning that only an explicit DeviceCommunicationControl or ReinitializeDevice APDU shall enable communications. The 'Time Duration' parameter shall be ignored and the time period considered to be indefinite if the 'Enable/Disable' parameter has a value of ENABLE.

If the responding BACnet-user does not have a clock and the time duration is not indefinite, then the request shall be considered invalid and the responding BACnet-user shall issue a Result(-) response.

16.1.1.1.2 Enable/Disable

This parameter is an enumeration that may take on the values ENABLE, DISABLE, or DISABLE_INITIATION. It is used to indicate whether the responding BACnet-user is to enable all, disable initiation, or disable all communications on the network interface. When this parameter has a value of ENABLE, communications shall be enabled. When this parameter has a value of DISABLE, network communications shall be disabled as described in the DeviceCommunicationControl service procedure. When this parameter has a value of DISABLE_INITIATION, the initiation of communications shall be disabled as described in the DeviceCommunicationControl service procedure.

16.1.1.1.3 Password

This optional parameter shall be a CharacterString of up to 20 characters. For those devices that require password protection, the service shall be denied if the parameter is absent or if the password is incorrect. For those devices that do not require a password, this parameter shall be ignored.

16.1.1.2 Result(+)

This parameter shall indicate that the service request succeeded.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.1.1.3 Result(-)

This parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

16.1.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The password is invalid or absent when one is required.	SECURITY	PASSWORD_FAILURE
The device does not have a clock and the 'Time Duration' parameter is not set to "indefinite".	SERVICES	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
If the request is valid and the 'Enable/Disable' parameter is the deprecated value DISABLE	SERVICES	SERVICE_REQUEST_DENIED

16.1.2 Service Procedure

After verifying the validity of the request, including the 'Time Duration' and 'Password' parameters, the responding BACnet-user shall respond with a 'Result(+)' service primitive. If the request is valid and the 'Enable/Disable' parameter is DISABLE_INITIATION, the responding BACnet-user shall discontinue the initiation of messages except for I-Am requests issued in accordance with the Who-Is service procedure and ConfirmedAuditNotification and UnconfirmedAuditNotification requests. Communication shall be disabled in this manner until either the 'Time Duration' has expired or a valid DeviceCommunicationControl (with 'Enable/Disable' = ENABLE) or, if supported, a valid ReinitializeDevice (with 'Reinitialized State of Device' = WARMSTART or COLDSTART) message is received.

If the responding BACnet-user does not have a clock and the 'Time Duration' parameter is not set to "indefinite," the APDU shall be ignored and a 'Result(-)' service primitive shall be issued. If the 'Password' parameter is invalid or absent when a password is required, the APDU shall be ignored and an Error-PDU with 'Error Class' = SECURITY and 'Error Code' = PASSWORD_FAILURE shall be issued.

If the request is valid and the 'Enable/Disable' parameter is the deprecated value DISABLE, the request shall be ignored and an Error-PDU with 'Error Class' = SERVICES and 'Error Code' = SERVICE_REQUEST_DENIED shall be issued.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.2 ConfirmedPrivateTransfer Service

The ConfirmedPrivateTransfer is used by a client BACnet-user to invoke proprietary or non-standard services in a remote device. The specific proprietary services that may be provided by a given device are not defined by this standard. The PrivateTransfer services provide a mechanism for specifying a particular proprietary service in a standardized manner. The only required parameters for these services are a vendor identification code and a service number. Additional parameters may be supplied for each service if required. The form and content of these additional parameters, if any, are not defined by this standard. The vendor identification code and service number together serve to unambiguously identify the intended purpose of the information conveyed by the remainder of the APDU or the service to be performed by the remote device based on parameters in the remainder of the APDU.

The vendor identification code is a unique code assigned to the vendor by ASHRAE. The mechanism for administering these codes is not defined in this standard. See Clause 23.

16.2.1 ConfirmedPrivateTransfer Service Structure

The structure of the ConfirmedPrivateTransfer service primitives is shown in Table 16-2. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-2. Structure of ConfirmedPrivateTransfer Service Primitives

Parameter Name	Req	Ind	Rsp	Conf
Argument	M	M(=)		
Vendor ID	M	M(=)		
Service Number	M	M(=)		
Service Parameters	U	U(=)		
Result(+)			S	S(=)
Vendor ID			M	M(=)
Service Number			M	M(=)
Result Block			C	C(=)
Result(-)			S	S(=)
Error Type			M	M(=)
Vendor ID			M	M(=)
Service Number			M	M(=)
Error Parameters			U	U(=)

16.2.1.1 Argument

This parameter shall convey the parameters for the ConfirmedPrivateTransfer confirmed service request.

16.2.1.1.1 Vendor ID

This parameter, of type Unsigned, shall specify the unique vendor identification code for the type of vendor-proprietary service to be performed.

16.2.1.1.2 Service Number

This parameter, of type Unsigned, shall specify the desired service to be performed.

16.2.1.1.3 Service Parameters

This optional parameter shall convey additional parameters for the service specified by 'Vendor ID' and 'Service Number'. The datatype and interpretation of these parameters is a local matter.

16.2.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result indicates that the request APDU was received and the recipient was able to perform the indicated proprietary service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.2.1.2.1 Vendor ID

This parameter, of type Unsigned, shall specify the unique vendor identification code for the vendor-proprietary service for which this is the result.

16.2.1.2.2 Service Number

This parameter, of type Unsigned, shall indicate the proprietary service for which this is the result.

16.2.1.2.3 Result Block

This conditional parameter, of type list of ANY, shall convey any additional results from the execution of the requested service. Interpretation of these results is a local matter.

16.2.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request has failed. The reason for failure shall be specified by the 'Error Type' parameter.

16.2.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

16.2.1.3.2 Vendor ID

This parameter, of type Unsigned, shall specify the unique vendor identification code for the vendor-proprietary service for which this error is the result.

16.2.1.3.3 Service Number

This parameter, of type Unsigned, shall indicate the proprietary service for which this error is the result.

16.2.1.3.4 Error Parameters

This optional parameter shall convey any additional error results from the execution of the requested service. Interpretation of these results is a local matter.

16.2.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to perform the specified proprietary service request. If successful, a 'Result(+)' response primitive shall be issued. If the request fails, a 'Result(-)' response primitive shall be issued.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.3 UnconfirmedPrivateTransfer Service

The UnconfirmedPrivateTransfer is used by a client BACnet-user to invoke proprietary or non-standard services in a remote device. The specific proprietary services that may be provided by a given device are not defined by this standard. The PrivateTransfer services provide a mechanism for specifying a particular proprietary service in a standardized manner. The only required parameters for these services are a vendor identification code and a service number. Additional parameters may be supplied for each service if required. The form and content of these additional parameters, if any, are not defined by this standard. The vendor identification code and service number together serve to unambiguously identify the intended purpose of the information conveyed by the remainder of the APDU or the service to be performed by the remote device based on parameters in the remainder of the APDU.

The vendor identification code is a unique code assigned to the vendor by ASHRAE. The mechanism for administering these codes is not defined in this standard. See Clause 23.

16.3.1 UnconfirmedPrivateTransfer Service Structure

The structure of the UnconfirmedPrivateTransfer service primitive is shown in Table 16-3. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-3. Structure of UnconfirmedPrivateTransfer Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Vendor ID	M	M(=)
Service Number	M	M(=)
Service Parameters	U	U(=)

16.3.1.1 Argument

This parameter shall convey the parameters for the UnconfirmedPrivateTransfer confirmed service request.

16.3.1.1.1 Vendor ID

This parameter, of type Unsigned, shall specify the unique vendor identification code for the type of vendor-proprietary service to be performed.

16.3.1.1.2 Service Number

This parameter, of type Unsigned, shall specify the desired service to be performed.

16.3.1.1.3 Service Parameters

This optional parameter shall convey additional parameters for the service specified by 'Vendor ID' and 'ServiceNumber'. The datatype and interpretation of these parameters is a local matter.

16.3.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. Actions taken in response to this service request are a local matter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.4 ReinitializeDevice Service

The ReinitializeDevice service is used by a client BACnet-user to instruct a remote device to reboot itself (cold start), reset itself to some predefined initial state (warm start), to activate network port object changes, or to control the backup or restore procedure. Resetting or rebooting a device is primarily initiated by a human operator for diagnostic purposes. Use of this service during the backup or restore procedure is usually initiated on behalf of the user by the device controlling the backup or restore. Due to the sensitive nature of this service, a password may be required by the responding BACnet-user prior to executing the service.

A BACnet device may support the ReinitializeDevice service by supporting only the restart choices COLDSTART and WARMSTART. Support for the backup and restore features of this service is claimed separately. If the device supports a Network Port Object using the pending changes functionality, then the restart choice ACTIVATE_CHANGES shall also be supported.

16.4.1 Structure

The structure of the ReinitializeDevice service primitives is shown in Table 16-4. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-4. Structure of ReinitializeDevice Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Reinitialized State of Device	M	M(=)		
Password	U	U(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)

16.4.1.1 Argument

This parameter shall convey the parameters for the ReinitializeDevice confirmed service request.

16.4.1.1.1 Reinitialized State of Device

This parameter allows the client user to specify the desired state of the device after its reinitialization. The value of the parameter may be one of COLDSTART, WARMSTART, ACTIVATE_CHANGES, STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, or ABORTRESTORE.. WARMSTART shall mean to reboot the device and start over, retaining all data and programs that would normally be retained during a brief power outage. The precise interpretation of COLDSTART shall be defined by the vendor.

If the value of the parameter is ACTIVATE_CHANGES or WARMSTART and the device is not ready due to a configuration procedure in progress, the request shall be considered invalid and the responding BACnet user shall issue a Result(-) response.

If the value of the parameter is one of STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, or ABORTRESTORE and communication has been disabled due to receipt of a DeviceCommunicationControl request with 'Enable/Disable' equal to DISABLE, the request shall be considered invalid and the responding BACnet user shall issue a Result(-) response.

The use of the backup and restore commands are defined in Clause 19.1.

16.4.1.1.2 Password

This optional parameter shall be a CharacterString of up to 20 characters. For those devices that require the password as a protection, the service request shall be denied if the parameter is absent or if the password is incorrect. For those devices that do not require a password, this parameter shall be ignored.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.4.1.2 Result(+)

This parameter shall indicate that the service request succeeded.

16.4.1.3 Result(-)

This parameter shall indicate that the service request has failed. The reason for the failure shall be specified by the 'Error Type' parameter.

16.4.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The password is invalid or absent when one is required.	SECURITY	PASSWORD_FAILURE
The device is in the process of being configured.	DEVICE	CONFIGURATION_IN_PROGRESS
Communication has been disabled due to receipt of a DeviceCommunicationControl request.	SERVICES	COMMUNICATION_DISABLED
The validation of network port object property values fails.	PROPERTY	INVALID_CONFIGURATION_DATA

16.4.2 Service Procedure

After verifying the validity of the request, including the 'Reinitialized State of Device' and 'Password' parameters, the responding BACnet-user shall pre-empt all other outstanding requests and respond with a 'Result(+)' primitive. If the request is valid and 'Reinitialized State of Device' is WARMSTART or COLDSTART, then the responding BACnet-user shall immediately proceed to perform any applicable shut-down procedures prior to reinitializing the device as specified by the requesting BACnet-user in the request.

If 'Reinitialized State of Device' is WARMSTART and the device is not ready due to its initial characterization being in progress, a 'Result (-)' response primitive shall be issued.

If 'Reinitialized State of Device' is one of STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, or ABORTRESTORE and communication has been disabled due to receipt of a DeviceCommunicationControl request with 'Enable/Disable' equal to DISABLE, the responding BACnet user shall respond with a Result(-) primitive. Otherwise, the responding BACnet user shall behave as described in Clause 19.1.

If the password is invalid or is absent when one is required, an Error-PDU with 'Error Class' of SECURITY and 'Error Code' of PASSWORD_FAILURE shall be issued.

See Clause 12.56 for a description of 'Reinitialized State of Device' when its value is ACTIVATE_CHANGES.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.5 ConfirmedTextMessage Service

The ConfirmedTextMessage service is used by a client BACnet-user to send a text message to another BACnet device. This service is not a broadcast or multicast service. This service may be used in cases when confirmation that the text message was received is required. The confirmation does not guarantee that a human operator has seen the message. Messages may be prioritized into normal or urgent categories. In addition, a given text message may be optionally classified by a numeric class code or class identification string. This classification may be used by the receiving BACnet device to determine how to handle the text message. For example, the message class might indicate a particular output device on which to print text or a set of actions to take when the text is received. In any case, the interpretation of the class is a local matter.

16.5.1 ConfirmedTextMessage Service Structure

The structure of the ConfirmedTextMessage service primitives is shown in Table 16-5. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-5. Structure of ConfirmedTextMessage Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Text Message Source Device	M	M(=)		
Message Class	U	U(=)		
Message Priority	M	M(=)		
Message	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

16.5.1.1 Argument

This parameter shall convey the parameters for the ConfirmedTextMessage service request.

16.5.1.1.1 Text Message Source Device

This parameter, of type BACnetObjectIdentifier, shall convey the value of the Object_Identifier property of the Device object of the device that initiated this text message.

16.5.1.1.2 Message Class

This parameter, if present, shall indicate the class of the received message. The datatype of this parameter shall be a choice of Unsigned or CharacterString. The interpretation of the meaning of any particular value for this parameter shall be a local matter.

16.5.1.1.3 Message Priority

This parameter, of type ENUMERATED, shall indicate the priority for message handling:

{NORMAL, URGENT}

16.5.1.1.4 Message

This parameter, of type CharacterString, shall be used to convey the text message.

16.5.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the requested service has succeeded.

16.5.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the requested service has failed. The reason for the failure shall be specified by the 'Error Type' parameter.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.5.1.3.1 Error Type

This parameter shall consist of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

16.5.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall take whatever local actions have been assigned to the indicated 'Message Class' and issue a 'Result(+)' service primitive. If the service request cannot be executed, a 'Result(-)'service primitive shall be issued indicating the encountered error.

Other than the requirement to return a success or failure response, actions taken in response to this notification are a local matter. However, typically the receiving device would take the text specified by the 'Message' parameter and display, print, or file it according to the classification specified by the 'Message Class' parameter. If the 'Message Class' parameter is omitted, then some general class might be assumed. If 'Message Priority' is URGENT, then clearly the messages should be considered as more important than existing NORMAL messages, which may be awaiting printing or some other action.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.6 UnconfirmedTextMessage Service

The UnconfirmedTextMessage service is used by a client BACnet-user to send a text message to one or more BACnet devices. This service may be broadcast, multicast, or addressed to a single recipient. This service may be used in cases where confirmation that the text message was received is not required. Messages may be prioritized into normal or urgent categories. In addition, a given text message may optionally be classified by a numeric class code or class identification string. This classification may be used by receiving BACnet devices to determine how to handle the text message. For example, the message class might indicate a particular output device on which to print text or a set of actions to take when the text message is received. In any case, the interpretation of the class is a local matter.

16.6.1 UnconfirmedTextMessage Service Structure

The structure of the UnconfirmedTextMessage service primitive is shown in Table 16-6. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-6. Structure of UnconfirmedTextMessage Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Text Message Source Device	M	M(=)
Message Class	U	U(=)
Message Priority	M	M(=)
Message	M	M(=)

16.6.1.1 Argument

The 'Argument' parameter shall convey the parameters for the UnconfirmedTextMessage service request.

16.6.1.1.1 Text Message Source Device

This parameter, of type BACnetObjectIdentifier, shall convey the value of the Object_Identifier property of the Device object of the device that initiated this text message.

16.6.1.1.2 Message Class

This parameter, if present, shall indicate the classification of the received message. The datatype of this parameter shall be a choice of Unsigned or CharacterString. The interpretation of the meaning of any particular value for this parameter shall be a local matter.

16.6.1.1.3 Message Priority

This parameter, of type ENUMERATED, shall indicate the priority for message handling:

{NORMAL, URGENT}

16.6.1.1.4 Message

This parameter, of type CharacterString, shall be used to convey the text message.

16.6.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. Actions taken in response to this service request are a local matter. However, typically the receiving device(s) would take the text block specified by the 'Message' parameter and display or print or file them according to the classification specified by the 'Message Class' parameter. If the 'Message Class' parameter is omitted, then some general class might be assumed. If 'Message Priority' is URGENT, then clearly the messages should be considered as more important than existing NORMAL messages, which may be awaiting printing or some other action.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.7 TimeSynchronization Service

The TimeSynchronization service is used by a requesting BACnet-user to notify a remote device of the correct current time. This service may be broadcast, multicast, or addressed to a single recipient. Its purpose is to notify recipients of the correct current time so that devices may synchronize their internal clocks with one another.

16.7.1 Structure

The structure of the TimeSynchronization service primitive is shown in Table 16-7. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-7. Structure of TimeSynchronization Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Time	M	M(=)

16.7.1.1 Argument

The 'Argument' parameter shall convey the parameters for the TimeSynchronization service request.

16.7.1.1.1 Time

This parameter, of type BACnetDateTime, shall convey the current date and time as determined by the clock in the device issuing the service request. This parameter shall contain a specific datetime value.

16.7.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. A device receiving a TimeSynchronization service indication shall update its local representation of time. This change shall be reflected in the Local_Time and Local_Date properties of the Device object.

No restrictions on the use of this service exist when it is invoked at the request of an operator. Otherwise, the use of this service is controlled by the value of the Time_Synchronization_Recipients property in the Device object. When the Time_Synchronization_Recipients list is of length zero, a device may not automatically send a TimeSynchronization request. When Time_Synchronization_Recipients list is of length one or more, a device may automatically send a TimeSynchronization request but only to the devices or addresses contained in the Time_Synchronization_Recipients list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.8 UTCTimeSynchronization Service

The UTCTimeSynchronization service is used by a requesting BACnet-user to notify one or more remote devices of the correct Universal Time Coordinated (UTC). This service may be broadcast, multicast, or addressed to a single recipient. Its purpose is to notify recipients of the correct UTC so that devices may synchronize their internal clocks with one another.

16.8.1 Structure

The structure of the UTCTimeSynchronization service primitive is shown in Table 16-8. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-8. Structure of UTCTimeSynchronization Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Time	M	M(=)

16.8.1.1 Argument

The 'Argument' parameter shall convey the parameters for the UTCTimeSynchronization service request.

16.8.1.1.1 Time

This parameter, of type BACnetDateTime, shall convey the UTC date and time. This parameter shall contain a specific datetime value.

16.8.2 Service Procedure

Since this is an unconfirmed service, no response primitives are expected. A device receiving a UTCTimeSynchronization service indication shall update its local representation of time and date by subtracting the value of the 'UTC_Offset' property of the Device object from the 'Time' parameter and taking the 'Daylight_Savings_Status' property of the Device object into account as appropriate to the locality. This change shall be reflected in the Local_Time and Local_Date properties of the Device object.

No restrictions on the use of this service exist when it is invoked at the request of an operator. Otherwise, the initiation of this service by a device is controlled by the value of the UTC_Time_Synchronization_Recipients property in the Device object. When the UTC_Time_Synchronization_Recipients list is of length zero, a device may not automatically send a UTCTimeSynchronization request. When UTC_Time_Synchronization_Recipients list is of length one or more, a device may automatically send a UTCTimeSynchronization request but only to the devices or addresses contained in the UTC_Time_Synchronization_Recipients list.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.9 Who-Has and I-Have Services

The Who-Has service is used by a sending BACnet-user to identify the Device object identifiers and network addresses of other BACnet devices whose local databases contain an object with a given Object_Name or a given Object_Identifier. The I-Have service is used to respond to Who-Has service requests or to advertise the existence of an object with a given Object_Name or Object_Identifier. The I-Have service request may be issued at any time and does not need to be preceded by the receipt of a Who-Has service request. The Who-Has and I-Have services are unconfirmed services.

16.9.1 Who-Has Service Structure

The structure of the Who-Has service primitive is shown in Table 16-9. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-9. Structure of Who-Has Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Instance Range Low Limit	U	U(=)
Device Instance Range High Limit	U	U(=)
Object Identifier	S	S(=)
Object Name	S	S(=)

16.9.1.1 Argument

The 'Argument' parameter shall convey the parameters for the Who-Has unconfirmed service request.

16.9.1.1.1 Device Instance Range Low Limit

This parameter is an unsigned integer in the range 0 - 4194303. In conjunction with the 'Device Instance Range High Limit' parameter, it defines the devices that are qualified to respond with an I-Have service request if the 'Object Identifier' or 'Object Name' criteria are satisfied as described in Clauses 16.9.1.1.3 and 16.9.1.1.4. If the 'Device Instance Range Low Limit' parameter is present, then the 'Device Instance Range High Limit' parameter shall also be present, and only those devices whose Device Object_Identifier instance number falls within the range 'Device Instance Range Low Limit' ≤ Object_Identifier Instance Number ≤ 'Device Instance Range High Limit' shall be qualified to respond. The value of the 'Device Instance Range Low Limit' shall be less than or equal to the value of the 'Device Instance Range High Limit'. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are omitted, then all devices that receive this message are qualified to respond with an I-Have service request.

16.9.1.1.2 Device Instance Range High Limit

This parameter is an unsigned integer in the range 0 - 4194303. In conjunction with the 'Device Instance Range Low Limit' parameter, it defines the devices that are qualified to respond with an I-Have service request if the 'Object Identifier' or 'Object Name' criteria are satisfied as described in Clauses 16.9.1.1.3 and 16.9.1.1.4. If the 'Device Instance Range High Limit' parameter is present, then the 'Device Instance Range Low Limit' parameter shall also be present, and only those devices whose Device Object_Identifier instance number falls within the range 'Device Instance Range Low Limit' ≤ Object_Identifier Instance Number ≤ 'Device Instance Range High Limit' shall be qualified to respond. The value of the 'Device Instance Range High Limit' shall be greater than or equal to the value of the 'Device Instance Range Low Limit'. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are omitted, then all devices that receive this message are qualified to respond with an I-Have service request.

16.9.1.1.3 Object Identifier

The 'Object Identifier' parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of the object that is to be located. If the 'Object Identifier' parameter is omitted, then the 'Object Name' parameter shall be present. If the 'Object Identifier' parameter is present, then only those devices that contain an object with an Object_Identifier property value matching the 'Object Identifier' parameter, which are qualified to respond as described in Clauses 16.9.1.1.1 and 16.9.1.1.2, shall respond with an I-Have service request.

16.9.1.1.4 Object Name

The 'Object Name' parameter, of type CharacterString, shall convey the value of the Object_Name property of the object that is to be located. If the 'Object Name' parameter is omitted, then the 'Object Identifier' parameter shall be present. If the 'Object Name' parameter is present, then only those devices that contain an object with an Object_Name property

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

value matching the 'Object Name' parameter, which are qualified to respond as described in Clauses 16.9.1.1.1 and 16.9.1.1.2, shall respond with an I-Have service request.

16.9.2 Service Procedure

The sending BACnet-user shall transmit the Who-Has unconfirmed request, normally using a broadcast address. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are present, then only those receiving BACnet-users whose Device Object_Identifier instance number falls in the range 'Device Instance Range Low Limit' ≤ Object_Identifier Instance Number ≤ 'Device Instance Range High Limit' shall be qualified to respond. If the 'Object Name' parameter is present, then only those qualified receiving BACnet-users that contain an object with an Object_Name property value matching the 'Object Name' parameter shall respond with an I-Have service request. If the 'Object Identifier' parameter is present, then only those qualified receiving BACnet-users that contain an object with an Object_Identifier property value matching the 'Object Identifier' parameter shall respond with an I-Have service request.

16.9.3 I-Have Service Structure

The structure of the I-Have service primitive is shown in Table 16-10. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-10. Structure of I-Have Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Identifier	M	M(=)
Object Identifier	M	M(=)
Object Name	M	M(=)

16.9.3.1 Argument

The 'Argument' parameter shall convey the parameters for the I-Have unconfirmed service request.

16.9.3.1.1 Device Identifier

The 'Device Identifier' parameter, of type BACnetObjectIdentifier, is the Device Object_Identifier of the device initiating the I-Have service request.

16.9.3.1.2 Object Identifier

The 'Object Identifier' parameter, of type BACnetObjectIdentifier, shall convey the Object_Identifier of the object that is being advertised as located in the initiating device. This identifier shall correspond to the value of the Object_Identifier property of the object being advertised.

16.9.3.1.3 Object Name

The 'Object Name' parameter, of type CharacterString, shall convey the name of the object that is being advertised as located in the initiating device. This name shall correspond to the value of the Object_Name property of the object being advertised.

16.9.4 Service Procedure

The sending BACnet-user shall broadcast or unicast the I-Have unconfirmed request. If the I-Have is broadcast, this broadcast may be on the local network only, a remote network only, or globally on all networks at the discretion of the application. If the I-Have is being transmitted in response to a previously received Who-Has, then the I-Have shall be transmitted in such a manner that the BACnet-user that sent the Who-Has will receive the resulting I-Have. Since the request is unconfirmed, no further action is required. A BACnet-user may issue an I-Have service request at any time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.10 Who-Is and I-Am Services

The Who-Is service is used by a sending BACnet-user to determine the Device object identifier, the network address, or both, of other BACnet devices that share the same internetwork. The Who-Is service is an unconfirmed service. The Who-Is service may be used to determine the Device object identifier and network addresses of all devices on the network, or to determine the network address of a specific device whose Device object identifier is known, but whose address is not. The I-Am service is also an unconfirmed service. The I-Am service is used to respond to Who-Is service requests. However, the I-Am service request may be issued at any time. It does not need to be preceded by the receipt of a Who-Is service request. In particular, a device may wish to broadcast an I-Am service request when it powers up. The network address is derived either from the MAC address associated with the I-Am service request, if the device issuing the request is on the local network, or from the NPCI if the device is on a remote network.

The Who-Is service may be used to discover devices supporting the Who-Am-I service that also require Device ID configuration. See Clause 19.

16.10.1 Who-Is Service Structure

The structure of the Who-Is service primitive is shown in Table 16-11. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-11. Structure of Who-Is Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Device Instance Range Low Limit	U	U(=)
Device Instance Range High Limit	U	U(=)

16.10.1.1 Argument

The 'Argument' parameter shall convey the parameters for the Who-Is unconfirmed service request.

16.10.1.1.1 Device Instance Range Low Limit

This parameter is an unsigned integer in the range 0 - 4194303. In conjunction with the 'Device Instance Range High Limit' parameter, it defines the devices that are qualified to respond with an I-Am service request. If the 'Device Instance Range Low Limit' parameter is present, then the 'Device Instance Range High Limit' parameter shall also be present, and only those devices whose Device Object_Identifier instance number falls within the range greater than or equal to 'Device Instance Range Low Limit' and less than or equal to 'Device Instance Range High Limit' shall be qualified to respond. The value of the 'Device Instance Range Low Limit' shall be less than or equal to the value of the 'Device Instance Range High Limit'. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are omitted, then all devices that receive this message are qualified to respond with an I-Am service request.

16.10.1.1.2 Device Instance Range High Limit

This parameter is an unsigned integer in the range 0 - 4194303. In conjunction with the 'Device Instance Range Low Limit' parameter, it defines the devices that are qualified to respond with an I-Am service request. If the 'Device Instance Range High Limit' parameter is present, then the 'Device Instance Range Low Limit' parameter shall also be present, and only those devices whose Device Object_Identifier instance number falls within the range greater than or equal to 'Device Instance Range Low Limit' and less than or equal to 'Device Instance Range High Limit' shall be qualified to respond. The value of the 'Device Instance Range High Limit' shall be greater than or equal to the value of the 'Device Instance Range Low Limit'. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are omitted, then all devices that receive this message are qualified to respond with an I-Am service request.

16.10.2 Service Procedure

The sending BACnet-user shall transmit the Who-Is unconfirmed request, normally using a broadcast address. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are omitted, then all receiving BACnet-users shall return their Device Object_Identifier in individual responses using the I-Am service. If the 'Device Instance Range Low Limit' and 'Device Instance Range High Limit' parameters are present, then only those receiving BACnet-users whose Device Object_Identifier instance number falls within the range greater than or equal to 'Device Instance Range Low Limit' and less than or equal to 'Device Instance Range High Limit' shall return their Device Object_Identifier using the I-Am service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the receiving BACnet-user has a Slave_Proxy_Enable property and the Slave_Proxy_Enable for the receiving port is TRUE, then the BACnet-user shall respond with an I-Am unconfirmed request for each of the slave devices on the MS/TP network that are present in the Slave_Address_Binding property and that match the device range parameters. The I-Am unconfirmed requests that are generated shall be generated as if the slave device originated the service. If the I-Am unconfirmed request is to be placed onto the MS/TP network on which the slave device resides, then the MAC address included in the packet shall be that of the slave device. In the case where the I-Am unconfirmed request is to be placed onto a network other than that on which the slave device resides, then the network layer shall contain SLEN and SNET filled in with the slave device's MAC address as if it were routing a packet originally generated by the slave device.

16.10.3 I-Am Service Structure

The structure of the I-Am service primitive is shown in Table 16-12. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-12. Structure of I-Am Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
I-Am Device Identifier	M	M(=)
Max APDU Length Accepted	M	M(=)
Segmentation Supported	M	M(=)
Vendor ID	M	M(=)

16.10.3.1 Argument

The 'Argument' parameter shall convey the parameters for the I-Am unconfirmed service request.

16.10.3.1.1 I-Am Device Identifier

The 'I-Am Device Identifier' parameter, of type BACnetObjectIdentifier, is the Device Object_Identifier of the device initiating the I-Am service request.

16.10.3.1.2 Max APDU Length Accepted

This parameter, of type Unsigned, shall convey the maximum number of octets that may be contained in a single, indivisible APDU. The value of this parameter shall be the same as the value of the Max_APDU_Length_Accepted property of the Device object. See Clause 12.11.18.

16.10.3.1.3 Segmentation Supported

This parameter, of type BACnetSegmentation, conveys the capabilities of the device initiating the I-Am service request with respect to processing segmented messages. The value of this parameter shall be the same as the value of the Segmentation_Supported property of the Device object. See Clause 12.11.19.

16.10.3.1.4 Vendor ID

This parameter, of type Unsigned16, shall convey the identity of the vendor who manufactured the device initiating the I-Am service request. The value of this parameter shall be the same as the value of the Vendor_Identifier property of the Device object. See Clause 12.11.6 and Clause 23.

16.10.4 Service Procedure

The sending BACnet-user shall broadcast or unicast the I-Am unconfirmed request. If the I-Am is broadcast, this broadcast may be on the local network only, a remote network only, or globally on all networks at the discretion of the application. If the I-Am is being sent in response to a previously received Who-Is, then the I-Am shall be sent in such a manner that the BACnet-user that sent the Who-Is will receive the resulting I-Am. Since the request is unconfirmed, no further action is required. A BACnet-user may issue an I-Am service request at any time.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.11 Who-Am-I and You-Are Services

The You-Are service is used by a client BACnet-user to configure the MAC address and BACnet Device object instance number in a remote device. The You-Are service provides a mechanism for specifying device identifier values across a network in a standardized manner. The vendor identification, model name, and serial number parameters together serve to unambiguously identify the remote device. Additional parameters are supplied for the MAC address or the Device object identifier, or both.

The Who-Am-I service is used by a sending BACnet-user to indicate that it requires identity configuration via the You-Are service. The Who-Am-I service provides a mechanism for requesting device identifier values across a network in a standardized manner. The vendor identification, model name, and serial number parameters are included in the request to unambiguously identify this device. The Who-Am-I service is also used to respond to a Who-Is service request that uses the Device Object_Identifier instance number of 4194303.

16.11.1 Who-Am-I Service Structure

The structure of the Who-Am-I service primitive is shown in Table 16-13. The terminology and symbology used in this table are explained in Clause 5.6.

Table 16-13. Structure of Who-Am-I Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Vendor ID	M	M(=)
Model Name	M	M(=)
Serial Number	M	M(=)

16.11.1.1 Argument

The 'Argument' parameter shall convey the parameters for the WhoAm-I service request.

16.11.1.2 Vendor ID

This parameter, of type Unsigned16, shall convey the identity of the vendor of the device initiating the Who-Am-I service request. The value of this parameter shall be the same as the value of the Vendor_Identifier property of the Device object. See 12.11.6 and Clause 23.

16.11.1.3 Model Name

This parameter, of type CharacterString, shall specify the model name of the device initiating the Who-Am-I service request. The value of this parameter shall be the same as the value of the Model_Name property of the Device object. See 12.11.7.

16.11.1.3.1 Serial Number

This parameter, of type CharacterString, shall specify the serial identifier of the device initiating the Who-Am-I service request. The value of this parameter shall be the same as the value of the Serial_Number property in the of the Device object. See 12.11.56.

16.11.2 Service Procedure

The sending BACnet-user shall broadcast or unicast the Who-Am-I unconfirmed request. If the Who-Am-I is broadcast, this broadcast may be on the local network only, a remote network only, or globally on all networks at the discretion of the application.

If the Who-Am-I is being sent in response to a previously received Who-Is, then the Who-Am-I shall be sent in such a manner that the BACnet-user that sent the Who-Is will receive the resulting Who-Am-I. The sending BACnet-user Device Object_Identifier instance number shall be 4194303 when determining its inclusion in the Who-Is range. See Clause 19.7.

Since the request is unconfirmed, no further action is required. The BACnet-user may issue a Who-Am-I service request at any time, but shall not repeat the service request more frequently than every 5 minutes except when the Who-Am-I is being sent in response to a previously received Who-Is, or in response to manual intervention.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

16.11.3 You-Are Service Structure

The structure of the You-Are service primitive is shown in Table 16-14. The terminology and symbology used in this table are explained in 5.6.

Table 16-14. Structure of You-Are Service Primitive

Parameter Name	Req	Ind
Argument	M	M(=)
Vendor ID	M	M(=)
Model Name	M	M(=)
Serial Number	M	M(=)
Device Identifier	C	C(=)
Device MAC Address	C	C(=)

16.11.3.1 Argument

The 'Argument' parameter shall convey the parameters for the You-Are service request.

16.11.3.1.1 Vendor ID

This parameter, of type Unsigned16, shall specify the identity of the vendor of the device that is qualified to receive this YouAre service request. The value of this parameter shall be compared to the value of the Vendor_Identifier property in the Device object by recipients of the You-Are service request. See 12.11.6 and Clause 23.

16.11.3.1.2 Model Name

This parameter, of type CharacterString, shall specify the model name of the device qualified to receive the You-Are service request. The value of this parameter shall be compared to the value of the Model_Name property in the Device object by recipients of the You-Are service request. See 12.11.7.

16.11.3.1.3 Serial Number

This parameter, of type CharacterString, shall specify the serial number of the device qualified to receive the You-Are service request. The value of this parameter shall be compared to the value of the Serial_Number property in the Device object by recipients of the You-Are service request. See 12.11.56.

16.11.3.1.4 Device Identifier

This parameter, of type BACnetObjectIdentifier, is the Device Object_Identifier to be assigned in the qualified device. Either the 'Device Identifier', or 'Device MAC Address', or both shall be present.

16.11.3.1.5 Device MAC Address

This parameter, of type OctetString, defines the device MAC address that is to be configured in the qualified device. For qualified devices that use VMAC addresses as defined in H.7.2, this parameter shall be ignored. Either the 'Device Identifier', or 'Device MAC Address', or both shall be present.

16.11.4 Service Procedure

If the sending BACnet-user knows the MAC address of the receiving BACnet device, this service shall be unicast. Otherwise, this service shall be broadcast. The receiving BACnet-user, whose Device object Vendor_identifier property value matches 'Vendor ID', Device object Serial_Number property value matches the 'Serial Number', the Device object Model_Name property value matches the 'Model Name', and the 'Device MAC Address' is valid for the device, shall change its Device object Object_Identifier property and its device MAC address (if it is changeable) on the received port accordingly, based on the presence or absence of 'Device Identifier' or 'Device MAC Address'. After accepting the 'Device Identifier' or 'Device MAC Address' if provided, the device shall subsequently generate an I-Am, except when the receiving BACnet-user is an MS/TP slave node. The device is required to maintain the value of the Device Object_Identifier property and device MAC address across power failures or "restarts."

If the instance portion of the 'Device Identifier' is 4194303, then the device shall become an unconfigured device as defined by Clause 19.7.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17 VIRTUAL TERMINAL SERVICES

Virtual Terminal (VT) services are used by a client BACnet-user to establish a connection to an application program server in another BACnet device. The purpose of this connection is to facilitate the bi-directional exchange of character-oriented data. Normally, these services would be used to permit an application program in one BACnet device to act as a "terminal emulator" that interacts with an "operator interface" application program in another BACnet device.

These connections will be referred to here as VT-sessions. Once a VT-session is established, both the client application program and server application program will be referred to as VT-users.

The VT services provide the following features and services to the VT-user:

(a) the means to establish a VT-session between two peer VT-users for the purpose of enabling virtual terminal information exchange;
(b) the means to select between different VT-class types, including character repertoire and encoding;
(c) the means to control the integrity of the communication;
(d) the means to terminate the VT-session unilaterally;
(e) the means to exchange virtual terminal data.

17.1 Virtual Terminal Model

Each VT-session is a bi-directional connection between two peer application processes. Once a session is established between these peers, data are exchanged through the use of Virtual Terminal Queues (VTQ). The VTQs are first-in, first-out (FIFO) queues. The purpose of modeling the data flow between peer processes as FIFO queues is to isolate the implementation of the peer application process that is on each side of the VT-session from the other. Normally a human operator using a BACnet device will request the operator interface application program to establish a VT-session with an operator interface application program in another BACnet device. The VTQ model uses two FIFO queues to allow those operator interfaces, or other application programs that can provide simultaneous bi-directional interaction, to do so through the BACnet protocol.

Figure 17-1 shows a typical relationship between an operator interface application program and a physical device, such as a CRT terminal.

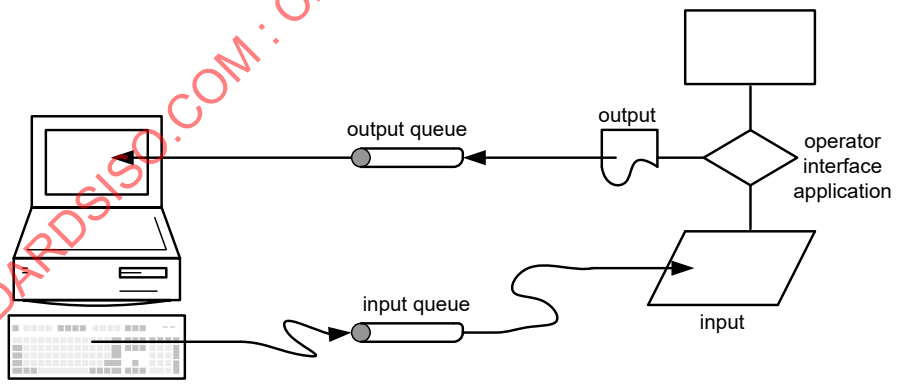


Figure 17-1. Relationship between an operator interface program and a physical device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Figure 17-2 shows how this model is extended using the VTQ concept.

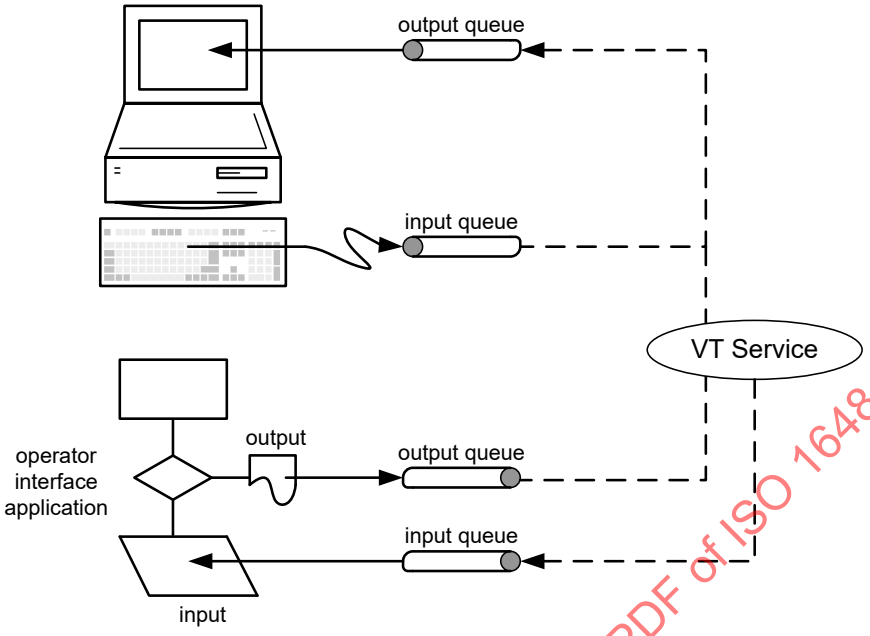


Figure 17-2. Extending the VT model to accommodate queues.

The peer processes at each end of a VT-session may not actually be agents for a physical device such as a CRT terminal. The VTQ model permits flexibility in the implementation of each BACnet device. There may, in fact, be several different processes that coordinate their use of VT Services within each BACnet device. For example, in a multi-window operator interface program, there may be several windows, each with its own interactive virtual terminal session to some other BACnet device. For this reason, each VT-session exists between two unique processes rather than between two BACnet devices. Each session, therefore, consists of two processes that act as agents for their respective application programs and their respective VTQs. These agent processes and their VTQs are called VT-Users. The model of this data flow is shown in Figure 17-3.

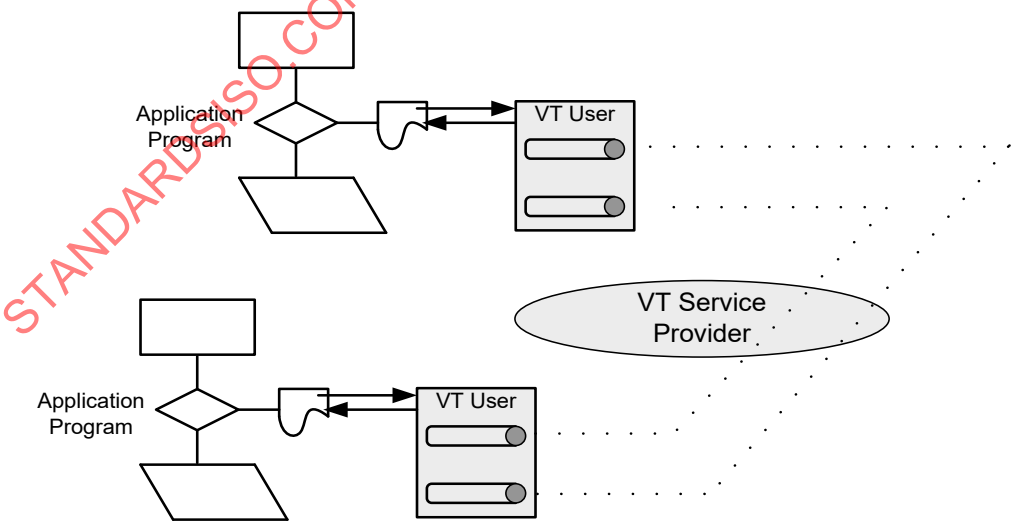


Figure 17-3. Virtual terminal data flow.

Once the virtual terminal session is established, character data are exchanged by the two peers through their respective VTQs. Normally, characters typed by a human operator would be passed directly to an input queue for forwarding to the peer's input queue, without any echoing by the local device to which the operator is connected. The peer application

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

program, upon receiving these characters from its input queue, would respond with characters placed in its output queue for forwarding back to the output queue of the operator device and so on. In particular, it shall be the responsibility of the operator interface application to generate all "screen" output, including carriage control and character echoing when appropriate.

Although the VT services model does provide a true peer-to-peer connection, as shown in Figure 17-3, a human operator typically uses one BACnet device to establish a virtual terminal connection to a second BACnet device. This second BACnet device contains an "operator interface" application program to which the operator's keystrokes are sent without filtering. The operator interface application program's output is conveyed through the VT services and ultimately displayed for the human operator. Normally, the BACnet device to which the operator is actually connected would recognize some local signal meaning "end virtual terminal session" but otherwise would not filter the operator's keystrokes.

17.1.1 Basic Services

There are three basic services provided: VT-Open, VT-Close, and VT-Data. The VT-Open service is used to establish a VT-session between peer processes. The VT-Close service is used to terminate a previously established session. The VT-Data service is used to exchange data between peer processes.

17.1.2 VT-classes

The classes of virtual terminal that are available in a peer VT service may be determined by examining the Device object in the peer device. The Device object has a property called VT_Classes_Supported, which may be read with the ReadProperty or ReadPropertyMultiple service to determine which VT-classes are available in that device.

17.1.3 Active VT-sessions

The active VT-sessions within a peer VT service may be determined by examining the Device object in the peer device. The Device object has a property called Active_VT_Sessions, which may be read with the ReadProperty or ReadPropertyMultiple service to determine which VT-session IDs are in use within that device.

17.1.4 State Diagram for VT-Open, VT-Data, and VT-Close

There are three phases of operation within a VT session context: IDLE, DATA EXCHANGE, and HOLD. In the IDLE phase, no VT-session exists. Once a VT-session is created through the use of the VT-Open service, the VT context enters the DATA EXCHANGE phase. The VT context remains in the DATA EXCHANGE phase until one of two events occurs:

(a) a successful VT-Close is performed, terminating the VT-session context, or
(b) a VT-Data request returns 'Result (-)'.

The HOLD phase occurs when a VT-Data request cannot be confirmed. Figure 17-4 shows the relationship between phases.

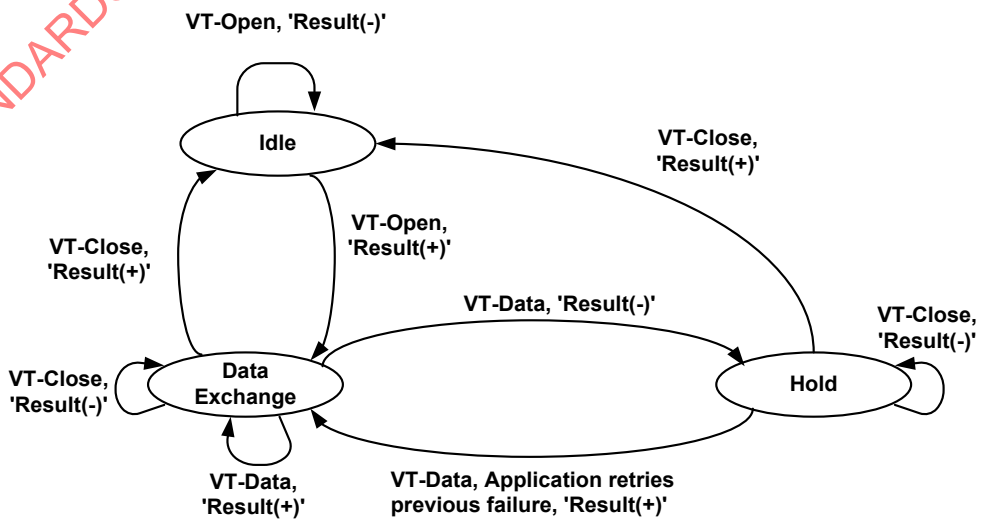


Figure 17-4. Virtual terminal services state diagram.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.1.5 VT Session Synchronization

Each of the peers participating in a VT-session shall maintain two VT-data flags that are used to synchronize the VT-session. One flag represents the device's role as a sending BACnet-user and is the sequence number, which that alternates between zero and one, of the next VT-data that is to be sent. This sequence number is incremented modulo 2 upon receipt of a 'Result(+)' response to a VT-Data request. This flag is initialized to 0 (the first VT-Data request uses a sequence number of 0) when the VT-session is established.

The other flag represents the device's role as a receiving BACnet-user and is the sequence number of the last VT-Data request that was correctly received. This sequence number is initialized to 1 (the next VT-Data indication is expected to have a sequence number of 0) when the VT session is established. The sequence number is incremented modulo 2 when a VT-Data indication with the expected sequence number is received and successfully processed.

The receiving VT-session context shall also store the last received 'All New Data Accepted' and 'Accepted Octet Count'. This is required in the event that a 'Result(+)' response to a VT-Data indication is lost, which would be detected by the receipt of a VT-Data indication with the same 'VT-Data Flag' as the previously received 'VT-Data Flag' saved in the VT session context.

17.1.6 VT Session Identifiers

Associated with each VT-session are two session identifiers, a 'Local VT Session Identifier' and a 'Remote VT Session Identifier'. These session identifiers provide a way to associate the data from a particular VT-Data request with the correct process. The value of the session identifiers is established as part of the VT-Open service. Each device selects its own 'Local VT Session Identifier', which shall be unique to all active VT-sessions in the device, without regard to whether the device's role in the session is a client or a server. The appropriate 'Remote VT Session Identifier' is obtained from the VT-Open service parameters.

When VT data are conveyed to a remote device, the 'Remote VT Session Identifier' is conveyed with the data. This session identifier is used by the remote device to identify the correct VT-session.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.2 VT-Open Service

The VT-Open service is used to establish a VT-session with a peer VT-user. The service request includes a VT-class type that identifies a particular set of assumptions about the character repertoire and encoding to be used with this session.

17.2.1 Structure

The structure of the VT-Open service primitives is shown in Table 17-1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 17-1. Structure of VT-Open Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
VT-class	M	M(=)		
Local VT Session Identifier	M	M(=)		
Result (+)			S	S(=)
Remote VT Session Identifier			M	M(=)
Result (-)			S	S(=)
Error Type			M	M(=)

17.2.1.1 Argument

This parameter shall convey the parameters for the VT-Open confirmed service request.

17.2.1.1.1 VT-class

This parameter, of type BACnetVTClass, shall identify the name of the desired class of session to be established. The standard enumeration is:

DEFAULT_TERMINAL
ANSI_X3.64
DEC_VT52
DEC_VT100
DEC_VT220
HP_700/94
IBM_3130

The enumeration DEFAULT_TERMINAL shall refer to a terminal with the characteristics defined in Clause 17.5. All VT-users are required to support at least one VT-class, namely DEFAULT_TERMINAL. Other VT-class types may also be supported by a given VT-user. It is possible to discover which VT-class types are supported by reading the VT_Classes_Supported property of the Device object.

17.2.1.1.2 Local VT Session Identifier

The 'Local VT Session Identifier' parameter shall be an unsigned integer in the range 0-255 indicating the unique VT-session in the requesting VT-user, which shall be used to receive VT-Data output from the opened virtual terminal. This identifier becomes the 'Remote VT Session Identifier' to the responding VT-user.

17.2.1.2 Result (+)

The 'Result (+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameter.

17.2.1.2.1 Remote VT Session Identifier

The 'Remote VT Session Identifier' parameter shall be an unsigned integer in the range 0-255 indicating a unique VT-session that exists within the responding VT-user's context. From the perspective of the responding VT-user, this parameter is the 'Local VT Session Identifier'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.2.1.3 Result (-)

The 'Result (-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

17.2.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

17.2.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to allocate the resources necessary to establish a VT-session. If there is no VT-user that can provide the desired VT-class, then the 'Result (-)' response shall be returned. If there is a VT-user that can provide the desired VT-class, then the responding BACnet-user shall attempt to establish a new VT-session. If the VT-user does not have the resources to establish another session, then the 'Result (-)' response shall be returned. If the VT-user has the resources, a new VT-session shall be created, the local VT-data Flags shall be initialized as specified in Clause 17.1.5, and a new VT-session Identifier shall be returned in the 'Result (+)' response.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.3 VT-Close Service

The VT-Close service is used to terminate a previously established VT-session with a peer VT-user. The service request may specify a particular VT-session to be terminated or a list of VT-sessions to be terminated.

17.3.1 Structure

The structure of the VT-Close service primitives is shown in Table 17-2. The terminology and symbology used in this table are explained in Clause 5.6.

Table 17-2. Structure of VT-Close Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
List of Remote VT Session Identifiers	M	M(=)		
Result (+)			S	S(=)
Result (-)			S	S(=)
Error Type			M	M(=)
List of VT Session Identifiers			C	C(=)

17.3.1.1 Argument

This parameter shall convey the parameters for the VT-Close confirmed service request.

17.3.1.1.1 List of Remote VT Session Identifiers

The 'List of Remote VT Session Identifiers' parameter shall consist of a list of one or more 'Remote VT Session Identifiers'. Each 'Remote VT Session Identifier' shall indicate the particular VT-session that is to be terminated.

17.3.1.2 Result (+)

The 'Result (+)' parameter shall indicate that the service request succeeded.

17.3.1.3 Result (-)

The 'Result (-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

17.3.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

17.3.1.3.2 List of VT Session Identifiers

If the 'Error Type' parameter returns an 'Error Code' of VT_SESSION_TERMINATION_FAILURE, then this parameter shall be included. If the 'Error Type' parameter indicates some other error, then this parameter shall be omitted. The 'List of VT Session Identifiers' parameter shall consist of a list of one or more 'VT Session Identifiers'. Each 'VT Session Identifier' shall indicate the particular VT-session that could not be terminated. The Session Identifiers returned are the ones that are local with respect to the device receiving the 'Result(-)' primitive, the requesting VT-user.

17.3.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to terminate each VT-session specified by the 'List of Remote VT Session Identifiers' parameter. From the viewpoint of the responding BACnet-user, these are 'Local VT Session Identifiers'. If one or more of the specified VT-sessions cannot be terminated for some reason, then all of the specified sessions that can be terminated shall be terminated and a 'Result (-)' response shall be returned. If all of the specified VT-sessions are successfully terminated, then the 'Result (+)' response shall be returned.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.4 VT-Data Service

The VT-Data service is used to exchange data with a peer VT-user through a previously established VT-session. The sending BACnet-user provides new input for the peer VT-user, which may accept or reject the new data. If the new data are rejected, then it is up to the sending BACnet-user to retry the request at a later time.

17.4.1 Structure

The structure of the VT-Data service primitives is shown in Table 17-3. The terminology and symbology used in this table are explained in Clause 5.6.

Table 17-3. Structure of VT-Data Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
VT-session Identifier	M	M(=)		
VT-new Data	M	M(=)		
VT-data Flag	M	M(=)		
Result (+)			S	S(=)
All new Data Accepted			M	M(=)
Accepted Octet Count			C	C(=)
Result (-)			S	S(=)
Error Type			M	M(=)

17.4.1.1 Argument

This parameter shall convey the parameters for the VT-Data confirmed service request.

17.4.1.1.1 VT-session Identifier

The 'VT-session Identifier' parameter shall indicate the particular VT-session to which data will be sent. It is the 'Remote VT Session Identifier' from the perspective of the requesting BACnet-user and the 'Local VT Session Identifier' from the perspective of the responding BACnet-user.

17.4.1.1.2 VT-new Data

The 'VT-new Data' parameter shall specify the octets of new data that are to be sent to the peer VT-user.

17.4.1.1.3 VT-data Flag

The 'VT-data Flag' parameter, of type Unsigned, shall indicate the expected sequence of VT-Data requests. It shall have values of 0 or 1, which alternate with each new VT-Data request for the same VT session.

17.4.1.2 Result (+)

The 'Result (+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters.

17.4.1.2.1 All New Data Accepted

The 'All New Data Accepted' parameter, of type BOOLEAN, shall be equal to TRUE if all of the 'VT-new Data' octets were accepted by the peer VT-user. In this case, the service shall be considered completed. If the 'All New Data Accepted' parameter is FALSE, then some of the 'VT-new Data' octets were unable to be accepted by the peer VT-user. Typically this could occur because of resource limitations in the peer VT-user. In this case, it is up to the sending BACnet user to re-issue the VT-Data request at a later time, including only those octets that could not be accepted.

17.4.1.2.2 Accepted Octet Count

The 'Accepted Octet Count' parameter shall only be present if the 'All New Data Accepted' parameter is FALSE. In this case, the 'Accepted Octet Count' parameter shall indicate the number of octets that were actually accepted from those presented in the 'VT-new Data' parameter of the service request. If the 'All New Data Accepted' parameter is TRUE, then the 'Accepted Octet Count' parameter shall be omitted.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.4.1.3 Result (-)

The 'Result (-)' parameter shall indicate that the service request has failed in its entirety. The reason for the failure shall be specified by the 'Error Type' parameter.

17.4.1.3.1 Error Type

This parameter consists of two component parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18.

17.4.2 Service Procedure

The sending BACnet user shall send the initial VT-Data request using a 'VT-data Flag' value of 0. Subsequent VT-Data requests for the same session, except retries, shall alternate sequence numbers 0 and 1. New VT-Data requests with the alternate sequence number shall not be transmitted until a 'Result(+)' has been received for the previous VT-Data request.

After verifying the validity of the request, the receiving BACnet-user shall attempt to locate the session specified by the 'VT-session Identifier' parameter. If the VT-session cannot be located, then the 'Result (-)' response shall be returned.

If the specified VT-session is found and the received 'VT-Data Flag' is the same as the last received 'VT-Data Flag' for this session, then this is a duplicate VT-Data request. The data shall be discarded, and a 'Result(+)' shall be returned containing the 'All New Data Accepted' and 'Accepted Octet Count' values that have been saved in the VT-session context from the previous VT-Data response.

If the specified VT-session is found and the received 'VT-Data Flag' is different from the last received 'VT-Data Flag' for this session, then this is a new VT-Data request. If all of the data can be added to the session's input queue, then the data shall be queued and a 'Result(+)' shall be returned with 'All New Data Accepted' = TRUE and the 'Accepted Octet Count' absent. If all of the data in the VT-Data request cannot be added to the session's input queue, then as many data as possible shall be queued and the remainder discarded. A 'Result(+)' shall be returned with 'All New Data Accepted' = FALSE and 'Accepted octet Count' equal to the number of octets that were able to be queued. In either case, the returned 'All New Data Accepted' and 'Accepted Octet Count' values shall be saved in the VT-session context.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

17.5 Default Terminal Characteristics

Every VT-user shall implement at least one VT-class, DEFAULT_TERMINAL. The default terminal is based on a limited set of functions that are commonly found in all types of interactive terminal devices. The characteristics of the default terminal include a character repertoire, control functions, and page size assumptions. No other assumptions may be made about the behavior of the VT-user.

17.5.1 Default Terminal Character Repertoire

The default terminal character repertoire shall be defined as a particular mapping between single octet values and their associated meanings. The default terminal character repertoire shall include three types of meanings for a given octet value:

(a) a particular symbol (SYMBOL), e.g., "A";
(b) a particular implied control function (CONTROL), e.g., Carriage Return;
(c) a null meaning (NUL), e.g., Unused, shall be ignored.

Those octets specified as NUL within the default terminal character repertoire have no assumed meanings and shall be ignored if they are received by either VT-user. The default terminal character repertoire is a subset of ASCII. Table 17-4 summarizes the octet encodings for each character in the default terminal character repertoire. The "Octet Value" field indicates octet encodings as decimal (base 10) values from 0 to 255. A range of octet values is indicated by two decimal numbers, e.g., 000-006.

Table 17-4. Default Terminal Character Repertoire

Octet Value	Type	Meaning
000 006	NUL	none
007	CONTROL	audible indication (BEL)
008	CONTROL	non-destructive backspace (BS)
009	CONTROL	horizontal tab (TAB)
010	CONTROL	line feed (LF)
011-012	NUL	none
013	CONTROL	carriage return (CR)
014-031	NUL	none
032	SYMBOL	space
033	SYMBOL	! exclamation
034	SYMBOL	double quote
035	SYMBOL	# pound sign
036	SYMBOL	$ dollar sign
037	SYMBOL	% percent
038	SYMBOL	& ampersand
039	SYMBOL	' apostrophe
040	SYMBOL	(left parenthesis
041	SYMBOL	) right parenthesis
042	SYMBOL	* asterisk
043	SYMBOL	+ plus sign
044	SYMBOL	, comma
045	SYMBOL	- minus sign
046	SYMBOL	. period
047	SYMBOL	/ forward slash
048	SYMBOL	0 zero
049	SYMBOL	1 one

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Octet Value	Type	Meaning
050	SYMBOL	2 two
051	SYMBOL	3 three
052	SYMBOL	4 four
053	SYMBOL	5 five
054	SYMBOL	6 six
055	SYMBOL	7 seven
056	SYMBOL	8 eight
057	SYMBOL	9 nine
058	SYMBOL	: colon
059	SYMBOL	; semicolon
060	SYMBOL	< left angle bracket (or less than)
061	SYMBOL	= equal sign
062	SYMBOL	> right angle bracket (or greater than)
063	SYMBOL	? question mark
064	SYMBOL	@ commercial at sign
065	SYMBOL	A
066	SYMBOL	B
067	SYMBOL	C
068	SYMBOL	D
069	SYMBOL	E
070	SYMBOL	F
071	SYMBOL	G
072	SYMBOL	H
073	SYMBOL	I
074	SYMBOL	J
075	SYMBOL	K
076	SYMBOL	L
077	SYMBOL	M
078	SYMBOL	N
079	SYMBOL	O
080	SYMBOL	P
081	SYMBOL	Q
082	SYMBOL	R
083	SYMBOL	S
084	SYMBOL	T
085	SYMBOL	U
086	SYMBOL	V
087	SYMBOL	W
088	SYMBOL	X
089	SYMBOL	Y
090	SYMBOL	Z
091	SYMBOL	[left square bracket
092	SYMBOL	\ back slash
093	SYMBOL	] right square bracket
094	SYMBOL	^ caret
095	SYMBOL	_ underscore

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Octet Value	Type	Meaning
096	SYMBOL	` accent grave
097	SYMBOL	a
098	SYMBOL	b
099	SYMBOL	c
100	SYMBOL	d
101	SYMBOL	e
102	SYMBOL	f
103	SYMBOL	g
104	SYMBOL	h
105	SYMBOL	i
106	SYMBOL	j
107	SYMBOL	k
108	SYMBOL	l
109	SYMBOL	m
110	SYMBOL	n
111	SYMBOL	o
112	SYMBOL	p
113	SYMBOL	q
114	SYMBOL	r
115	SYMBOL	s
116	SYMBOL	t
117	SYMBOL	u
118	SYMBOL	v
119	SYMBOL	w
120	SYMBOL	x
121	SYMBOL	y
122	SYMBOL	z
123	SYMBOL	{ left curly bracket
124	SYMBOL	| vertical bar
125	SYMBOL	} right curly bracket
126	SYMBOL	~ tilde
127	CONTROL	non-destructive backspace (DEL)
128-255	NUL	none

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

17.5.2 Control Functions

There are six octet codes within the default terminal character repertoire that perform control functions. In this context, control function means some action that is implied by the receipt of one of these control codes.

17.5.2.1 Octet Code 007

The octet code 007 shall represent an audible indication (BEL). Normally this would be used to sound a tone or bell signal.

17.5.2.2 Octet Codes 008 and 127

The octet code 008 shall represent a non-destructive backspace (BS). This shall cause the cursor of the virtual "output device" to be moved one character position to the left. If the cursor is at the extreme left or beginning of a line, then BS shall have no effect. This function shall only change the current position, without overwriting or altering any characters that have been previously displayed on the same "line." The octet code 127 shall be considered to be equivalent to 008 and shall have the same effects.

© ISO 2022 – All rights reserved

17.5.2.3 Octet Code 013

The octet code 013 shall represent a carriage return (CR). This shall cause the cursor to be reset to the beginning of the current "line." Subsequently received characters would then overwrite existing characters on the current "line."

17.5.2.4 Octet Code 010

The octet code 010 shall represent a line feed (LF). This shall cause the cursor to be advanced to the next line but shall not change cursor position within the line. Typically this code is used in conjunction with CR.

17.5.2.5 Octet Code 009

The octet code 009 shall represent a horizontal advance to the next tab stop (TAB). This shall cause the cursor to be advanced to the next tab position on the current line. This function shall only change the cursor position, without overwriting or altering any characters that have previously been displayed on the same line. Tab stops shall exist at every eight character positions, as shown in Figure 17-5.

17.5.3 Page Size Assumptions

There shall be only two assumptions about page size within the Default-terminal context. First, pages are assumed to have 80 columns.

```
        1         2         3         4         5
12345678901234567890123456789012345678901234567890 ...
        T         T         T         T         T         T ...
```

Figure 17-5. VT tab positions.

Each SYMBOL character received is assumed to occupy one column. NUL characters have no effect on column position. CONTROL characters have different effects, as described in Clause 17.5.2. Second, each page is assumed to be unconstrained in length. This is equivalent to a printer with continuous form paper.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

18 ERROR, REJECT, and ABORT CODES

All errors associated with the BACnet protocol are enumerated according to a category called "Error Class." Within each Error Class, the errors are further enumerated individually by "Error Code." It is thus possible for an application to take remedial action based upon two levels of granularity.

18.1 Error Class - DEVICE

This Error Class pertains to circumstances that affect the functioning of an entire BACnet device. The presence of one of these errors generally indicates that the entire service request has failed.

CONFIGURATION_IN_PROGRESS - A service request has been temporarily declined because the addressed BACnet device is in the process of being configured, either by means local to the device or by means of other protocol services.

DEVICE_BUSY - A service request has been temporarily declined because the addressed BACnet device expects to be involved in higher priority processing for a time in excess of the usual request/confirm timeout period.

INCONSISTENT_CONFIGURATION - This error code is used when a device is misconfigured and hence cannot process a request.

INTERNAL_ERROR - There are cases where some internal error is encountered. These are cases that are never expected to occur, but if they do the manufacturer should be contacted.

NOT_CONFIGURED - A device may require configuration, possibly vendor-specific, before it becomes functional. If it is not configured, it can receive BACnet requests but cannot reasonably process them.

OPERATIONAL_PROBLEM - A service request has been declined because the addressed BACnet device has detected an operational problem that prevents it from carrying out the requested service.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.2 Error Class - OBJECT

This Error Class pertains to problems related to identifying, accessing, and manipulating BACnet objects, whether BACnet-defined or not. Since these errors generally apply to individual object characteristics, they do not necessarily signal that an entire service request has failed.

BUSY - A service request has been temporarily declined because the addressed object is involved in a process that precludes execution of the service.

DYNAMIC_CREATION_NOT_SUPPORTED - An attempt has been made to create an object using an object type that cannot be created dynamically.

FILE_FULL - This applies to the case when a File Object becomes filled to a designed limit, as opposed to a No Space Available / Out of Memory situation.

INVALID_OPERATION_IN_THIS_STATE - The operation specified in a service parameter is invalid in the current state of the object.

LOG_BUFFER_FULL - The attempted operation would result in the addition of a log record to an object whose log buffer is full.

NO_ALARM_CONFIGURED - The BACnet object referenced by the service does not support, or is not configured for, event generation.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NO_OBJECTS_OF_SPECIFIED_TYPE - A search of the addressed BACnet device's object database has failed to find any objects of the object type specified in the service request.

OBJECT_DELETION_NOT_PERMITTED - An attempt has been made to delete an object that cannot be deleted or is currently protected from deletion.

OBJECT_IDENTIFIER_ALREADY_EXISTS - An attempt has been made to create a new object using an object identifier already in use.

OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED - The requested action cannot be executed because the specified object does not support the optional functionality required.

READ_ACCESS_DENIED - An attempt has been made to read the properties of an object defined as inaccessible through the BACnet protocol read services.

UNKNOWN_OBJECT - An Object_Identifier has been specified for an object that does not exist in the object database of the addressed BACnet device.

UNSUPPORTED_OBJECT_TYPE - An object type has been specified in a service parameter that is unknown or unsupported in the addressed BACnet device.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.3 Error Class - PROPERTY

This Error Class pertains to problems related to identifying, accessing, and manipulating the properties of BACnet objects, whether BACnet-defined or not. Since these errors generally apply to individual property characteristics, they do not necessarily signal that an entire service request has failed.

CHARACTER_SET_NOT_SUPPORTED - A character string value was encountered that is not a supported character set.

DATATYPE_NOT_SUPPORTED - The data is of, or contains, a datatype not supported by this instance of this property.

DUPLICATE_ENTRY - An attempt has been made to write to a SEQUENCE OF, with a value that has duplicate entries in a property that does not accept duplicates, such as duplicate times within a single SEQUENCE OF TimeValue within a BACnetSpecialEvent or in a BACnetDailySchedule array element.

DUPLICATE_NAME - An attempt has been made to write to an Object_Name property with a value that is already in use in a different Object_Name property within the device.

DUPLICATE_OBJECT_ID - An attempt has been made to write to an Object_Identifier property with a value that is already in use in a different Object_Identifier within the same device.

INCONSISTENT_SELECTION_CRITERION - A property has been referenced with a datatype inconsistent with the 'Comparison Value' specified in an 'Object Selection Criteria' service parameter. This error would arise, for example, if an analog property were compared against a Boolean constant, or vice-versa.

INVALID_ARRAY_INDEX - An attempt was made to access an array property using an array index that is outside the range permitted for this array.

INVALID_DATA_ENCODING - The encoding of a property value specified in a service parameter is not valid for the datatype of the property.

INVALID_DATATYPE - The datatype of a property value specified in a service parameter does not match the datatype of the property referenced by the specified Property_Identifier.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

INVALID_VALUE_IN_THIS_STATE - The value specified in a service parameter is invalid in the current state of the property.

LIST_ITEM_NOT_NUMBERED - List items were selected by sequence number, but the list items do not support a sequence number.

LIST_ITEM_NOT_TIMESTAMP - List items were selected by time, but the list items do not support a timestamp.

LOGGED_VALUE_PURGED - A previously logged value was purged due to a change to the list of logged properties.

NO_PROPERTY_SPECIFIED - No data was logged due to a device or object instance equal to 4194303 in the list of logged properties.

NOT_CONFIGURED_FOR_TRIGGERED_LOGGING - The attempted logging operation is only allowed when the Logging_Type property has the value TRIGGERED.

NOT_COV_PROPERTY - The property is not conveyed by COV notification.

OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED - An attempt has been made to write a value to a property that would require the device to exhibit non-supported optional functionality.

PROPERTY_IS_NOT_AN_ARRAY - An attempt has been made to access a property as an array and that property does not have an array datatype.

READ_ACCESS_DENIED - An attempt has been made to read a property defined as inaccessible through the BACnet protocol read services.

UNKNOWN_PROPERTY - A Property_Identifier has been specified in a service parameter that is unknown or unsupported in the addressed BACnet device for objects of the referenced object type.

UNKNOWN_FILE_SIZE - This error code is returned when the File_Size property is read and the size of the file is unknown.

VALUE_NOT_INITIALIZED - An attempt was made to read a property whose value has not been initialized.

VALUE_OUT_OF_RANGE - An attempt has been made to write to a property with a value that is outside the range of values defined for the property.

VALUE_TOO_LONG - A property value is too long to send in the current message context and an Abort is not an option, such as when sending an UnconfirmedCOVNotification.

WRITE_ACCESS_DENIED - An attempt has been made to write to a property defined as inaccessible through the BACnet protocol write services.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.4 Error Class - RESOURCES

This Error Class pertains to problems related to the resources of a BACnet device that affect its capacity to carry out protocol service requests.

NO_SPACE_FOR_OBJECT - An attempt to create an object has failed because not enough dynamic memory space exists in the addressed BACnet device.

NO_SPACE_TO_ADD_LIST_ELEMENT - An attempt to add an element to a list has failed because not enough dynamic memory space exists in the addressed BACnet device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NO_SPACE_TO_WRITE_PROPERTY - An attempt to write a property has failed because not enough dynamic memory space exists in the addressed BACnet device.

OUT_OF_MEMORY - There are many internal operations during the processing of typical messages that may rely on acquiring dynamically allocated space. This indicates the failure of such an allocation.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.5 Error Class - SECURITY

This Error Class pertains to problems related to security services. Without exception, these errors signal the inability of the responding BACnet-user to carry out the desired service in its entirety and are thus "fatal".

ACCESS_DENIED - This error is used when access is denied but READ_ACCESS_DENIED and WRITE_ACCESS_DENIED are not appropriate.

BAD_DESTINATION_ADDRESS - The destination address in the message is not valid. (Reserved for future use)

BAD_DESTINATION_DEVICE_ID - The Destination Device Instance in the security parameter does not match the local device instance. (Reserved for future use)

BAD_SIGNATURE - The signature in a secure message was incorrect. (Reserved for future use)

BAD_SOURCE_ADDRESS - The source address in a secure message was incorrect or missing. (Reserved for future use)

DUPLICATE_MESSAGE - The message has been rejected as a duplicate of a previously received message. (Reserved for future use)

ENCRYPTION_NOT_CONFIGURED - The device is not configured to accept encrypted messages. (Reserved for future use)

ENCRYPTION_REQUIRED - The device requires encryption for the requested operation. (Reserved for future use)

MALFORMED_MESSAGE - The message size is invalid, or security parameters are missing or malformed.

PASSWORD_FAILURE - The 'Operator Name' and 'Operator Password' did not associate correctly.

READ_ACCESS_DENIED - The requesting device did not provide security credentials of sufficient authorization to allow the request.

SECURITY_NOT_CONFIGURED - The device is not configured for the provided security on the receiving port. (Reserved for future use)

SOURCE_SECURITY_REQUIRED - The operation requested requires that the source secure or encrypt the request. (Reserved for future use)

SUCCESS - The operation was successful.

UNKNOWN_AUTHENTICATION_TYPE - The authentication method in a secure message is unknown to the receiving device. (Reserved for future use)

WRITE_ACCESS_DENIED - The requesting device did not provide security credentials of sufficient authorization to allow the request.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.6 Error Class - SERVICES

This Error Class pertains to problems related to the execution of protocol service requests, whether BACnet-defined or not.

CHARACTER_SET_NOT_SUPPORTED - A character string value was encountered that is not a supported character set.

COMMUNICATION_DISABLED - Communication has been disabled due to receipt of a DeviceCommunicationControl request. This error is not expected in response to service requests but rather is expected to be used internally when actions that would normally result in initiation of a BACnet service fail due to communications being disabled. As such, this error code would be seen in objects that record the result of failed operations, such as logging objects.

COV_SUBSCRIPTION_FAILED - COV Subscription failed for some reason.

FILE_ACCESS_DENIED - Generated in response to an AtomicReadFile or AtomicWriteFile service request for access to a file that is currently locked or otherwise not accessible.

INCONSISTENT_OBJECT_TYPE - A device receives a service request for an object whose type is inconsistent with the service requested, or for an object that doesn't support the service. An example is an AtomicReadFile request received for an object that is not a File object.

INCONSISTENT_PARAMETERS - A conflict exists because two or more of the parameters specified in the service request are mutually exclusive.

INVALID_CONFIGURATION_DATA - The configuration data provided was invalid or corrupt.

INVALID_DATA_ENCODING - The encoding of a value specified in a service parameter is not valid for the datatype of the parameter.

INVALID_EVENT_STATE - The 'Event State Acknowledged' parameter conveyed by an AcknowledgeAlarm service request does not match the 'To State' parameter of the most recent occurrence of the same transition type of the event being acknowledged.

INVALID_FILE_ACCESS_METHOD - Generated in response to an AtomicReadFile or AtomicWriteFile request that specifies a 'File Access Method' that is not valid for the specified file.

INVALID_FILE_START_POSITION - Generated in response to an AtomicReadFile or AtomicWriteFile request that specifies an invalid 'File Start Position' or invalid 'File Start Record' parameter.

INVALID_PARAMETER_DATATYPE - The datatype of a value specified for a service parameter is not appropriate to the parameter.

INVALID TAG - This error indicates that a syntax error was encountered in the request.

INVALID_TIMESTAMP - The 'Time Stamp' parameter conveyed by an AcknowledgeAlarm service request does not match the time of the most recent occurrence of the event being acknowledged.

LIST_ELEMENT_NOT_FOUND - A list data item required for carrying out the service request was not found.

MISSING_REQUIRED_PARAMETER - A parameter required for the execution of a service request has not been supplied.

OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED - The parameters of a service are such that the device would be required to exhibit non-supported optional functionality.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

PARAMETER_OUT_OF_RANGE - Generated in response to a confirmed request APDU that conveys a parameter whose value is outside the range defined for this service.

PROPERTY_IS_NOT_A_LIST - An attempt has been made to access a property via either the AddListElement service or the RemoveListElement service and that property does not have a list datatype.

PROPERTY_IS_NOT_AN_ARRAY - An attempt has been made to access a property as an array and that property does not have an array datatype.

SERVICE_REQUEST_DENIED - A request has been made to execute a service for which the requesting BACnet device does not have the appropriate authorization.

UNKNOWN_SUBSCRIPTION - No subscription can be found that matches the specified object, property, and process identifier for the received notification.

VALUE_OUT_OF_RANGE - The requested action cannot be executed because one of the parameters provided is outside of the range supported by the device.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class.

18.7 Error Class - COMMUNICATION

This Error Class pertains to problems related to network communications. These codes indicate problems reported by a remote device in abort and reject PDUs, or they indicate problems detected internally. These error codes are stored in properties of objects whose operation involves the network communications, such as the Trend Log object's Log_Buffer property. This Error Class shall not be conveyed in error PDUs.

ABORT_APDU_TOO_LONG - An APDU was received from the local application program whose overall size exceeds the maximum transmittable length or exceeds the maximum number of segments accepted by the server.

ABORT_APPLICATION_EXCEEDED_REPLY_TIME - A device receives a confirmed request but its application layer has not responded within the published APDU Timeout period.

ABORT_BUFFER_OVERFLOW - An input buffer capacity has been exceeded in this device or was reported by the remote device.

ABORT_INSUFFICIENT_SECURITY - The transaction is aborted due to receipt of a PDU secured differently than the original PDU of the transaction.

ABORT_INVALID_APDU_IN_THIS_STATE - An APDU was received, by this device or the remote device, that was not expected in the present state of the Transaction State Machine.

ABORT_OUT_OF_RESOURCES - A device receives a request but cannot start processing because it has run out of some internal resource.

ABORT_PREEMPTED_BY_HIGHER_PRIORITY_TASK - The transaction was aborted to permit higher priority processing by this device or the remote device.

ABORT_SECURITY_ERROR - The Transaction is aborted due to receipt of a security error.

ABORT_SEGMENTATION_NOT_SUPPORTED - An Abort PDU specifying an abort code of SEGMENTATION_NOT_SUPPORTED was sent or received by this device.

ABORT_TSM_TIMEOUT - A transaction state machine timer exceeded the timeout applicable for the current state, causing the transaction machine to abort the transaction.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ABORT_PROPRIETARY - An abort PDU with a proprietary reason was sent or received by this device.

ABORT_WINDOW_SIZE_OUT_OF_RANGE - A device receives a request that is segmented, or receives any segment of a segmented request, where the Proposed Window Size field of the PDU header is either zero or greater than 127.

ABORT_OTHER - This device sent or received an abort PDU with a reason of OTHER.

ADDRESSING_ERROR - A network request failed due to an addressing error.

BVLC_FUNCTION_UNKNOWN - The indicated BVLC function is unknown.

BVLC_PROPRIETARY_FUNCTION_UNKNOWN - The indicated BVLC proprietary function is unknown.

DELETE_FDT_ENTRY_FAILED - A Delete-Foreign-Device-Table-Entry request failed.

DISTRIBUTE_BROADCAST_FAILED - A broadcast network request failed due to a failure of a Distribute-Broadcast-To-Network.

DNS_ERROR - A DNS error occurred.

DNS_NAME_RESOLUTION_FAILED - The DNS host name resolution failed

DNS_RESOLVER_FAILURE - The DNS resolver failed.

DNS_UNAVAILABLE - The DNS name resolution service is unavailable

HEADER_ENCODING_ERROR - A header of the message has encoding errors.

HEADER_NOT_UNDERSTOOD - A message header that must be understood is not supported.

HTTP_ERROR - An error occurred in HTTP.

HTTP_NOT_A_SERVER - No inbound requests are supported. The host is not an HTTP server.

HTTP_NO_UPGRADE - Server does not accept upgrade to the WebSocket protocol.

HTTP_PROXY_AUTHENTICATION_FAILED - Proxy Authentication failed

HTTP_RESOURCE_NOT_LOCAL - Redirect to another location of the peer WebSocket port received.

HTTP_RESPONSE_MISSING_HEADER - Missing header fields in response.

HTTP_RESPONSE_TIMEOUT - No response from server within timeout.

HTTP_RESPONSE_SYNTAX_ERROR - Syntax error in HTTP response received.

HTTP_RESPONSE_VALUE_ERROR - Errors in values of the HTTP response received.

HTTP_TEMPORARY_UNAVAILABLE - No more HTTP connections are available currently.

HTTP_UNEXPECTED_RESPONSE_CODE - Server reports unexpected response code.

HTTP_UPGRADE_REQUIRED - No upgrade request was received by the server.

HTTP_UPGRADE_ERROR - Upgrading to the WebSocket protocol failed.

HTTP_WEBSOCKET_HEADER_ERROR - Response contains any other error in HTTP header fields.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

IP_ADDRESS_NOT_REACHABLE - The IP address of the peer is not reachable.

IP_ERROR - An error occurred on the IP protocol level.

INVALID_TAG - This error indicates that an improper tag was found when parsing the response to a confirmed service request or an unconfirmed service request.

MESSAGE_INCOMPLETE - A message was presented that is incomplete.

MESSAGE_TOO_LONG - A network request failed due a message that was too long to make it to its destination.

NETWORK_DOWN - This error indicates that the local network connection was not established when the request was initiated.

NODE_DUPLICATE_VMAC - A duplicate VMAC has been detected.

NOT_A_BACNET_SC_HUB - The node received a message that would require it to be a hub, but the node is not a hub.

NOT_ROUTER_TO_DNET - A Reject-Message-To-Network with reason 1 was returned in response to a network request.

PAYLOAD_EXPECTED - A payload was expected to be present in the message.

READ_BDT_FAILED - A Read-Broadcast-Distribution-Table request failed.

READ_FDT_FAILED - A Read-Foreign-Device-Table request failed.

REGISTER_FOREIGN_DEVICE_FAILED - A Register-Foreign-Device request failed.

REJECT_BUFFER_OVERFLOW - An input buffer capacity has been exceeded in this device or has been reported by the remote device.

REJECT_INCONSISTENT_PARAMETERS - The remote device sent a reject PDU with a reason of INCONSISTENT_PARAMETERS.

REJECT_INVALID_PARAMETER_DATA_TYPE - The remote device sent a reject PDU with a reason of INVALID_PARAMETER_DATATYPE.

REJECT_INVALID_TAG - This device or the remote device encountered an invalid tag while parsing a message.

REJECT_MISSING_REQUIRED_PARAMETER - The remote device sent a reject PDU with a reason of MISSING_REQUIRED_PARAMETER.

REJECT_PARAMETER_OUT_OF_RANGE - The remote device sent a reject PDU with a reason of PARAMETER_OUT_OF_RANGE.

REJECT_TOO_MANY_ARGUMENTS - The remote device sent a reject PDU with a reason of TOO_MANY_ARGUMENTS.

REJECT_UNDEFINED_ENUMERATION - The remote device sent a reject PDU with a reason of UNDEFINED_ENUMERATION.

REJECT_UNRECOGNIZED_SERVICE - The remote device sent a reject PDU with a reason of UNRECOGNIZED_SERVICE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

REJECT_PROPRIETARY - This reject reason indicates that a proprietary reject reason was sent or received by this device.

REJECT_OTHER - The remote device sent a reject PDU with a reason of OTHER.

ROUTER_BUSY - A network request failed due to a router en-route being busy.

SECURITY_ERROR - A network request failed due a security error en-route.

TCP_CONNECT_TIMEOUT - The TCP connection could not be established due to no response within timeout.

TCP_CONNECTION_REFUSED - The TCP connection is not accepted by the peer.

TCP_CLOSED_BY_LOCAL - The TCP connection was closed by the local endpoint.

TCP_CLOSED_OTHER - The TCP connection was closed due to some unspecified reason.

TCP_ERROR - An error occurred in TCP.

TIMEOUT - This error indicates that a request timed out before a response was received from the remote device.

TLS_CLIENT_AUTHENTICATION_FAILED - Authentication of the client failed.

TLS_CLIENT_CERTIFICATE_ERROR - The client certificate contains an error that prevents it from being authenticated.

TLS_CLIENT_CERTIFICATE_EXPIRED - Client certificate validity window does not include current time.

TLS_CLIENT_CERTIFICATE_REVOKED - Client certificate is revoked.

TLS_ERROR - An error occurred in TLS.

TLS_SERVER_AUTHENTICATION_FAILED - Authentication of the server failed.

TLS_SERVER_CERTIFICATE_ERROR - The server certificate contains an error that prevents it from being authenticated.

TLS_SERVER_CERTIFICATE_EXPIRED - Server certificate validity window does not include current **time.**

TLS_SERVER_CERTIFICATE_REVOKED - Server certificate is revoked.

UNEXPECTED_DATA - Data was found that was not expected.

UNKNOWN_DEVICE - This error indicates that a request was not initiated because the remote device could not be found.

UNKNOWN_ROUTE - This error indicates that a request was not initiated because a route to the network where the remote device resides could not be found.

UNKNOWN_NETWORK_MESSAGE - A network request failed that relied on a network message unknown to the receiver.

WEBSOCKET_CLOSE_ERROR - An error occurred in closing the WebSocket connection.

WEBSOCKET_CLOSED_BY_PEER - The WebSocket connection was closed by the peer.

WEBSOCKET_CLOSED_ABNORMALLY - The WebSocket connection was closed abnormally.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

WEBSOCKET_DATA_AGAINST_POLICY - An endpoint has closed the WebSocket connection due to data received that is violating its policy.

WEBSOCKET_DATA_INCONSISTENT - An endpoint has closed the WebSocket connection due to data received that is inconsistent with the type of the message.

WEBSOCKET_DATA_NOT_ACCEPTED - An endpoint has closed the WebSocket connection due to data of a type not accepted.

WEBSOCKET_ENDPOINT_LEAVES - An endpoint is "going away", such as a server going down or a client having left the WebSocket connection.

WEBSOCKET_ERROR - A WebSocket error occurred, and the WebSocket connection is closed.

WEBSOCKET_EXTENSION_MISSING - The initiating peer failed to establish the WebSocket connection due to extensions not confirmed by the answering peer.

WEBSOCKET_FRAME_TOO_LONG - An endpoint has closed the WebSocket connection due to data received that is too long to be processed.

WEBSOCKET_PROTOCOL_ERROR - An endpoint has closed the WebSocket connection due to a protocol error.

WEBSOCKET_SCHEME_NOT_SUPPORTED - The WebSocket URI presented to the WebSocket port indicates a scheme whose respective protocol variant is not supported by the WebSocket port.

WEBSOCKET_UNKNOWN_CONTROL_MESSAGE - A WebSocket control message was received that was not understood.

WEBSOCKET_REQUEST_UNAVAILABLE - The answering peer has closed the WebSocket connection due to a condition that prevented it from executing the request received.

WRITE_BDT_FAILED - A Write-Broadcast-Distribution-Table request failed.

OTHER - This error indicates that a communication error occurred other than those previously enumerated for this Error Class.

18.8 Error Class - VT

This Error Class pertains to problems related to the execution of Virtual Terminal services.

NO_VT_SESSIONS_AVAILABLE - This error indicates that the target device could not fulfill a VT-Open request because of resource limitations.

UNKNOWN_VT_CLASS - This error indicates that the 'VT-Class' specified in a VT-Open request was not recognized by the target device.

UNKNOWN_VT_SESSION - This error indicates that the 'VT-Session ID' specified in a VT-Data or VT-Close request was not recognized by the target device.

VT_SESSION_ALREADY_CLOSED - This error indicates that an attempt has been made to close a VT-session that has been previously terminated.

VT_SESSION_TERMINATION_FAILURE - This error indicates that one of the 'VT-Sessions' specified in a VT-Close request could not be released for some implementation-dependent reason.

OTHER - This error code is returned for a reason other than any of those previously enumerated for this Error Class

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

18.9 Reject Reason

Only confirmed request PDUs can be rejected. The possible reasons for rejecting the PDU are enumerated in this clause.

BUFFER_OVERFLOW - A buffer capacity has been exceeded.

INCONSISTENT_PARAMETERS - Generated in response to a confirmed request APDU that omits a conditional service argument that should be present or contains a conditional service argument that should not be present. This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.

INVALID_DATA_ENCODING - Generated in response to a confirmed request APDU that conveys a parameter whose encoding is not valid for the datatype of the parameter.

INVALID_PARAMETER_DATA_TYPE - Generated in response to a confirmed request APDU in which the encoding of one or more of the service parameters does not follow the correct type specification. This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.

INVALID_TAG - While parsing a message, an invalid tag was encountered. Since an invalid tag could confuse the parsing logic, any of the following Reject Reasons may also be generated in response to a confirmed request containing an invalid tag: INCONSISTENT_PARAMETERS, INVALID_PARAMETER_DATA_TYPE, MISSING_REQUIRED_PARAMETER, and TOO_MANY_ARGUMENTS.

MISSING_REQUIRED_PARAMETER - Generated in response to a confirmed request APDU that is missing at least one mandatory service argument. This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.

PARAMETER_OUT_OF_RANGE - Generated in response to a confirmed request APDU that conveys a parameter whose value is outside the range defined for this service.

TOO_MANY_ARGUMENTS - Generated in response to a confirmed request APDU in which the total number of service arguments is greater than specified for the service. This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.

UNDEFINED_ENUMERATION - Generated in response to a confirmed request APDU in which one or more of the service parameters is decoded as an enumeration that is not defined by the type specification of this parameter.

UNRECOGNIZED_SERVICE - Generated in response to a confirmed request APDU in which the Service Choice field specifies an unknown or unsupported service.

OTHER - Generated in response to a confirmed request APDU that contains a syntax error for which an error code has not been explicitly defined.

18.10 Abort Reason

APDU_TOO_LONG - An APDU was received from the local application program whose overall size exceeds the maximum transmittable length or exceeds the maximum number of segments accepted by the server.

APPLICATION_EXCEEDED_REPLY_TIME - A device receives a confirmed request but its application layer has not responded within the published APDU Timeout period.

BUFFER_OVERFLOW - A buffer capacity has been exceeded.

INSUFFICIENT_SECURITY - The transaction is aborted due to receipt of a PDU secured differently than the original PDU of the transaction.

INVALID_APDU_IN_THIS_STATE - Generated in response to an APDU that is not expected in the present state of the Transaction State Machine.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

OUT_OF_RESOURCES - A device receives a request but cannot start processing because it has run out of some internal resource.

PREEMPTED_BY_HIGHER_PRIORITY_TASK - The transaction shall be aborted to permit higher priority processing.

SECURITY_ERROR - The Transaction is aborted due to receipt of a security error.

SEGMENTATION_NOT_SUPPORTED - Generated in response to an APDU that has its segmentation bit set to TRUE when the receiving device does not support segmentation. It is also generated when a BACnet-ComplexACK-PDU is large enough to require segmentation but it cannot be transmitted because either the transmitting device or the receiving device does not support segmentation.

TSM_TIMEOUT - A transaction state machine timer exceeded the timeout applicable for the current state, causing the transaction machine to abort the transaction.

WINDOW_SIZE_OUT_OF_RANGE - A device receives a request that is segmented, or receives any segment of a segmented request, where the Proposed Window Size field of the PDU header is either zero or greater than 127.

OTHER - This abort reason is returned for a reason other than any of those previously enumerated.

18.11 Confirmed Service Common Errors

Some errors are generic and can occur when any confirmed service is requested. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
During the processing of the request, dynamically allocated memory was not available at an intermediate step so the request cannot be completed.	RESOURCES	OUT_OF_MEMORY
An unexpected internal error occurred that cannot be recovered from.	DEVICE	INTERNAL_ERROR
The device is not completely configured and therefore can't fulfill the request.	DEVICE	NOT_CONFIGURED
Some misconfiguration is preventing the request from being fulfilled.	DEVICE	INCONSISTENT_CONFIGURATION

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19 BACnet PROCEDURES

This clause defines several procedures that are commonly required in building automation and control systems. Each procedure makes use of BACnet capabilities defined elsewhere in this standard.

19.1 Backup and Restore

This clause describes the procedures to be used to backup and restore the configuration of BACnet devices.

19.1.1 The Backup and Restore Procedures

In BACnet building control systems, many devices will have configuration data that is set up by a vendor's proprietary configuration tool. This setup may consist of network visible BACnet objects and/or non-network visible settings. This section outlines the standard method that BACnet devices will employ if an interoperable device backup and restore feature is to be provided.

The backup and restore procedures use File objects to hold and transfer the configuration data. The content and format of the configuration files is a local matter. The choice of whether to use stream-based files or record-based files is a local matter. The services required to support the backup and restore procedures are ReinitializeDevice, ReadProperty, WriteProperty, AtomicWriteFile, AtomicReadFile, and optionally CreateObject, ReadPropertyMultiple, or WritePropertyMultiple.

19.1.2 Backup

For the purposes of this discussion, the device performing the backup procedure will be referred to as device A, and the device being backed up will be device B.

19.1.2.1 Initiation of the Backup Procedure

Device A sends a ReinitializeDevice(STARTBACKUP, <password>) message to device B. Device A will await a response from device B before continuing with the backup procedure.

19.1.2.2 Preparation for Backup

Before starting a backup procedure, device A shall read the Backup_Preparation_Time property, if present, from device B's Device object. If the property is not present in device B, the value shall be assumed to be 0.

Upon receipt of the ReinitializeDevice(STARTBACKUP, <password>)message, if device B is able to perform a backup procedure, device B shall respond with a 'Result(+)' to the ReinitializeDevice service request. Device B shall set its Backup_And_Restore_State to PREPARING_FOR_BACKUP. Upon receipt of a Result(+), device A shall monitor the Backup_And_Restore_State property and not continue with the backup until the property contains the value PERFORMING_A_BACKUP. During the time period immediately following the Result(+) defined by the Backup_Preparation_Time, device B is allowed to ignore requests from device A and as such device A shall not consider a lack of response during this period to be an error condition. It is a local matter whether device A initiates the monitoring of the Backup_And_Restore_State property during or after this time period. Once device B changes its Backup_And_Restore_State to PERFORMING_A_BACKUP, it shall not ignore requests from device A regardless of whether the Backup_Preparation_Time time period has expired.

If device B is unable to perform a backup procedure or is already performing a backup procedure, then it will respond to the ReinitializeDevice service request with a 'Result(-)' response. Assuming device B supports the backup procedure and the request was properly formulated, the valid Error Class:Error Codes that can be returned are :

DEVICE:CONFIGURATION_IN_PROGRESS - if device B is already processing a backup or a restore request.

SECURITY:PASSWORD_FAILURE - if the password that was provided was incorrect or if a password is required and one was not provided.

After device B responds to the ReinitializeDevice request with a 'Result(+)', device B has Backup_Preparation_Time seconds to prepare for the backup procedure. During this period of time, device B is not required to respond to any BACnet service requests. Once this period of time elapses, device B is required to respond to read requests for properties of the Device object. When device B has successfully completed its backup preparations in their entirety, the

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

configuration File objects shall exist in the device and the Backup_And_Restore_State property shall be set to PERFORMING_A_BACKUP. The creation of configuration File objects during this time shall not have an effect on the Database_Revision property.

If device B is unable to successfully complete its backup preparations, it shall set its Backup_And_Restore_State to BACKUP_FAILURE. Device A shall end the backup procedure when it detects device B's state is set to BACKUP_FAILURE.

It is a local matter as to whether device B will respond to other requests while performing a backup procedure. The exception to this is that device B is required to accept and fulfill read requests by device A that consist of accesses to device B's Device object and/or its configuration File objects. Note that Device B is allowed to return an UNKNOWN_FILE_SIZE error in response to requests for the File_Size property of any of its configuration files if the file size is unknown. Any services that are rejected due to an in-progress backup procedure will be rejected with an 'Error Class' of DEVICE and 'Error Code' of DEVICE_BUSY.

It is a local matter as to whether device B will continue to perform control actions while it is in backup mode. If device B changes its operational behavior during a backup procedure, then the System_Status property of the Device object shall be set to BACKUP_IN_PROGRESS.

19.1.2.3 Loading the Backup Parameters

Upon receipt of a 'Result(+)' response from device B to the ReinitializeDevice(STARTBACKUP, <password>) message, device A will read the Configuration_Files property of the Device object. This property will be used to determine the files to read and in what order the files will be read. The value of the Configuration_Files property is not guaranteed to contain a complete or correct set of configuration File object references before the backup request is accepted by device B.

19.1.2.4 Backing Up the Configuration Files

Once device A has determined the files that make up the device configuration image, device A will determine the type of each file and will use the AtomicReadFile service to read each configuration file from device B. Each file will be read as a stream of bytes, or as a sequence of records depending on the File_Access_Method property of the File object. Stream access files will be read in byte order and record access files will be read in record order. The files will be read in the same order as they appear in the Configuration_Files property.

It is a local matter as to what device A does with the configuration files, although the intent of the service is to allow an operator to archive the setup of device B such that device B may be restored at a later date should its configuration become corrupt.

It is left up to the implementor of device A as to whether the files read from device B will be made available for examination by tools developed by the implementor of device B. It is recommended that record access files be stored on device A as a sequence of BACnet OCTET STRINGS.

19.1.2.5 Ending the Backup Procedure

When all of the configuration files have been read, device A sends a ReinitializeDevice(ENDBACKUP, <password>) message to device B. Device B will perform whatever actions are required to complete the backup in order to place the device back into the state it was in before the backup procedure or into any other state as defined by the vendor. Device B shall not remain in the BACKUP_IN_PROGRESS mode after the backup procedure has ended.

If device A needs to abort the backup for any reason (i.e. the user aborts the procedure, device B fails to allow reads from a configuration file, or device A detects any other condition that inhibits the backup procedure), device A shall attempt to send ReinitializeDevice(ENDBACKUP, <password>) to device B. Upon receipt of this message, device B shall end the backup procedure. If the backup procedure is aborted, device A should not assume that the configuration files are still valid and continue to read them.

The receipt of the ReinitializeDevice(ENDBACKUP, <password>) message shall cause device B to exit backup mode.

If device B does not receive any messages related to the backup procedure from device A for the number of seconds specified in the Backup_Failure_Timeout property of its Device object, device B should assume that the backup procedure has been aborted, and device B should exit backup mode. A message related to the backup procedure is

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

defined to be any ReadProperty, ReadPropertyMultiple, WriteProperty, WritePropertyMultiple, CreateObject, or AtomicReadFile request that directly accesses a configuration File object.

When the backup procedure ends, device B shall set its Backup_and_Restore_State to IDLE. The deletion of configuration File objects during the backup procedure shall not have an effect on the Database_Revision property.

19.1.3 Restore

For the purposes of this discussion, the device performing the restore procedure will be referred to as device A, and the device being restored will be device B.

19.1.3.1 Initiation of the Restore Procedure

Device A sends a ReinitializeDevice(STARTRESTORE, <password>) message to device B. Device A will await a response from device B before continuing the restore procedure.

19.1.3.2 Preparation for Restore

Before starting a restore procedure, device A shall read the Restore_Preparation_Time property from device B's Device object. If the property is not present in device B, the value shall be assumed to be 0.

Upon receipt of a restore request, if device B is able to perform a restore procedure, device B shall respond with a 'Result(+)' to the ReinitializeDevice service request. Device B shall set its Backup_And_Restore_State to PREPARING_FOR_RESTORE.

If device B is unable to perform a restore procedure, then it will respond to the ReinitializeDevice service request with a 'Result(-)' response. Assuming device B supports the restore procedure and the request was properly formulated, the valid Error Class:Error Codes that can be returned are:

DEVICE:CONFIGURATION_IN_PROGRESS - if device B is already processing a backup or a restore request.

SECURITY:PASSWORD_FAILURE - if the password that was provided was incorrect or if a password is required and one was not provided.

After device B responds to the ReinitializeDevice request with a 'Result(+)', device B has Restore_Preparation_Time seconds to prepare for the restore procedure. During this period of time, device B is not required to respond to any BACnet service requests. Once this period of time elapses, device B is required to respond to read requests for properties of the Device object. When device B has completed its restore preparations in their entirety, the configuration File objects shall exist in the device, or device B shall be able to accept CreateObject requests from device A to create the configuration File objects, and the Backup_And_Restore_State property shall be set to PERFORMING_A_RESTORE. Once device B changes its Backup_And_Restore_State to PERFORMING_A_RESTORE, it shall not ignore requests from device A regardless of whether the Restore_Preparation_Time time period has expired. The creation of configuration File objects during the Restore procedure, whether automatically created by the device or by the execution of the CreateObject service, shall not impact the value of the Database_Revision property.

If device B is unable to successfully complete its restore preparations, it shall set its Backup_And_Restore_State to RESTORE_FAILURE. Device A shall abort the restore procedure when it detects device B's state is set to RESTORE_FAILURE.

Upon receipt of a Result(+), device A shall monitor the Backup_And_Restore_State property and not continue with the restore until the property contains the value PERFORMING_A_RESTORE. During the time period immediately following the Result(+) defined by the Restore_Preparation_Time, device B is allowed to ignore requests from device A and as such device A shall not consider a lack of response during this period to be an error condition. It is a local matter whether device A initiates the monitoring of the Backup_And_Restore_State property during or after this time period.

It is a local matter as to whether device B will respond to other requests while it is in restore mode. The exception to this is that device B shall accept and fulfill read and write requests by device A that consist of accesses to device B's Device object and/or its configuration File objects. Any services that are rejected due to an in-progress backup procedure will be rejected with an 'Error Class' of DEVICE and 'Error Code' of CONFIGURATION_IN_PROGRESS.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device B shall be prepared to answer device A's requests for information from device B's Device object. If device B cannot service requests from devices other than device A, then device B shall reject those services with an 'Error Class' of DEVICE and an 'Error Code' of CONFIGURATION_IN_PROGRESS.

It is a local matter as to whether device B will continue to perform control actions while it is in restore mode. If device B changes its operational behavior during a restore procedure, then the System_Status property of the Device object shall be set to DOWNLOAD_IN_PROGRESS.

19.1.3.3 Restoring the Configuration Files

Device A will use the AtomicWriteFile service to write each configuration file to device B. If any of the files do not exist in device B, then device A will attempt to create the files using the CreateObject service. Any files that already exist in the device, and differ in size from the image being written to them, will be truncated by writing 0 to the File_Size property of the File object before the contents are written to the file.

The configuration files will be written as a stream of bytes, or as a sequence of records, depending on the value of the File_Access_Method property of the File object. Note that there is no standard file format for record-based files, whereas any file can be written as a stream of bytes.

Each configuration file written to the device should be a valid configuration file obtained from the vendor, from a vendor's configuration tool, or from a previous backup procedure. The files will be written to the device in the same order as they were retrieved during the backup procedure, or as specified by the vendor if the files were obtained from another source.

Device B is allowed to reject any write operation to the configuration file if it has determined that the content of the write is invalid (internal CRC error, Invalid type code, etc.). If this is the case, device B will respond with an 'Error Class' of SERVICES and an 'Error Code' of INVALID_CONFIGURATION_DATA. It is a local matter as to whether device A will retry the request and how many times device A will retry, but device A should abort the restore procedure if device B continues to return an error.

19.1.3.4 Ending the Restore Procedure

When device A has completely written all of the configuration files to device B, device A shall send ReinitializeDevice(ENDRESTORE, <password>). Device B will perform whatever actions are required to complete the restore procedure within Restore_Completion_Time seconds after responding with a Result(+), which should include a validation of the restored configuration. If the validation fails, it is a local matter as to what device B will do beyond changing its System_Status property to something other than DOWNLOAD_IN_PROGRESS.

If device A needs to abort the restore for any reason (i.e. the user aborts the procedure, device B fails to allow writes to a configuration file, or device A detects any other condition that inhibits the restore procedure), device A shall attempt to send ReinitializeDevice(ABORTRESTORE, <password>) to device B. Upon receipt of this message, device B shall abort the restore procedure within Restore_Completion_Time seconds after responding with a Result(+).

If device B does not receive any messages related to the restore procedure from device A for the number of seconds specified in the Backup_Failure_Timeout property of its Device object, device B should assume that the restore procedure has been aborted, and device B should exit restore mode. A message related to the restore procedure is defined to be any ReadProperty, ReadPropertyMultiple, WriteProperty, WritePropertyMultiple, CreateObject, or AtomicWriteFile request that directly accesses a configuration File object.

When the restore procedure ends successfully, device B shall set its Backup_and_Restore_State to IDLE and shall set the value of the Database_Revision property to the value it had before the restore, and then increment it.

Once the restore procedure has ended, whether it was successful or not, device B shall change its System_Status property to something other than DOWNLOAD_IN_PROGRESS.

If the restore is successful, no other actions by device A shall be required, and device B will update the Last_Restore_Time property in its Device object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If the restore failed or was aborted and device B is unable to recover its old configuration, or cannot establish a default configuration, device B shall set its System_Status to DOWNLOAD_REQUIRED. Every attempt shall be made to leave device B in a state that will accept additional restore procedures.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.2 Command Prioritization

In building control systems, an object may be manipulated by a number of entities. For example, the present value of a Binary Output object may be set by several applications, such as demand metering, optimum start/stop, etc. Each such application program has a well-defined function it needs to perform. When the actions of two or more application programs conflict with regard to the value of a property, there is a need to arbitrate between them. The objective of the arbitration process is to ensure the desired behavior of an object that is manipulated by several program (or non-program) entities. For example, a start/stop program may specify that a particular Binary Output should be ON, while demand metering may specify that the same Binary Output should be OFF. In this case, the OFF should take precedence. An operator may be able to override the demand metering program and force the Binary Output ON, in which case the ON should take precedence.

In BACnet, this arbitration is provided by a prioritization scheme that assigns varying levels of priorities to commanding entities on a system-wide basis. Each object that contains a commandable property is responsible for acting upon prioritized commands in the order of their priorities. While there is a trade-off between the complexity and the robustness of any such mechanism, the scheme described here is intended to be effective but applicable to even simple BACnet devices.

The following property types are involved in the prioritization mechanism:

(a) Commandable Property: Each object that supports command prioritization has one or more distinguished properties that are referred to as "Commandable Properties." The value of these properties is controlled by the command prioritization mechanism.
(b) Priority_Array: This property is a read-only array that contains prioritized commands or NULLs in the order of decreasing priority. The highest priority (lowest array index) with a non-NULL value is the active command.
(c) Relinquish_Default: This property shall be of the same datatype (and engineering units) as the Commandable Property. When all entries in the Priority_Array are NULL, the value of the Commandable Property shall have the value specified by the Relinquish_Default property.

Although the Command object is used to write a set of values to a group of object properties, command prioritization is not involved unless the properties are commandable.

19.2.1 Prioritization Mechanism

For BACnet objects, commands are prioritized based upon a fixed number of priorities that are assigned to command-issuing entities. A prioritized command (one that is directed at a commandable property of an object) is performed via a WriteProperty service request or a WritePropertyMultiple service request. The request primitive includes a conditional 'Priority' parameter that ranges from 1 to 16. Each commandable property of an object has an associated priority table that is represented by a Priority_Array property. The Priority_Array consists of an array of commanded values in order of decreasing priority. The first value in the array corresponds to priority 1 (highest), the second value corresponds to priority 2, and so on, to the sixteenth value that corresponds to priority 16 (lowest).

An entry in the Priority_Array may have a commanded value or a NULL. A NULL value indicates that there is no existing command at that priority. An object continuously monitors all entries within the priority table in order to locate the entry with the highest priority non-NULL value and sets the commandable property to this value.

A commanding entity (application program, operator, etc.) may issue a command to write to the commandable property of an object, or it may relinquish a command issued earlier. Relinquishing of a command is performed by a write operation similar to the command itself, except that the commandable property value is NULL. Relinquishing a command places a NULL value in the Priority_Array corresponding to the appropriate priority. This prioritization approach shall be applied to local actions that change the value of commandable properties as well as to write operations via BACnet services.

If an attempt is made to write to a commandable property without explicitly specifying the priority, a default priority of 16 (the lowest priority) shall be assumed. If an attempt is made to write to a property that is not commandable with a specified priority, the priority shall be ignored. The Priority_Array property is read-only. Its values are changed indirectly by writing to the commandable property itself.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

If an attempt is made to relinquish a non-commandable property and for which NULL is not a supported datatype, if no other error conditions exist, the property shall not be changed, and the write to that property shall be considered successful.

19.2.1.1 Commandable Properties

The prioritization scheme is applied to certain properties of objects. The standard commandable properties and objects are as follows:

OBJECT	COMMANDABLE PROPERTY
Access Door	Present_Value
Analog Output	Present_Value
Analog Value	Present_Value
Binary Lighting Output	Present_Value
Binary Output	Present_Value
Binary Value	Present_Value
BitString Value	Present_Value
Channel	Present_Value (see Clause 19.2.1.6)
CharacterString Value	Present_Value
Date Value	Present_Value
Date Pattern Value	Present_Value
DateTime Value	Present_Value
DateTime Pattern Value	Present_Value
Integer Value	Present_Value
Large Analog Value	Present_Value
Lighting Output	Present_Value
Multi-state Output	Present_Value
Multi-state Value	Present_Value
OctetString Value	Present_Value
Positive Integer Value	Present_Value
Time Value	Present_Value
Time Pattern Value	Present_Value

The designated properties of the Access Door, Analog Output, Binary Lighting Output, Binary Output, Lighting Output, and Multi-state Output objects are commandable (prioritized) by definition. The designated properties of the Analog Value, Binary Value, Multi-state Value, BitString Value, CharacterString Value, Date Value, Date Pattern Value, DateTime Value, DateTime Pattern Value, Large Analog Value, OctetString Value, Integer Value, Time Value, Time Pattern Value, and Positive Integer Value objects may optionally be commandable. Individual vendors, however, may decide to apply prioritization to any of the vendor-specified properties. These additional commandable properties shall have associated Priority_Array and Relinquish_Default properties with appropriate names. See Clause 23.3. The Channel object is a special exception, see Clause 19.2.1.6.

19.2.1.2 Prioritized Commands

Prioritized commands, i.e. commands directed at commandable properties, are either WriteProperty service requests or WritePropertyMultiple service requests. In either case, the request primitive shall contain (among others) the following parameters:

Property Identifier:	Commandable_Property
Property Value:	Desired Value
Priority:	Priority

The end result of a successful write operation is to place a desired value in the priority table at the appropriate priority. If another value was already present at that priority, it shall be overwritten with the new value, without any regard to the identity of the previous commanding entity.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.2.1.3 Relinquish Commands

When a commanding entity no longer desires to control a commandable property, it issues a relinquish command. A relinquish command is also either a WriteProperty service request or a WritePropertyMultiple service request. In either case, the request primitive shall contain (among others) the following parameters:

Property Identifier:	Commandable_Property
Property Value:	NULL
Priority:	Priority

The placement of NULL in the value parameter indicates the absence of any command at that priority. When all elements of the priority table array contain NULL, the commandable property shall assume the value defined in the Relinquish_Default property of the object.

It is possible for an application entity to relinquish at a priority other than its own, resulting in unpredictable behavior. If more than one application is assigned the same priority, it is possible for one application entity to write-over (or relinquish) the commands from the other application entity, resulting in unpredictable operation. To minimize this possibility, it is very important not to allow more than one commanding entity to assume the same priority level within the system.

19.2.1.4 Value Source

The optional identification of value sources is described in Clause 19.5.

19.2.1.5 Command Overwrite

Whenever a command is issued to a commandable property, the value is placed in the Priority_Array at the appropriate priority position, without any regard to the current value residing there. The new command overwrites the existing command. No notification of such overwrite is made to the original commanding entity.

19.2.1.6 Prioritization for Channel Objects

Channel objects have commandable Present_Value properties, even though the Channel object itself does not contain Priority_Array or Relinquish_Default properties. The Channel object passes the value written to Present_Value on to another object property, which may itself be commandable. In this case, any priority provided when the Channel object Present_Value is written is propagated on to its constituent member references. The Last_Priority property of the Channel object remembers the most recently provided priority value.

19.2.2 Application Priority Assignments

Commanding entities are assigned one of the 16 possible priority levels. The assignment of most priorities is site dependent and represents the objectives of the site management. Table 19-1 contains the standard priorities. Other applications that need prioritization include Temperature Override, Demand Limiting, Optimum Start/Stop, Duty Cycling, and Scheduling. The relative priorities of these applications may vary from site to site and are not standardized. For interoperability at any particular site, the only requirement is that all devices implement the same priority scheme. The positions marked Available are open for assignment to DDC programs, EMS programs, etc. The interpretation of what conditions constitute Manual-Life Safety or Automatic-Life Safety decisions is a local matter. All commandable objects within a device shall be configurable to accept writes to all priorities except priority 6.

Table 19-1. Standard Command Priorities

Priority Level	Application	Priority Level	Application
1	Manual-Life Safety	9	Available
2	Automatic-Life Safety	10	Available
3	Available	11	Available
4	Available	12	Available

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Priority Level	Application	Priority Level	Application
5	Critical Equipment Control	13	Available
6	Minimum On/Off	14	Available
7	Available	15	Available
8	Manual Operator	16	Available

19.2.3 Minimum_On_Time and Minimum_Off_Time

If the commandable property is the Present_Value property of a Binary Output object or a Binary Value object and that object possesses the optional Minimum_On_Time and Minimum_Off_Time properties, then minimum on and minimum off times shall behave according to the algorithm described in this clause.

Command priority 6 is reserved for use by this algorithm and may not be used for other purposes in any object.

(a) the Present_Value is ACTIVE and the time since the last change of state of the Present_Value is less than the Minimum_On_Time, then element 6 of the Priority_Array shall contain a value of ACTIVE.
(b) If the Present_Value is INACTIVE and the time since the last change of state of the Present_Value is less than the Minimum_Off_Time, then element 6 of the Priority_Array shall contain a value of INACTIVE.
(c) If neither (a) nor (b) is true, then element 6 of the Priority_Array shall contain a value of NULL.

These rules imply actions to be taken when the Present_Value is written and actions to be taken based on elapsed time. The means by which these actions are implemented is a local matter, so long as the behavior described in this clause is achieved.

When a write to a commandable property occurs at any priority, the specified value or relinquish (NULL) is always written to the appropriate slot in the priority table, regardless of any minimum on or off times.

The Priority_Array is then examined by the local priority maintenance entity to determine the highest priority that contains a non-NULL value. If this value differs from the Present_Value immediately before the write, then a change of state has occurred. If such a change of state occurs, the new value is also written to priority 6 in the Priority_Array and the time of the change is noted. The means by which the timing is performed is a local matter.

When the minimum on or off time signified by a non-NULL value in priority 6 has elapsed, the local minimum time maintenance entity shall write a NULL to priority 6 and re-examine the Priority_Array to determine the new Present_Value. If this value indicates a change of state, then the appropriate actions shall be taken as described above.

The effect of a non-NULL value in priority 6 is that writes at any lower priority (larger priority number) cannot cause a change of state. Thus, minimum on or off time protection may be achieved relative to these priorities.

Writes to any priority higher than 6 (smaller priority number) may cause changes of state regardless of Minimum_On_Time or Minimum_Off_Time. Thus, these priorities should be used only for critical or emergency use. Note, however, that changes of state caused by a write to these high priorities will also cause writes to priority 6 as described above. Thus, if a NULL is subsequently written to the high priority while minimum time is in effect, that time shall be observed before any change of state is made as a result of a value at a lower priority.

For additional discussion of minimum on and off time processing see Annex I.

19.2.4 Prioritization for Command Objects

A Command object is capable of issuing commands just as any other command-issuing entity. A Command object may be related to an application with any priority. The Action property of the Command object contains all of the parameters necessary for writing to commandable properties. See Clause 12.10.8.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.2.5 Prioritization for Loop Objects

Loop objects may need to interact with objects that have a commandable property, even though, in general, they will not use BACnet services to do so. Each Loop object has a Priority_For_Writing property that designates the appropriate priority of this control loop with respect to the commandable property. See Clause 12.17.28.

19.2.6 Prioritization for Schedule Objects

Schedule objects may need to interact with objects that have a commandable property, even though, in general, they will not use BACnet services to do so. Each Schedule object has a Priority_For_Writing property that designates the appropriate priority of this schedule with respect to the commandable property. See Clause 12.24.11.

19.2.7 Prioritization for Access Point Objects

Access Point objects interact with the Present_Value property of Access Door objects; however, if the Access Door objects are local to the device, they will not use BACnet services to do so. Each Access Point object has a Priority_For_Writing property that designates the priority to be used to command the Access Door objects.

19.2.8 Prioritization for Staged ValueObjects

Staged Value objects may need to interact with objects that have a commandable property, even though, in general, they will not use BACnet services to do so. Each Staged Value object has a Priority_For_Writing property that designates the appropriate priority of this staged value with respect to the commandable property. See Clause 12.62.15.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.3 Device Restart Procedure

When a BACnet device restarts, there are a number of different configuration items that can be lost. For example, a device need not remember which devices have subscribed to receive change-of-value notifications or to which values they have subscribed. For this reason, other devices may be interested in determining when a device has restarted. This section outlines how a device may interoperably indicate that it has restarted.

When a device is powered on, when it restarts due to a ReinitializeDevice service (ACTIVATE_CHANGES, COLDSTART or WARMSTART), or when it restarts for some other reason, the device shall transmit an UnconfirmedCOVNotification request. The 'Subscriber Process Identifier' parameter shall be 0, the 'Monitored Object Identifier' parameter shall reference the Device object, the 'Time Remaining' parameter shall be 0, and the 'List of Values' parameter shall contain three values, the System_Status, the Time_Of_Device_Restart, and the Last_Restart_Reason properties of the Device object. The device shall transmit this message after the complete power-up or restart sequence has been completed so that the system-status value is accurate.

The device shall send the restart notification to each recipient in the Restart_Notification_Recipients property of the Device object.

MS/TP slave devices are not able to support this procedure, although they may support the Time_Of_Device_Restart and Last_Restart_Reason properties.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.4 Determining Maximum Conveyable APDU

This clause describes a method for determining the maximum conveyable APDU size to a remote network. This size may be affected by the existence of routers in the path to the remote network that are incapable of handling the maximum APDU size of the end nodes. If this method is used by numerous devices, it consumes considerable communications bandwidth and should be used sparingly since normal communications may be disrupted when used excessively. The method should only be performed when necessary and preferably after a randomly chosen delay in order to reduce the quantity of devices that may be using the method simultaneously. Once a node has determined the maximum conveyable APDU length for a remote network, it shall cache the determined value.

During this test, it is important that the client not generate any other messages to the remote network that are larger than the currently outstanding test message being sent. This will ensure that any Reject-Message-To-Network with a "Reject Reason" of 4 (MESSAGE_TOO_LONG) for the destination network applies to the outstanding test message.

A device on the remote network is selected which is capable of receiving a large APDU. It is a local matter as to how the client determines which device on the remote network to perform the test with.

The client first reads the remote peer's Max_APDU_Length_Accepted property. This step verifies that the remote peer is online and reachable from the client. If no response is received from the remote peer, a different remote peer needs to be selected and the test restarted.

The client then generates a request of a size equal to the smaller of the local maximum APDU the client supports and the maximum APDU length supported by the remote peer. The preferred message to send is a ConfirmedPrivateTransfer-Request with a "Vendor ID" of 0, a "Service Number" of 0 and "Service Parameters" containing a single application tagged OCTET_STRING with N data octets, where N is the number of octets required to generate an APDU of the desired size. This message is reserved by ASHRAE for this use and as such will not result in a change of state of the remote peer (it will have no net effect on the remote peer's operation).

Table 19-2. Calculating Test Message OCTET STRING Size

Desired_APDU_Size	Number of OCTET STRING Data Octets (excluding tag octets)
Up to 263 octets	Desired_APDU_Size - 12
Over 263 octets	Desired_APDU_Size - 14

19.4.1 Example ConfirmedPrivateTransfer Service

This is an example test message using a BACnet confirmed service.

Service = ConfirmedPrivateTransfer
'VendorID' = 0
'ServiceNumber' = 0
'ServiceParameters' = (An OCTET STRING with length calculated according to Table 19-2, e.g., 1476 - 14 = 1462)

19.4.2 Encoding for Example

X'00' PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'05' Maximum APDU Size Accepted=1476 octets
X'55' Invoke ID=85
X'12' Service Choice=18 (ConfirmedPrivateTransfer)

X'09' SD Context Tag 0 (Vendor ID, L=1)
X'00' 0 (ASHRAE)
X'19' SD Context Tag 1 (Service Number, L=1)
X'00' 0 (ASHRAE Max Conveyable APDU Test Message)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'2E' PD Opening Tag 2 (Service Parameters)
X'65' Application Tag 6 (Octet String, L>4)
X'FE' Extended Length > 253
X'05B6' Extended Length = 1462
X'00' ... '00' (1462 octets of octet string data)
X'2F' PD Closing Tag 2 (Service Parameters)

19.4.3 Procedure

Upon sending such a request, the BACnet device can expect one of the following situations to occur:

(a) The remote peer responds. Any response by the remote peer, be it a positive or negative response, indicates that the request was successfully conveyed through the internetwork to the peer.
(b) A Reject-Message-To-Network with a "Reject Reason" of 4 for the destination network is received. This indicates that the message cannot be conveyed through the internetwork to the peer because it is too large.
(c) A Reject-Message-To-Network with a "Reject Reason" of a value other than 4 for the destination network is received indicating that some other failure is stopping completion of the test.
(d) No response is received from the remote peer.

If situation 3 occurs, the test cannot continue and the maximum conveyable APDU cannot be determined at this time.

If situation 2 or 4 occurs, the message was too large. A shorter ConfirmedPrivateTransfer should be sent. Given that BACnet networks have specific maximum conveyable APDU sizes, the next size to check should be the next lower maximum conveyable APDU for the standard BACnet data link types. See Clause 20.1.2.5 for a list of suggested APDU lengths to test.

If situation 4 repeatedly occurs, regardless of the size of test message used, either the remote peer is now offline, the network has become disjoint, or some other catastrophic failure is occurring. The test should be restarted with a different selected remote peer device.

Clients that send messages to determine the maximum conveyable APDU length for remote networks shall cache the determined values so as to not have to repeat the test whenever the information is needed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.5 Value Source Mechanism

In building control systems, an object may be manipulated by a number of entities, and it may be important to know which entity is the current source of the current Present_Value (or Mode in the case of Life Safety objects). The priority at which an object is commanded indicates the general process that is controlling it (scheduling, manual override, etc.), the value source mechanism allows for an object to indicate the device, and/or object that was the source of the command or write operation.

The rest of this clause refers to the Present_Value property when describing the value source mechanism. This is done for clarity of the specification. When considering the mechanism for objects which apply it to a different property (such as Mode in the case of Life Safety objects), references to Present_Value are to be understood as references to the property that the mechanism is actually applied to.

The Channel object is a commandable object type but instead of having a Priority_Array, it just passes the priority through when writing to target properties. As such, it behaves like a non-commandable object when considering the value source mechanism. When the Channel object writes or commands objects, the value source for the operation is the Channel object.

By default, the value source mechanism only identifies the device that was the source of the write or command operation. The mechanism does allow, but does not require, client devices to update the value source information to accurately identify the object that is the true source of the operation.

19.5.1 Value Source Properties

For commandable Present_Value properties, the value source mechanism works in conjunction with command prioritization to identify which device or object has commanded, or relinquished, a value at each of the priority levels.

For non-commandable Present_Value properties, the value source mechanism identifies the source of the last write operation.

19.5.1.1 Present_Value, Priority_Array, Relinquish_Default

See Clause 19.2 for a description of command prioritization.

19.5.1.2 Value_Source

The Value_Source property indicates the source of the current Present_Value. The value indicates either the device or the object that provided the current value of the Present_Value property.

For commandable Present_Value properties, the value of the property is taken from the entry of the Value_Source_Array which is related to the Priority_Array entry which provided the current Present_Value. For example, if the Present_Value is from index 8 of the Priority_Array, then the Value_Source property shall have the value from index 8 of the Value_Source_Array.

If there is no active value source, i.e. the Present_Value has taken on the value of Relinquish_Default, then the Value_Source property shall have the value 'None'.

For non-commandable Present_Value properties, the Value_Source is either the value that is set when the Present_Value was last written, or is the value provided by the source device after the Present_Value was last written. In objects with a non-commandable Present_Value property, there will be no Value_Source_Array property.

19.5.1.3 Value_Source_Array

The Value_Source_Array property holds the last value source for each priority. When the Present_Value is commanded, the Value_Source_Array is updated to indicate where the command originated.

When a property is written or commanded, the value source information is set to the source device's Device object, if known, or the source device's network address if the device instance is not known.

When Present_Value is commanded or written by the local device, the value source information shall be updated. If the operation was initiated by a local object, then that object shall be identified in the value source information.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

When minimum on / off control takes effect, or relinquishes, the value source used for priority 6 is the commanded object (i.e. the object which contains the Value_Source_Array property.)

After commanding or writing the Present_Value of an object, the device, and only the device which wrote or commanded the value, may update the value source information, at the same priority as the command in case of commanding, to set the source device instance or to indicate the object that initiated the operation. The writing or commanding device accomplishes this by writing to the Value_Source property at the same priority that the Present_Value was written or commanded at. Writing to the Value_Source property may be requested by a subsequent WriteProperty request or, when WritePropertyMultiple is used, by a respective property value subsequent to the property value written or commanded. Attempts to write to the Value_Source property by a device other than the device that wrote the property or commanded the property at a particular priority shall cause an error being returned and the write to Value_Source shall not be performed.

For commandable objects, when the Value_Source is written, the value is placed into Value_Source_Array at the entry that corresponds to the priority it was written and the Value_Source property will be updated only if that is the current command priority.

19.5.1.4 Last_Command_Time

Last_Command_Time indicates the time at which the Present_Value, Current_Command_Priority, or Value_Source last changed due to a command or write of Present_Value. The Last_Command_Time is not updated when Value_Source is written directly. The Last_Command_Time shall only be present in commandable objects that contain the Priority_Array property. It shall not be present in commandable objects without Priority_Arrays or that pass-through priority such as Channel objects.

Note that when Present_Value changes due to a relinquish command, the Last_Command_Time indicates the time at which the relinquish command was received and not the earlier time at which the now current priority command was provided.

19.5.1.5 Command_Time_Array

The Command_Time_Array property records the Last_Command_Time for each incoming priority write. The Command_Time_Array shall only be present in commandable objects that contain the Priority_Array property. It shall not be present in commandable objects without Priority_Arrays or that pass-through priority such as Channel objects.

19.5.2 Change of Value Reporting With Value Source Information

To allow reporting of value source information along with the value of the property, COV subscriptions for the Value_Source property are handled differently than for other properties.

When Value_Source is subscribed to, via SubscribeCOVProperty, COV notifications will be sent under the same circumstances as COV for the Present_Value of the object, and whenever the value of Value_Source changes. See Table 13-1a-2 for more information on COV reporting for the Value_Source property.

It is a local matter whether 1 or 2 notifications are generated in the case where the Value_Source property is written directly with updated information. In the case where a single notification is generated, that notification shall include the updated value source.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.6 Audit Logging

Audit logging consists of recording records in audit logs which document the sequence of activities that affect the system.

BACnet audit logging is made up of four actors: the client that requests the operation being logged (the operation source), the server that performs the operation (the operation target), the logger that stores the audit records, and the viewer that consumes the audit log. In this framework, both the operation source and the operation target are given the opportunity to provide audit log records to the audit logger. This allows BACnet devices to provide these features:

(a) audit logging when either the operation source or the operation target does not support audit logging;
(b) logging of extended audit information that is not available for reporting by the operation target;
(c) recording of failed actions by the operation source when the operation target does not respond;
(d) recording of failed actions by the operation target where unauthorized operation sources do not log their attempts.

The operation source generates audit log entries which describe the operation it requested and which optionally contain extra information including the BACnet object which is the source of the operation (if applicable), user identification (if applicable), and a description of, or reason for, the operation.

The server generates audit log entries which describe the operation it has been requested to perform.

The logger receives notifications from local and remote operation sources and operation targets and places the notifications in Audit Log objects.

Operation source side auditing is controlled via:

(a) an Audit Reporter object in the operation source device which generates the audit notifications; and
(b) the Audit_Notification_Recipient property in the operation source's Device object.

Operation target side auditing is controlled via:

(a) configuration properties in the object operated upon by the operation being reported;
(b) Audit Reporter objects which generate the audit notifications; and
(c) the Audit_Notification_Recipient property in the operation target's Device object.

The following figures provide a diagrammatic overview of the architecture discussed in the rest of this clause.

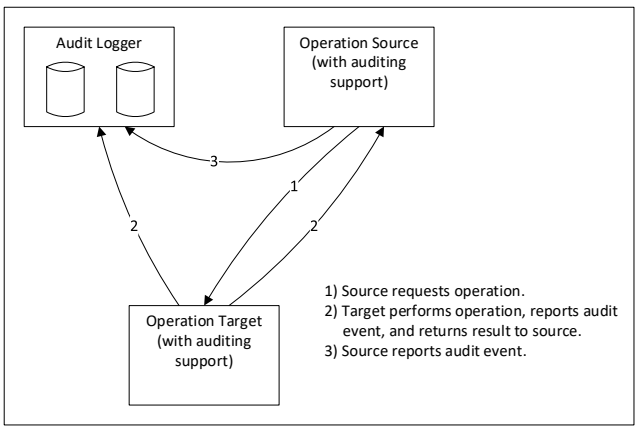


Figure 19-1: General flow of operations and audit notifications from both operation sources and operation targets.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

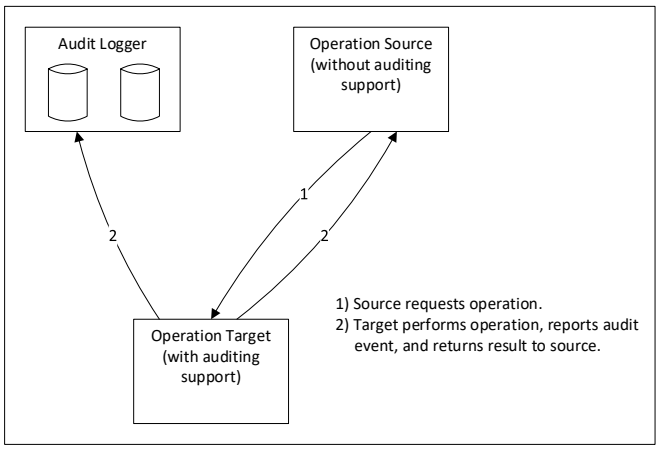


Figure 19-2: General flow of operations and audit notifications from operation targets with no operation source notifications.

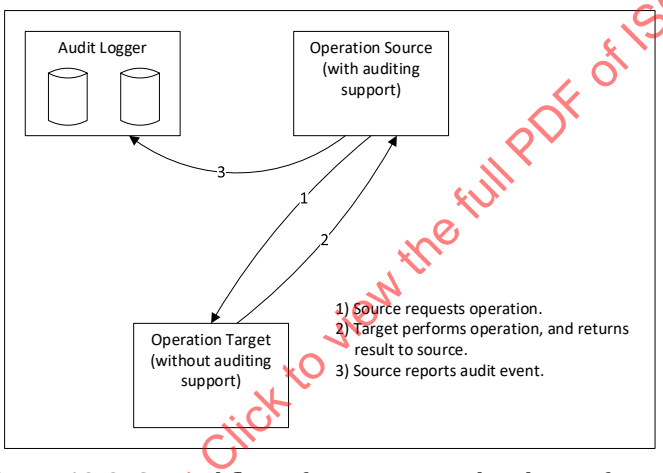


Figure 19-3: General flow of operations and audit notifications from operation source only.

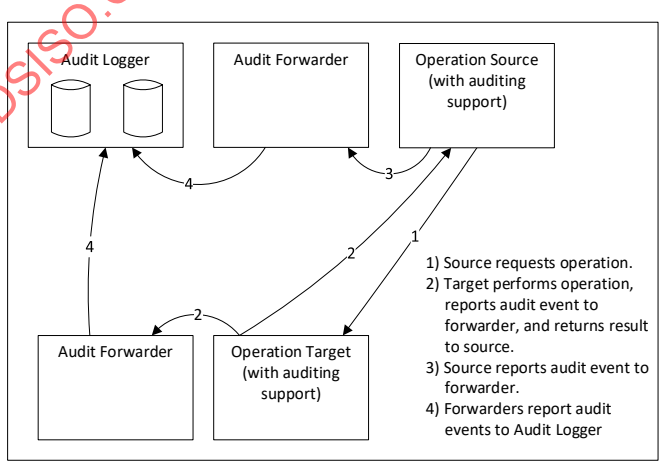


Figure 19-4: General flow of operations and audit notifications from both operation sources and operation targets with audit forwarders.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.6.1 Audit Notification Generation

Operation targets generate audit notifications for the following operations (subject to filtering):

(a) a BACnet request for an operation type described in Table 19-5, which is executed successfully;
(b) a local request for an operation type described in Table 19-5, which is successfully performed but which is not the result of the normal operation of the object (for example, updating of an input object's Present_Value based on reading of the physical input would not be auditable, but the setting of the Present_Value of a value object by a local program would be auditable); and
(c) any action the local device determines to be relevant in an audit log.

It is a local matter whether or not an operation target generates audit notifications for BACnet requests for an operation type described in Table 19-5, which fail. It is important to consider when such operations will result in notifications as this could form the basis of a denial of service attack through the increased workload and network traffic required to report the failed operations.

Operation Sources generate audit notifications for the following operations (subject to filtering):

(a) a BACnet request for an operation type described in Table 19-5, which is requested; and
(b) any action the local device determines to be relevant in an audit log.

When reporting a successful operation, no result code is included in the audit notification. When reporting a failed operation, a result code shall be included.

19.6.1.1 Audit Notification Generation Through Monitoring

It is possible for a device that is not involved in the actions to monitor the network traffic and generate audit notifications on behalf of operation sources and/or operation targets.

Such a product is installed so that it is able to monitor all communications of interest that originate with an operation source (or all operation sources), or all communications destined for an operation target (or all operation targets). When BACnet messages are detected which contain auditable actions, the product generates audit records as if it were the operation source or the operation target device (depending on which device it is performing auditing for.)

The configuration and control of products that provide such functionality is outside the scope of this standard.

19.6.2 Audit Reporter Objects

Audit notifications are generated by Audit Reporter objects. Both operation source and operation target devices which generate audit notifications contain Audit Reporter objects.

In the most basic case, there will be a single Audit Reporter object responsible for all audit notifications. In complex scenarios, a device may have multiple Audit Reporter objects each generating audit notifications for a different set of objects to allow for a finer granularity of control over audit notifications.

When a device supports audit reporting and is a BACnet client, exactly one Audit Reporter object shall have the Audit_Source_Reporter property set to TRUE and is used for configuration of operation source audit reporting. An Audit Reporter object with this flag set shall not be deletable. If creation of Audit Reporter objects is supported by the device, any created Audit Reporter objects shall have the Audit_Source_Reporter flag set to FALSE.

All write operations to an Audit Reporter object shall result in an audit notification by the containing device, if Audit_Level is not NONE.

Audit Reporter objects may optionally support delaying the sending of audit notifications so that multiple notifications can be sent in a single request. The Maximum_Send_Delay property limits how long audit notifications may be delayed.

19.6.3 Audit Notification Configuration

Any device which supports auditing shall contain at least one Audit Reporter object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Auditing can be configured on a per Audit Reporter level and on a per object level. Group enabling and disabling of audit reporting is controlled via the Audit_Level and Auditable_Operations properties of the Audit Reporter objects.

Audit_Level controls the level of audit reporting at the Audit Reporter or at the object level. When the Audit_Level property in the Audit Reporter object is NONE, the Audit Reporter object shall not generate any audit notifications, with the exception of generating a notification when its Audit_Level property is modified. When the Audit_Level property in an object (other than an Audit Reporter object) is NONE, the device shall not generate any audit notifications for the object, with the exception of generating a notification when the object's Audit_Level is modified.

The Auditable_Operations property in an object controls which operations on the object an Audit Reporter object generates audit notifications for. If the property is not present, then the Auditable_Operations property in the associated Audit Reporter object is used instead. Irrespective of the value of an Auditable_Operations property, changes to an Auditable_Operations property will always result in an audit notification by the server device if Audit_Level is not NONE.

When a device is acting as an operation source, the Auditable_Operations property of the Audit Reporter object, with the Audit_Source_Reporter property set to TRUE, controls which operations initiated by the device will be reported. The Auditable_Operations property of the target object has no impact on the audit notifications generated by the operation source.

The Audit_Priority_Filter property specifies the priorities for which audit notifications will be generated for write operations. Any auditable write operation to the commandable property of an object with a priority associated with a bit set to 0 in this property shall not result in an audit notification. For example, if bit 7 of this property is set and bit 10 is not set, a manual override at priority 8 would result in an audit notification, whereas a WriteProperty request at priority 11 would not (the difference in the bit numbers versus the priorities in this example is due to bits being 0-based and priorities being 1-based). This property has no effect on the auditing of non-write operations nor on the auditing of operations on properties other than commandable properties. If this property is not present in an object with a commandable property, then no command priority based audit filtering will be applied. The purpose of this property is to allow normal control actions to occur without audit notifications being generated while ensuring that override actions are audited.

The Audit_Level property determines which properties in the object are auditable. Irrespective of the value of this property, changes to this property shall always result in an audit notification by the server device.

Table 19-3. Audit Level

Operation	Comment
NONE	No operations on the object will result in audit notifications except for the noted modifications to the audit configuration.
DEFAULT	The value for the audit level for this object is taken from the Audit_Level property of the associated Audit Reporter object.
AUDIT_CONFIG	Operations on configuration properties will result in audit notifications. Operations on the Present_Value will not result in audit notifications unless Present_Value has been designated as a configurable item for the product.
AUDIT_ALL	Operations on all properties will result in audit notifications.

The BACnetAuditLevel enumeration is extensible providing a method for vendors to specify new audit levels for objects that require them. No extensible audit level value shall be equivalent to NONE (result in no auditable operations on the object.)

For the audit level AUDIT_CONFIG, it is a local matter whether or not any given property in an object is considered a configuration property. This is a local matter specifically to allow implementations to make this designation based on the needs of the specific application of the product.

19.6.4 Audit Notifications

The parameters sent in an audit notification are described in Table 19-4.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table 19-4. Audit Notification Parameters

Parameter	Data Type	Value
Source Timestamp	BACnetTimeStamp	The time the operation was requested by the operation source. This field shall not be included in operation target notifications unless the source and target are the same device.
Target Timestamp	BACnetTimeStamp	The time the operation was performed by the operation target. This field shall not be included in operation source notifications unless the source and target are the same device.
Source Device	BACnetRecipient	The device requesting the operation. If known, the BACnetObjectIdentifier form shall be provided.
Source Object	BACnetObjectIdentifier	The object requesting the operation. This field shall not be included in operation target notifications, unless the source and target are the same device, or when there is no source object to report.
Operation	BACnetAuditOperation	The operation being requested.
Source Comment	CharacterString	A human readable description of the operation. This value may be system generated. For systems where human comments are required for documenting actions such as required by some pharmaceutical installations, the human entered comment shall be provided in this parameter in addition to any system generated comment. For GENERAL operation audit notifications, this field shall include a system generated description of the operation.
Target Comment	CharacterString	A human readable description of the operation. For GENERAL operation audit notifications, this field shall include a system generated description of the operation.
Invoke Id	Unsigned8	The invoke id from the service request that resulted in this audit record. This provides the ability to more easily match operation source and operation target audit records. This field shall not be included in operations consisting of unconfirmed service requests. This field shall not be included if the source and target are the same device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Parameter	Data Type	Value
Source User Id	Unsigned16	The user performing the operation. The value in this field depends on the user authentication model in use. See Clause 24.14 on user authentication for information on this value. This field shall not be included if no user identification is available.
Source User Role	Unsigned8	The role of the user performing the operation. The value in this field depends on the user authentication model in use. See Clause 24.14 on user authentication for information on this value. This field shall not be included if no user role information is available.
Target Device	BACnetRecipient	The target device of the operation. If known, the BACnetObjectIdentifier form shall be provided.
Target Object	BACnetObjectIdentifier	The target object of the operation. This field shall be absent if the target is not an object.
Target Property	BACnetPropertyReference	The target of the operation if it is targeted at a property. This field shall be absent if the target is not a property.
Target Priority	Unsigned(1..16)	For operations which include a priority (such as when a command occurs) this field is present and contains the provided priority. This field may be absent if the provided priority is 16. This field shall be absent if the target is not a property.
Target Value	ABSTRACT-SYNTAX.&Type	For operations which include a value (such as when a write occurs) this field is present and contains the new value. When the value being written is larger than 32 encoded octets, it is a local matter whether the target value is included in the audit notification. This allowance is included so that large audit notifications can be avoided where the data value is large. Audit loggers may discard this parameter value if it exceeds 500 octets when not acting as a forwarder.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Parameter	Data Type	Value
Current Value	ABSTRACT-SYNTAX.&Type	When an operation includes a value (such as when a write occurs) and the current value (before the operation) of the target property is known, this field is present and contains the current property value. When the current value is larger than 32 encoded octets, it is a local matter whether the current value is included in the audit notification. This allowance is included so that large audit notifications can be avoided where the data value is large. Audit loggers may discard this parameter value if it exceeds 500 octets when not acting as a forwarder.
Result	Error	When the operation fails, this field shall be present and indicates the error that occurred.

19.6.5 Audit Operations

The 'Operation' field of an audit notification indicates the operation requested or performed. The mapping of standard BACnet services to the BACnetAuditOperation enumeration is shown in Table 19-5. When a service allows multiple object or property targets (e.g., ReadPropertyMultiple, WritePropertyMultiple, etc.), each target shall be reported in a separate notification.

Table 19-5. Service to Audit Operation Mapping

Audit Operation	BACnet Service	Comment
READ	ReadProperty ReadPropertyMultiple ReadRange AtomicReadFile AuditLogQuery	When reporting a file operation, no 'Target Property' value is provided. This setting should be used with caution as it can result in a large increase in network traffic.
WRITE	WriteProperty WritePropertyMultiple AtomicWriteFile WriteGroup AddListElement RemoveListElement	When reporting a file operation, no 'Target Property', 'Target Value', or 'Current Value' values are provided. These entries can be discarded if the write operation does not result in a change (the 'Target Value' equals the 'Current Value', and re-writing the property has no side-effects).
CREATE	CreateObject	Initial values are not provided in a CREATE operation audit record. Initial values may be optionally reported by reporting WRITE operations.
DELETE	DeleteObject	No 'Target Property', 'Target Value', or 'Current Value' values are provided.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Audit Operation	BACnet Service	Comment
LIFE_SAFETY	LifeSafetyOperation	The 'Target Value' shall include the 'Requesting Process Identifier' and 'Request' parameters in a BACnetLifeSafetyOperationInfo sequence. The 'Source Comment' shall be set to, or include, the 'Requesting Source' parameter value.
ACKNOWLEDGE_ALARM	AcknowledgeAlarm	The 'Target Value' shall include the 'Event State Acknowledged', and 'Timestamp' parameters in a BACnetAcknowledgeAlarmInfo sequence. The 'Source Comment' shall be set to, or include, the 'Acknowledgement Source' parameter value.
DEVICE_DISABLE_COMM	DeviceCommunicationControl	Used when a DISABLE or DISABLE_INITIATION DeviceCommunicationControl operation is performed. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_ENABLE_COMM	DeviceCommunicationControl	Used when an ENABLE DeviceCommunicationControl operation is performed, or when a DISABLE or DISABLE_INITIATION DeviceCommunicationControl operation's lifetime expires. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_RESET	ReinitializeDevice	WARMSTART and COLDSTART options of the ReinitializeDevice service. An audit record is generated when the device is reset for any reason. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_BACKUP	ReinitializeDevice	START_BACKUP, ABORT_BACKUP, or END_BACKUP options of the ReinitializeDevice service. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_RESTORE	ReinitializeDevice	START_RESTORE, ABORT_RESTORE, END_RESTORE options of the ReinitializeDevice service. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Audit Operation	BACnet Service	Comment
SUBSCRIPTION	SubscribeCOV SubscribeCOVProperty	The 'Target Value' parameter shall be of type BOOLEAN and indicates whether (TRUE) the operation is a subscription, or (FALSE) the operation is an unsubscription. Subscription timeouts shall be reported as unsubscriptions. No 'Current Value' parameter is provided.
NOTIFICATION	Event Notification COV Notification	Receipt of event notification and COV notification messages are treated as source auditable operations; whereas the generation of event notifications and COV notifications are treated as target auditable operations. This setting should be used with caution as it can result in a large increase in network traffic. The 'Target Property' shall be the Event_State property, and the 'Target Value' shall be the new Event_State.
AUDITING_FAILURE		Used to report that audit records were dropped. The 'Target Timestamp' field is used to report the time of earliest record that was dropped. The 'Source Device' and 'Target Device' fields are set to reference the local device. The 'Current Value' field is set to an Unsigned count of the number of records dropped.
NETWORK_CHANGES		Used to report changes to Network Port objects which are a result of something other than application layer BACnet services (such as BVLL messages or registration timeouts).
GENERAL		Any other actions that the device needs to report to the audit log. Given the non-standard nature of this audit record type, it is important that information be formatted into the 'Comment' parameter. Where a device reports a wide variety of other actions, or there is a large variation in the expected frequency of different types of actions, it is recommended that implementers provide a method of configuring which actions result in notifications when the GENERAL operation notifications are enabled.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.6.6 High Volume Situations

Under some high traffic conditions, a device might be unable to generate audit notifications for all auditable actions due to local resource constraints. When such a condition exists, the device shall selectively drop audit records attempting to keep the most important records for distribution to the audit logger. The device shall report to the audit logger the number of dropped records using an audit notification with an operation of AUDITING_FAILURE.

Except for notifications mandated by this standard for changes in audit settings, the relative priority of audit notifications is a local matter, it is suggested that when dropping notifications, non-modification audit notifications should be dropped first (e.g., audit operation is READ, NOTIFICATION, etc.). Audit notifications that are mandated by this standard for changes in audit settings shall be deemed to be the highest priority records and the newest of these shall be the last records to be dropped.

19.6.7 Audit Logs

19.6.7.1 Audit Logger

The logging device is the central repository for audit logs. Since it is expected that audit logs will be very large, such a device is expected to have a large amount of storage dedicated to audit logging.

The general structure is that one or more Audit Log objects exist in the audit logger into which received audit notifications are placed. It is a local matter how many and which Audit Log objects any particular audit notification is placed into with the exception that, for any specific object, there is at least one Audit Log object which contains the complete audit history for the object.

Audit loggers differ from other loggers in that archiving of audit log records is accomplished by operation sources and operation targets pushing notifications to the logger and BUFFER_READY is not used to inform archival loggers of new notification records. Also, given that auditing should not be interrupted, there is no support for the Stop_When_Full function.

19.6.7.2 Audit Forwarder

An audit forwarder is a specialized audit logger that is used to reduce network traffic caused by audit logging.

The audit forwarder collects audit notifications from multiple requests into an Audit Log object and then forwards them to a parent audit logger. Once forwarded successfully, audit notifications may be removed from the forwarder's Audit Log object.

Audit forwarders are not required to support the audit querying functionality and as such are not considered a source of audit information for audit reporting. The audit forwarder is simply used for reducing the impact of audit reporting on network usage.

Audit forwarders shall not discard, or change, parameter values in audit notifications from other devices when forwarding the notifications.

While audit forwarder functionality can be placed in any device in the system, it is recommended that it be placed in routers as this minimizes the impact on network traffic.

19.6.7.3 Audit Logger Hierarchies

When the operation source and operation target devices involved in an auditable operation report audit notifications to different audit loggers, there will be two distinct records of the operation in two different audit loggers. If there is to be a consolidated log, then a hierarchy of audit loggers shall be setup. A complete log with all audit record pairs will only be available in the topmost audit logger in the hierarchy. The hierarchy is achieved through the Member_Of property of the Audit Log object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.6.8 Security of Audit Notifications

The BACnet audit logging framework does not provide any security. Security of the network shall be achieved through other means such as the security mechism described in Annex AB.

Given that values are stored in Audit Logs, care should be taken in configuring access to Audit Log objects so that information that needs to be restricted can be.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

19.7 Unconfigured Device Discovery and Address Assignment

The You-Are service provides a mechanism that a sending BACnet-user, such as a workstation, may use to specify a device identifier value across a network to a remote device in a standardized manner. The vendor identification, model name, and serial number parameters together serve to unambiguously identify the remote device. Additional parameters are supplied for the MAC address or the Device object identifier, or both.

The Who-Am-I service provides a mechanism for requesting device identifier values across a network in a standardized manner. The vendor identification, model name, and serial number parameters are included in the request to unambiguously identify this device.

The Who-Is service may be used to discover devices supporting the Who-Am-I service. The Who-Is service relies on devices in a BACnet network having both a valid Device Identifier and a network MAC address assigned to them. Unconfigured devices might require either a valid Device Identifier, a network MAC address, or both.

Prior to configuration, devices that require Device ID configuration shall only support initiation of Who-Am-I and execution of Who-Is and You-Are services, and shall use Device Identifier 4194303. Devices that require a network MAC address shall only support execution of the You-Are service. It is possible for a collection of unconfigured devices to saturate a network segment and respond with Who-Am-I service at the same time resulting in an incomplete set of discovered devices. This issue can be managed by configuring the discovered devices using the You-Are service, and repeating the Who-Is service unconfigured device discovery process.

19.7.1 Device Identifier Assignment

A Device in a BACnet network might have a network MAC address, but require a Device Identifier, and still be connected to the network. Discovering these unconfigured devices may be performed by using the Who-Is service parameters Device Instance Range Low Limit with a value of 4194303, and Device Instance Range High Limit with a value of 4194303. These unconfigured devices respond with Who-Am-I service. The discovered devices can then be assigned a valid Device Identifier using the You-Are service.

19.7.2 Network MAC Address Assignment

A Device in a BACnet network might have a Device Identifier, but require a network MAC address, and still be connected to the network. These devices may be remotely assigned a valid network MAC address using the You-Are service.

19.7.3 Device Identifier and Network MAC Address Assignment

A Device in a BACnet network might require a Device Identifier and a network MAC address, and still be connected to the network. These devices may be remotely assigned a valid Device Identifier and a network MAC address using the You-Are service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20 ENCODING BACnet PROTOCOL DATA UNITS

Application Layer Protocol Data Units (APDUs) are used in BACnet to convey the information contained in the application service primitives and associated parameters.

ISO Standard 8824, Specification of Abstract Syntax Notation One (ASN.1), has been selected as the method for representing the data content of BACnet services. Clause 21 contains an ASN.1 definition for each service defined by this standard. ASN.1 provides an abstract syntax. However, the exact bit-by-bit layout of an APDU may have several forms, depending on the encoding rules that are selected.

Within the Open Systems Interconnection model, the encoding rules to be used are chosen by the presentation layer through a process of negotiation. This negotiation is used by cooperating systems to determine not only the basic encoding rules, of which ISO 8825, Specification of Basic Encoding Rules for ASN.1, is an example, but also whether or not the APDU is to be subjected to other manipulations such as data compression, encryption, or character code conversion.

Because BACnet's collapsed OSI architecture does not incorporate any presentation layer functionality, APDU encoding must be defined and agreed to by communicating devices in advance. BACnet's encoding rules have been designed to take into account the requirements of building automation and control systems for simplicity and compactness. As a result, they differ, in some respects, from ISO 8825 while still permitting the encoding of BACnet APDUs that have been represented using ASN.1. This means that BACnet services and procedures could be used in their entirety in a future OSI-compliant network by adding the presentation layer capability to negotiate either the encoding rules contained in this standard or any other encoding rules that might later be available.

The encoding of ASN.1 specified in ISO 8825 is intended to apply uniformly to all data elements in a PDU. Each data element is represented by three components: (1) identifier octets, (2) length octets, and (3) contents octets. The explicit identification of each data element allows parsers to be developed that can decode any PDU without prior knowledge of its format or semantic content. The alternative is to implicitly identify each data element, generally by mutual agreement as to its data format and location within the PDU. The former approach tends to result in greater generality at the expense of greater overhead; the latter approach tends to reduce overhead while limiting future extensibility.

The approach taken in BACnet is a compromise. The fixed portion of each APDU containing protocol control information is encoded implicitly and is described in Clause 20.1. The variable portion of each APDU containing service-specific information is encoded explicitly and is described in Clause 20.2. The resulting scheme significantly reduces overhead while preserving the possibility of easily adding new services in the future.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.1 Encoding the Fixed Part of BACnet APDUs

BACnet APDUs consist of protocol control information and, possibly, user data.

"Protocol control information" (PCI) comprises data required for the operation of the application layer protocol, including the type of APDU, information to match service requests and service responses, and information to carry out the reassembly of segmented messages. This information is contained in the "header," or fixed part, of the APDU.

"User data" comprises information specific to individual service requests or responses. This portion of the APDU will be referred to as the 'variable part' of the APDU.

Because every APDU contains PCI fields, BACnet encodes the PCI without the use of tags or length information even though the ASN.1 might indicate the presence of tags in the syntactical descriptions of the APDUs. Tags are used to encode the variable-content user data as specified in Clause 20.2. This selective use of tags results in a considerable reduction in overhead.

The remainder of this clause lays out the format of each APDU type.

20.1.1 Encoding the BACnetPDU CHOICE Tag

All BACnet messages are defined by an ASN.1 production called the BACnetPDU. See Clause 21. BACnetPDU is a choice of one of eight BACnet APDU types. For all BACnet APDUs, this choice shall be encoded as a four-bit binary number in the bits 4 through 7 of the first octet of the APDU header, with bit 7 being the most significant bit. These bits indicate the value of the tag (0 - 7), which represents the APDU type choice. This encoding is illustrated in the examples below for each APDU type.

20.1.2 BACnet-Confirmed-Request-PDU

The BACnet-Confirmed-Request-PDU is used to convey the information contained in confirmed service request primitives.

```
BACnet-Confirmed-Request-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 0 for this PDU type
        segmented-message               [1] BOOLEAN,
        more-follows                    [2] BOOLEAN,
        segmented-response-accepted     [3] BOOLEAN,
        reserved                        [4] Unsigned (0..3), -- shall be set to zero
        max-segments-accepted           [5] Unsigned (0..7), -- as per Clause 20.1.2.4
        max-apdu-length-accepted        [6] Unsigned (0..15), -- as per Clause 20.1.2.5
        invoke-id                       [7] Unsigned (0..255),
        sequence-number                 [8] Unsigned (0..255) OPTIONAL, -- only if segmented msg
        proposed-window-size            [9] Unsigned (1..127) OPTIONAL, -- only if segmented msg
        service-choice                  [10] BACnetConfirmedServiceChoice,
        service-request                 [11] BACnet-Confirmed-Service-Request
-- Context specific tags 0..11 are NOT used in header encoding
        }
```

The parameters of the BACnet-Confirmed-Request-PDU have the following meanings.

20.1.2.1 segmented-message

This parameter indicates whether or not the confirmed service request is entirely, or only partially, contained in the present PDU. If the request is present in its entirety, the value of the 'segmented-message' parameter shall be FALSE. If the present PDU contains only a segment of the request, this parameter shall be TRUE.

20.1.2.2 more-follows

This parameter is only meaningful if the 'segmented-message' parameter is TRUE. If 'segmented-message' is TRUE, then the 'more-follows' parameter shall be TRUE for all segments comprising the confirmed service request except for the last and shall be FALSE for the final segment. If 'segmented-message' is FALSE, then 'more-follows' shall be set FALSE by the encoder and shall be ignored by the decoder.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.1.2.3 segmented-response-accepted

This parameter shall be TRUE if the device issuing the confirmed request will accept a segmented complex acknowledgment as a response. It shall be FALSE otherwise. This parameter is included in the confirmed request so that the responding device may determine how to convey its response.

20.1.2.4 max-segments-accepted

This parameter specifies the maximum number of segments that the device will accept. This parameter is included in the confirmed request so that the responding device may determine how to convey its response. The parameter shall be encoded as follows:

B'000'	Unspecified number of segments accepted
B'001'	2 segments accepted
B'010'	4 segments accepted
B'011'	8 segments accepted
B'100'	16 segments accepted
B'101'	32 segments accepted
B'110'	64 segments accepted
B'111'	Greater than 64 segments accepted

20.1.2.5 max-apdu-length-accepted

This parameter specifies the maximum size of a single APDU that the issuing device will accept. This parameter is included in the confirmed request so that the responding device may determine how to convey its response. The exact maximum APDU length accepted by the issuing device may be larger than indicated in this parameter and is indicated in the Max_APDU_Length_Accepted property of the Device object of the issuing device. See Clause 12.11.18 and Clause 5.2.1.2.
The parameter shall be encoded as follows:

B'0000'	Up to MinimumMessageSize (50 octets)
B'0001'	Up to 128 octets
B'0010'	Up to 206 octets (fits in a LonTalk frame)
B'0011'	Up to 480 octets (fits in an ARCNET frame)
B'0100'	Up to 1024 octets
B'0101'	Up to 1476 octets (fits in a 1497 octet NPDU in one Ethernet frame), or larger than 1476 octets
B'0110'	reserved by ASHRAE
B'0111'	reserved by ASHRAE
B'1000'	reserved by ASHRAE
B'1001'	reserved by ASHRAE
B'1010'	reserved by ASHRAE
B'1011'	reserved by ASHRAE
B'1100'	reserved by ASHRAE
B'1101'	reserved by ASHRAE
B'1110'	reserved by ASHRAE
B'1111'	reserved by ASHRAE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

20.1.2.6 invoke-id

This parameter shall be an integer in the range 0 - 255 assigned by the service requester. It shall be used to associate the response to a confirmed service request with the original request. In the absence of any error, the 'invoke-id' shall be returned by the service provider in a BACnet-SimpleACK-PDU or a BACnet-ComplexACK-PDU. In the event of an error condition, the 'invoke-id' shall be returned by the service provider in a BACnet-Error-PDU, BACnet-Reject-PDU, or BACnet-Abort-PDU as appropriate.

The 'invoke-id' shall be generated by the device issuing the service request. It shall be unique for all outstanding confirmed request APDUs generated by the device. The same 'invoke-id' shall be used for all segments of a segmented service request. Once an 'invoke-id' has been assigned to an APDU, it shall be maintained within the device until either a response APDU is received with the same 'invoke-id' or a no response timer expires (see Clause 5.3). In either case, the 'invoke-id' value shall then be released for reassignment. The algorithm used to pick a value out of the set of unused

© ISO 2022 – All rights reserved

values is a local matter. The storage mechanism for maintaining the used 'invoke-id' values within the requesting and responding devices is also a local matter. The requesting device may use a single 'invoke-id' space for all its confirmed APDUs or multiple 'invoke-id' spaces (one per destination device address) as desired. Since the 'invoke-id' values are only source-device-unique, the responding device shall maintain the 'invoke-id' as well as the requesting device address until a response has been sent. The responding device may discard the 'invoke-id' information after a response has been sent.

20.1.2.7 sequence-number

This optional parameter is only present if the 'segmented-message' parameter is TRUE. In this case, the 'sequence-number' shall be a sequentially incremented unsigned integer, modulo 256, which identifies each segment of a segmented request. The value of the received 'sequence-number' is used by the responder to acknowledge the receipt of one or more segments of a segmented request. The 'sequence-number' of the first segment of a segmented request shall be zero.

20.1.2.8 proposed-window-size

This optional parameter is only present if the 'segmented-message' parameter is TRUE. In this case, the 'proposed-window-size' parameter shall specify as an unsigned binary integer the maximum number of message segments containing 'invoke-id' the sender is able or willing to send before waiting for a segment acknowledgment PDU (see Clauses 5.2 and 5.3). The value of the 'proposed-window-size' shall be in the range 1 - 127.

20.1.2.9 service-choice

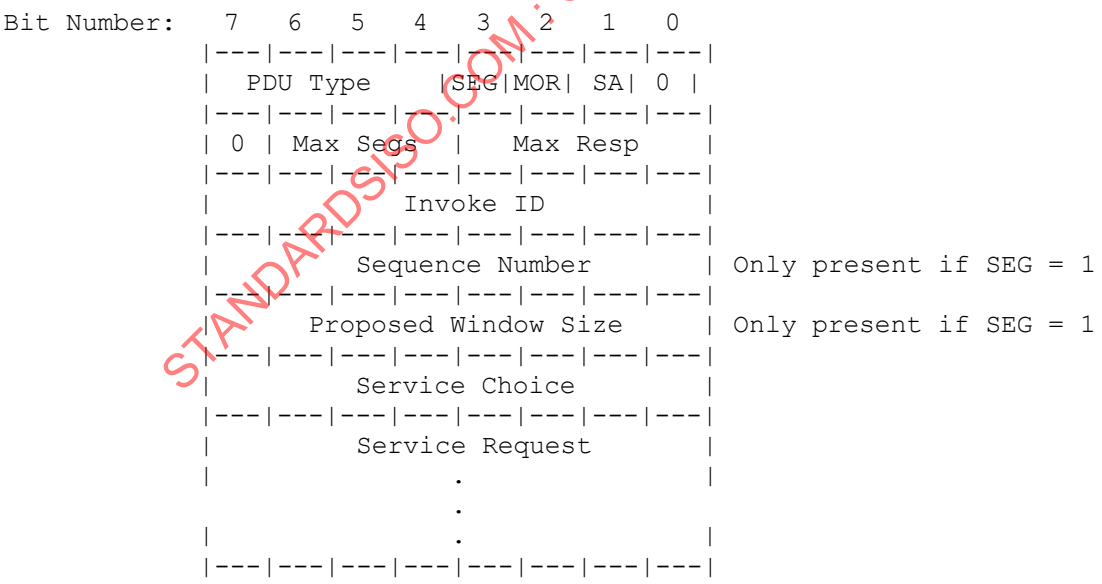
This parameter shall contain the value of the BACnetConfirmedServiceChoice. See Clause 21.

20.1.2.10 service-request

This parameter shall contain the parameters of the specific service that is being requested, encoded according to the rules of Clause 20.2. These parameters are defined in the individual service descriptions in this standard and are represented in Clause 21 in accordance with the rules of ASN.1.

20.1.2.11 Format of the BACnet-Confirmed-Request-PDU

The format of the BACnet-Confirmed-Request-PDU is:



STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The PDU fields have the following values:

PDU Type =	0 (BACnet-Confirmed-Service-Request-PDU)
SEG =	0 (Unsegmented Request)
	1 (Segmented Request)
MOR =	0 (No More Segments Follow)
	1 (More Segments Follow)
SA =	0 (Segmented Response not accepted)
	1 (Segmented Response accepted)
Max Segs =	(0..7) (Number of response segments accepted per Clause 20.1.2.4)
Max Resp =	(0..15) (Size of Maximum APDU accepted per Clause 20.1.2.5)
Invoke ID =	(0..255)
Sequence Number =	(0..255) Only present if SEG = 1
Proposed Window Size =	(1..127) Only present if SEG = 1
Service Choice =	BACnetConfirmedServiceChoice
Service Request =	Variable Encoding per Clause 20.2.

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.3 BACnet-Unconfirmed-Request-PDU

The BACnet-Unconfirmed-Request-PDU is used to convey the information contained in unconfirmed service request primitives.

```
BACnet-Unconfirmed-Request-PDU ::= SEQUENCE {
        pdu-type                [0] Unsigned (0..15), -- 1 for this PDU type
        reserved                [1] Unsigned (0..15), -- shall be set to zero
        service-choice          [2] BACnetUnconfirmedServiceChoice,
        service-request         [3] BACnet-Unconfirmed-Service-Request
-- Context specific tags 0..3 are NOT used in header encoding
        }
```

The parameters of the BACnet-Unconfirmed-Request-PDU have the following meanings.

20.1.3.1 service-choice

This parameter shall contain the value of the BACnetUnconfirmedServiceChoice. See Clause 21.

20.1.3.2 service-request

This parameter shall contain the parameters of the specific service that is being requested, encoded according to the rules of Clause 20.2. These parameters are defined in the individual service descriptions in this standard and are represented in Clause 21 in accordance with the rules of ASN.1.

20.1.3.3 Format of the BACnet-Unconfirmed-Request-PDU

The format of the BACnet-Unconfirmed-Request-PDU is:

```
Bit Number:  7    6    5    4    3    2    1    0
           |---|---|---|---|---|---|---|---|
           |   PDU Type      | 0 | 0 | 0 | 0 |
           |---|---|---|---|---|---|---|---|
           |          Service Choice         |
           |---|---|---|---|---|---|---|---|
           |          Service Request        |
           |                .                |
                            .
           |                .                |
           |---|---|---|---|---|---|---|---|
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The PDU fields have the following values:

PDU Type = 1 (BACnet-Unconfirmed-Service-Request-PDU)
Service Choice = BACnetUnconfirmedServiceChoice
Service Request = Variable Encoding per Clause 20.2.

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.4 BACnet-SimpleACK-PDU

The BACnet-SimpleACK-PDU is used to convey the information contained in a service response primitive ('Result(+)') that contains no other information except that the service request was successfully carried out.

```
BACnet-SimpleACK-PDU ::= SEQUENCE {
        pdu-type                [0] Unsigned (0..15), -- 2 for this PDU type
        reserved                [1] Unsigned (0..15), -- shall be set to zero
        original-invoke-id      [2] Unsigned (0..255),
        service-ack-choice      [3] BACnetConfirmedServiceChoice
-- Context specific tags 0..3 are NOT used in header encoding
        }
```

The parameters of the BACnet-SimpleACK-PDU have the following meanings.

20.1.4.1 original-invoke-id

This parameter shall be the 'invoke-id' contained in the confirmed service request being acknowledged.

20.1.4.2 service-ack-choice

This parameter shall contain the value of the BACnetConfirmedServiceChoice corresponding to the service contained in the previous BACnet-Confirmed-Service-Request that has resulted in this acknowledgment. See Clause 21.

20.1.4.3 Format of the BACnet-SimpleACK-PDU

The format of the BACnet-SimpleACK-PDU is:

```
Bit Number:    7   6   5   4    3  2   1   0
             |---|---|---|---|---|---|---|---|
             |   PDU-Type      | 0 | 0 | 0 | 0 |
             |---|---|---|---|---|---|---|---|
             |       Original Invoke ID      |
             |---|---|---|---|---|---|---|---|
             |      Service Ack Choice       |
             |---|---|---|---|---|---|---|---|
```

Note that this APDU is always three octets long.

The PDU fields have the following values:

PDU Type = 2 (BACnet-SimpleACK-PDU)
Original Invoke ID = (0..255)
Service ACK Choice = BACnetConfirmedServiceChoice

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.5 BACnet-ComplexACK-PDU

The BACnet-ComplexACK-PDU is used to convey the information contained in a service response primitive ('Result(+)') that contains information in addition to the fact that the service request was successfully carried out.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnet-ComplexACK-PDU ::= SEQUENCE {
        pdu-type                    [0] Unsigned (0..15), -- 3 for this PDU type
        segmented-message           [1] BOOLEAN,
        more-follows                [2] BOOLEAN,
        reserved                    [3] Unsigned (0..3), -- shall be set to zero
        original-invoke-id          [4] Unsigned (0..255),
        sequence-number             [5] Unsigned (0..255) OPTIONAL, --only if segment
        proposed-window-size        [6] Unsigned (1..127) OPTIONAL, -- only if segment
        service-ack-choice          [7] BACnetConfirmedServiceChoice,
        service-ack                 [8] BACnet-Confirmed-Service-ACK
-- Context specific tags 0..8 are NOT used in header encoding
        }
```

The parameters of the BACnet-ComplexACK-PDU have the following meanings.

20.1.5.1 segmented-message

This parameter indicates whether or not the confirmed service response is entirely, or only partially, contained in the present APDU. If the response is present in its entirety, the 'segmented-message' parameter shall be FALSE. If the present APDU contains only a segment of the response, this parameter shall be TRUE.

20.1.5.2 more-follows

This parameter is only meaningful if the 'segmented-message' parameter is TRUE. If 'segmented-message' is TRUE, then the 'more-follows' parameter shall be TRUE for all segments comprising the confirmed service response except for the last and shall be FALSE for the final segment. If 'segmented-message' is FALSE, then 'more-follows' shall be set FALSE by the encoder and shall be ignored by the decoder.

20.1.5.3 original-invoke-id

This parameter shall be the 'invoke-id' contained in the confirmed service request being acknowledged. The same 'original-invoke-id' shall be used for all segments of a segmented acknowledgment.

20.1.5.4 sequence-number

This optional parameter is only present if the 'segmented-message' parameter is TRUE. In this case, the 'sequence-number' shall be a sequentially incremented unsigned integer, modulo 256, which identifies each segment of a segmented response. The value of the received 'sequence-number' is used by the original requester to acknowledge the receipt of one or more segments of a segmented response. The sequence-number of the first segment of a segmented response shall be zero.

20.1.5.5 proposed-window-size

This optional parameter is only present if the 'segmented-message' parameter is TRUE. In this case, the 'proposed-window-size' parameter shall specify as an unsigned binary integer the maximum number of message segments containing 'original-invoke-id' the sender is able or willing to send before waiting for a segment acknowledgment APDU (see Clauses 5.2 and 5.3). The value of the 'proposed-window-size' shall be in the range 1 - 127.

20.1.5.6 service-ack-choice

This parameter shall contain the value of the BACnetConfirmedServiceChoice corresponding to the service contained in the previous BACnet-Confirmed-Service-Request that has resulted in this acknowledgment. See Clause 21.

20.1.5.7 service-ack

This parameter shall contain the parameters of the specific service acknowledgment that is being encoded according to the rules of Clause 20.2. These parameters are defined in the individual service descriptions in this standard and are represented in Clause 21 in accordance with the rules of ASN.1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.1.5.8 Format of the BACnet-ComplexACK-PDU

The format of the BACnet-ComplexACK-PDU is:

```
Bit Number:    7    6    5    4    3    2    1    0
             |---|---|---|---|---|---|---|---|
             |  PDU Type      |SEG|MOR| 0 | 0 |
             |---|---|---|---|---|---|---|---|
             |        Original Invoke ID      |
             |---|---|---|---|---|---|---|---|
             |          Sequence Number       | Only present if SEG = 1
             |---|---|---|---|---|---|---|---|
             |        Proposed Window Size    | Only present if SEG = 1
             |---|---|---|---|---|---|---|---|
             |         Service ACK Choice     |
             |---|---|---|---|---|---|---|---|
             |            Service ACK         |
             |                .               |
                              .
             |                .               |
             |---|---|---|---|---|---|---|---|
```

The PDU fields have the following values:

PDU Type =	3 (BACnet-ComplexACK-PDU)
SEG =	0 (Unsegmented Response) 1 (Segmented Response)
MOR =	0 (No More Segments Follow) 1 (More Segments Follow)
Original Invoke ID =	(0..255)
Sequence Number =	(0..255) Only present if SEG = 1
Proposed Window Size =	(1..127) Only present if SEG = 1
Service ACK Choice =	BACnetConfirmedServiceChoice
Service ACK =	Variable Encoding per Clause 20.2.

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.6 BACnet-SegmentACK-PDU

The BACnet-SegmentACK-PDU is used to acknowledge the receipt of one or more APDUs containing portions of a segmented message. It may also request the next segment or segments of the segmented message.

```
BACnet-SegmentACK-PDU ::= SEQUENCE {
        pdu-type                [0] Unsigned (0..15), -- 4 for this PDU type
        reserved                [1] Unsigned (0..3), -- shall be set to zero
        negative-ack            [2] BOOLEAN,
        server                  [3] BOOLEAN,
        original-invoke-id      [4] Unsigned (0..255),
        sequence-number         [5] Unsigned (0..255),
        actual-window-size      [6] Unsigned (1..127)
-- Context specific tags 0..6 are NOT used in header encoding
        }
```

The parameters of the BACnet-SegmentACK-PDU have the following meanings.

20.1.6.1 negative-ack

This parameter shall be TRUE if the Segment-ACK PDU is being sent to indicate a segment received out of order. Otherwise, it shall be FALSE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.1.6.2 server

This parameter shall be TRUE when the SegmentACK PDU is sent by a server, that is, when the SegmentACK PDU is in acknowledgment of a segment or segments of a Confirmed-Request PDU.

This parameter shall be FALSE when the SegmentACK PDU is sent by a client, that is, when the SegmentACK PDU is in acknowledgment of a segment or segments of a ComplexACK PDU.

20.1.6.3 original-invoke-id

This parameter shall be the 'invoke-id' contained in the segment being acknowledged.

20.1.6.4 sequence-number

This parameter shall contain the 'sequence-number' of a previously received message segment. It is used to acknowledge the receipt of that message segment and all earlier segments of the message.

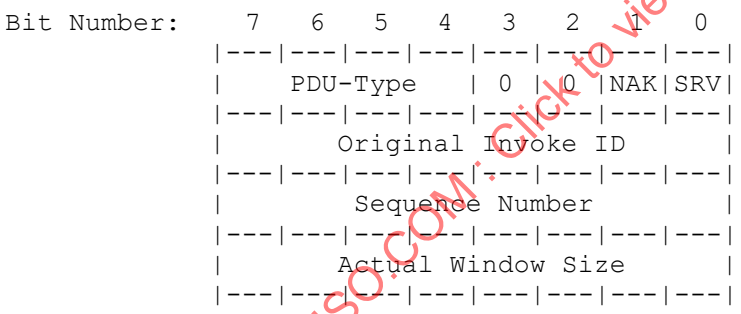
If the 'more-follows' parameter of the received message segment is TRUE, then the 'sequence-number' also requests continuation of the segmented message beginning with the segment whose 'sequence-number' is one plus the value of this parameter, modulo 256.

20.1.6.5 actual-window-size

This parameter shall specify as an unsigned binary integer the number of message segments containing 'original-invoke-id' the sender will accept before sending another SegmentACK. See Clause 5.3 for additional details. The value of the 'actual-window-size' shall be in the range 1 - 127.

20.1.6.6 Format of the BACnet-SegmentACK-PDU

The format of the BACnet-SegmentACK-PDU is:



Note that this PDU is always four octets long.

The PDU fields have the following values:

PDU Type =	4 (BACnet-SegmentACK-PDU)
NAK =	0 (Normal Acknowledgment)
	1 (Negative Acknowledgment, Segment Out of Order)
SRV =	0 (Sent by Client)
	1 (Sent by Server)
Original Invoke ID =	(0..255)
Sequence Number =	(0..255)
Actual Window Size =	(1..127)

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.7 BACnet-Error-PDU

The BACnet-Error-PDU is used to convey the information contained in a service response primitive ('Result(-)') that indicates the reason why a previous confirmed service request failed either in its entirety or only partially.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnet-Error-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 5 for this PDU type
        reserved                        [1] Unsigned (0..15), -- shall be set to zero
        original-invoke-id              [2] Unsigned (0..255),
        error-choice                    [3] BACnetConfirmedServiceChoice,
        error                           [4] BACnet-Error
-- Context specific tags 0..4 are NOT used in header encoding
        }
```

The parameters of the BACnet-Error-PDU have the following meanings.

20.1.7.1 original-invoke-id

This parameter shall be the 'invoke-id' contained in the confirmed service request to which the error is a response.

20.1.7.2 error-choice

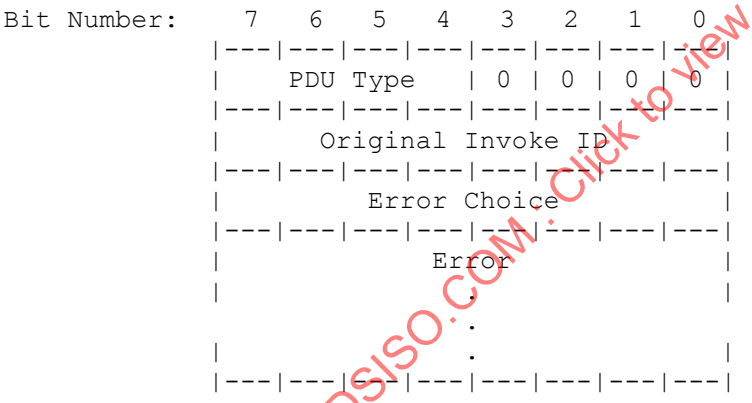
This parameter, of type BACnetConfirmedServiceChoice, shall contain the tag value of the BACnet-Error choice. See Clause 21.

20.1.7.3 error

This parameter, of type BACnet-Error, indicates the reason the indicated service request could not be carried out. This parameter shall be encoded according to the rules of Clause 20.2.

20.1.7.4 Format of the BACnet-Error-PDU

The format of the BACnet-Error-PDU is:



The PDU fields have the following values:

PDU Type =	5 (BACnet-Error-PDU)
Original Invoke ID =	(0..255)
Error Choice =	BACnetConfirmedServiceChoice
Error =	Variable Encoding per Clause 20.2.

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.8 BACnet-Reject-PDU

The BACnet-Reject-PDU is used to reject a received confirmed request APDU based on syntactical flaws or other protocol errors that prevent the APDU from being interpreted or the requested service from being provided. Only confirmed request APDUs may be rejected (see Clause 18.9). A BACnet-Reject-PDU shall be sent only before the execution of the service, such as during the interval after a syntax check is performed on the request but before the service procedure is executed. Such a syntax check may occur as segments are received and thus may result in a BACnet-Reject-PDU being returned before the complete request has been received.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

There are error conditions where a valid reject-reason and an equally valid error code exist that can be used to describe the condition. In such cases it is a local matter whether or not a BACnet-Reject-PDU is used to convey the error, with the exception that a BACnet-Reject-PDU shall not be returned if service execution has commenced and the execution has resulted in a standard network visible change in the device's state. For example, a WritePropertyMultiple-Request shall not be rejected if at least one of the write operations has already been applied. In such cases an error code shall be used.

```
BACnet-Reject-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 6 for this PDU type
        reserved                        [1] Unsigned (0..15), -- shall be set to zero
        original-invoke-id              [2] Unsigned (0..255),
        reject reason                   [3] BACnetRejectReason
-- Context specific tags 0..3 are NOT used in header encoding
        }
```

The parameters of the BACnet-Reject-PDU have the following meanings.

20.1.8.1 original-invoke-id

This parameter shall be the 'invoke-id' of the PDU being rejected.

20.1.8.2 reject-reason

This parameter, of type BACnetRejectReason, contains the reason the PDU with the indicated 'invoke-id' is being rejected.

20.1.8.3 Format of the BACnet-Reject-PDU

The format of the BACnet-Reject-PDU is:

```
Bit Number:  7   6   5   4   3   2   1   0
            |---|---|---|---|---|---|---|---|
            |    PDU-Type   | 0 | 0 | 0 | 0 |
            |---|---|---|---|---|---|---|---|
            |       Original Invoke ID      |
            |---|---|---|---|---|---|---|---|
            |          Reject Reason        |
            |---|---|---|---|---|---|---|---|
```

Note that this PDU is always three octets long.

The PDU fields have the following values:

```
PDU Type =                      6 (BACnet-Reject-PDU)
Original Invoke ID =            (0..255)
Reject Reason =                 One octet containing the reject reason enumeration
```

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

20.1.9 BACnet-Abort-PDU

The BACnet-Abort-PDU is used to terminate a transaction between two peers.

```
BACnet-Abort-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 7 for this PDU type
        reserved                        [1] Unsigned (0..7), -- shall be set to zero
        server                          [2] BOOLEAN,
        original-invoke-id              [3] Unsigned (0..255),
        abort-reason                    [4] BACnetAbortReason
-- Context specific tags 0..4 are NOT used in header encoding
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The parameters of the BACnet-Abort-PDU have the following meanings.

20.1.9.1 server

This parameter shall be TRUE when the Abort PDU is sent by a server. This parameter shall be FALSE when the Abort PDU is sent by a client.

20.1.9.2 original-invoke-id

This parameter shall be the 'invoke-id' of the transaction being aborted.

20.1.9.3 abort-reason

This parameter, of type BACnetAbortReason, contains the reason the transaction with the indicated invoke ID is being aborted.

20.1.9.4 Format of the BACnet-Abort-PDU

The format of the BACnet-Abort-PDU is:

```
Bit Number:    7    6    5    4    3    2    1    0
             |---|---|---|---|---|---|---|---|
             |     PDU Type    | 0 | 0 | 0 |SRV|
             |---|---|---|---|---|---|---|---|
             |       Original Invoke ID        |
             |---|---|---|---|---|---|---|---|
             |          Abort Reason           |
             |---|---|---|---|---|---|---|---|
```

Note that this PDU is always three octets long.

The PDU fields have the following values:

PDU Type =	7 (BACnet-Abort-PDU)
SRV =	0 (Sent by Client) 1 (Sent by Server)
Original Invoke ID =	(0..255)

Bits shown in the diagram as '0' shall be set to zero. These bits are currently unused and are reserved by ASHRAE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.2 Encoding the Variable Part of BACnet APDUs

The encoding of the header portions of BACnet APDUs has been specified in Clause 20.1. This clause describes the encoding procedures for the variable portion of BACnet APDUs referred to hereafter as "service parameters." These parameters are of types BACnet-Confirmed-Service-Request, BACnet-Unconfirmed-Service-Request, BACnet-Confirmed-Service-ACK, and BACnet-Error. Each parameter is unambiguously defined by means of ASN.1 productions in Clause 21.

All data elements in service parameters are identified by constructs known as "tags." Each tag refers to a unique parameter or subparameter.

BACnet encoding uses two classes of tag. The first identifies fundamental datatypes used or defined in this standard, such as BOOLEANs, Unsigneds, CharacterStrings, Date, Time, or BACnetObjectIdentifiers. Where a datatype appears in upper case, its semantics are identical to the corresponding ASN.1 universal datatype of the same name, as indicated in Clause 21.

Such tags are called "application" tags, and the values of the tags are specified in Clause 20.2.1.4.

The second class of tag is used to identify data elements whose datatype may be inferred from the context in which they appear. These tags are called "context specific" tags.

ASN.1 defines two other classes of tags, "universal" and "private." The encoding scheme used by BACnet does not allow, nor do any BACnet ASN.1 productions require, the use of these classes of tags.

In some instances, the datatype of a parameter cannot be deduced from the context in which it appears. In such cases, both a context specific and one or more application tags are required. These cases are indicated in the ASN.1 productions by service parameters whose datatypes are indicated by the keywords ABSTRACT-SYNTAX.&TYPE (ANY), CHOICE, SEQUENCE, or SEQUENCE OF.

The clauses that follow show how each tagged element is identified, its length specified, and its value encoded.

20.2.1 General Rules For Encoding BACnet Tags

BACnet tags are encoded in an initial octet and zero or more conditional subsequent octets. The initial octet is defined as follows:

```
Bit Number:    7       6     5      4      3      2      1      0
             |-----|-----|-----|-----|-----|-----|-----|-----|
             |        Tag Number       |Class|Length/Value/Type|
             |-----|-----|-----|-----|-----|-----|-----|-----|
```

where Tag Number = the tag number within the class
Class = the class of tag (application or context specific)
Length/Value/Type = whether the data following the tag is primitive or constructed and specifies the length or value of primitive data.

20.2.1.1 Class

The Class bit shall be zero for application tags. The Class bit shall be one for context specific tags.

20.2.1.2 Tag Number

Tag numbers ranging from zero to 14 (inclusive) shall be encoded in the Tag Number field of the initial octet as a four bit binary integer with bit 7 the most significant bit.

Tag numbers ranging from 15 to 254 (inclusive) shall be encoded by setting the Tag Number field of the initial octet to B'1111' and following the initial tag octet by an octet containing the tag number represented as an eight-bit binary integer with bit 7 the most significant bit.

The encoding does not allow, nor does BACnet require, tag numbers larger than 254. The value B'11111111' of the subsequent octet is reserved by ASHRAE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

20.2.1.3 Length/Value/Type

The content of the length/value/type field of the initial octet distinguishes between primitive and constructed encodings and specifies the length or value of primitive data. A primitive encoding is one in which the data do not contain other tagged encodings. A constructed encoding is one in which the data do contain other tagged encodings.

20.2.1.3.1 Primitive Data

If the data being encoded are application class BOOLEAN data, then the Boolean value shall be encoded by setting the length/value/type field of the initial octet to B'000' if the Boolean value is FALSE or B'001' if the Boolean value is TRUE. In this case, the length/value/type field shall be interpreted as a value.

If the data being encoded are primitive (that is, not constructed) and not application class BOOLEAN data, then the value of the data shall be encoded according to Clause 20.2.2 through 20.2.14, and the length/value/type field of the initial tag octet shall specify the length of the primitive data in octets as follows:

Data length in octets ranging from zero to four (inclusive) shall be encoded in the length/value/type field of the initial octet as a three-bit binary integer with bit 2 the most significant bit.

Data length in octets ranging from 5 to 253 (inclusive) shall be encoded by setting the length/value/type field of the initial octet to B'101' and following the initial tag octet or, if the Tag Number has been extended, following the Tag Number extension octet by an octet containing the data length represented as an eight-bit binary integer with bit 7 the most significant bit.

Data length in octets ranging from 254 to 65535 (inclusive) shall be encoded by setting the length/value/type field of the initial octet to B'101' and following the initial tag octet or, if the Tag Number has been extended, following the Tag Number extension octet by an octet containing D'254' and two additional octets whose value contains the data length represented as a 16-bit binary integer with the most significant octet first.

Data length in octets ranging from 65536 to 2^{32}-1 (inclusive) shall be encoded by setting the length/value/type field of the initial octet to B'101' and following the initial tag octet or, if the Tag Number has been extended, following the Tag Number extension octet by an octet containing D'255' and four additional octets whose value contains the data length represented as a 32-bit binary integer with the most significant octet first.

Data lengths larger than 2^{32}-1 are not encodable using primitive tags.

Note that with the exception of 8-octet IEEE-754 double precision floating point values and certain bit, character, and octet strings, the length of BACnet application-tagged primitives will fit in the tag octet without extension.

20.2.1.3.2 Constructed Data

If the production being encoded contains tagged elements, then the encoding is called "constructed" and shall consist of

(a) an "opening" tag whose Tag Number field shall contain the value of the tag number, whose Class field shall indicate "context specific," and whose length/value/type field shall have the value B'110';
(b) the complete encoding, with tags, of the zero, one, or more elements that comprise the data;
(c) a "closing" tag, whose Class and Tag Number fields shall contain the same values as the "opening" tag and whose length/value/type field shall have the value B'111'.

In this case, the length/value/type fields of the "opening" and "closing" tags shall be interpreted as types.

Note that a contained tagged element may itself be a constructed element. This recursion does not result in ambiguous encoding, as each "opening" tag is required to have a corresponding "closing" tag that will be contained within any outer "opening" and "closing" tags.

20.2.1.4 Application Tags

The Tag Number field of an encoded BACnet application tag shall specify the application datatype as follows:

Tag Number: 0 = Null
1 = Boolean
2 = Unsigned Integer

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

3 = Signed Integer (2's complement notation)
4 = Real (ANSI/IEEE-754 floating point)
5 = Double (ANSI/IEEE-754 double precision floating point)
6 = Octet String
7 = Character String
8 = Bit String
9 = Enumerated
10 = Date
11 = Time
12 = BACnetObjectIdentifier
13, 14, 15 = Reserved for ASHRAE

Note that all currently defined BACnet Application datatypes are primitively encoded.

20.2.1.5 Context-Specific Tags

The Tag Number field of an encoded BACnet context-specific tag shall contain the value of the context-specific tag number.

The data delimited by a context-specific tag may be either primitive or constructed.

20.2.1.6 Summary of the Tagging Rules

While the tagged portion of a BACnet PDU cannot be interpreted without knowledge of the context, the tagging rules described in Clause 20.2 result in a tagged stream that can be unambiguously parsed even without a priori knowledge of the context.

(a) The first octet in a stream shall be a tag, either context specific or application class.
(b) If a tag is application class, then the format and extent of its data are known according to the definitions in Clause 20.2. In particular, the data, if any, may be bypassed and the next tag in the stream found.
(c) If a tag is context specific and primitive, then it contains primitive (untagged) data of some type. The length/value/type field of the tag specifies the length of this data. Thus, while the datatype and format of the data may be unknown, its length is known exactly. This allows the data, if any, to be bypassed and the next tag in the stream found.
(d) If a tag is constructed (length/value/type = 6), then it is the opening tag of a pair. Following this tag shall be a sequence of zero or more tagged elements, followed by the closing tag of the pair with length/value/type = 7. The tagged stream between opening and closing tags may be parsed according to these same four rules via a process of recursive descent until only primitive tags are encountered or until no tags are encountered.

20.2.2 Encoding of a Null Value

The encoding of a Null value shall be primitive, with no contents octet.

Example: Application-tagged null value

ASN.1 =	NULL
Application Tag =	Null (Tag Number = 0)
Encoded Tag =	X'00'

20.2.3 Encoding of a Boolean Value

Application-tagged Boolean values shall be encoded within a single octet by setting the length/value/type field to B'000' if the value to be encoded is FALSE or B'001' if the value to be encoded is TRUE.

Example: Application-tagged Boolean value

ASN.1 =	BOOLEAN
Value =	FALSE
Application Tag =	Boolean (Tag Number = 1)
Encoded Tag =	X'10'

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Context-tagged Boolean primitive data shall contain one contents octet. The value of this octet shall be B'00000000' if the value to be encoded is FALSE or B'00000001' if the value to be encoded is TRUE.

Example: Context-tagged Boolean value

ASN.1 =	[2] BOOLEAN
Value =	TRUE
Context Tag =	2
Encoded Tag =	X'29'
Encoded Data =	X'01'

NOTE: The Boolean datatype differs from the other datatypes in that the encoding of a context-tagged Boolean value is not the same as the encoding of an application-tagged Boolean value. This is done so that the application-tagged value may be encoded in a single octet, without a contents octet. While this same encoding could have been used for the context-tagged case, doing so would require that the context be known in order to distinguish between a length or a value in the length/value/type field. This was considered to be undesirable. See Clause 20.2.20.

20.2.4 Encoding of an Unsigned Integer Value

The encoding of an unsigned integer value shall be primitive, with at least one contents octet.

Unsigned integers shall be encoded in the contents octet(s) as binary numbers in the range 0 to $(2^{8*L} - 1)$ where L is the number of octets used to encode the value and L is at least one. Values encoded into more than one octet shall be conveyed with the most significant octet first. All unsigned integers shall be encoded in the smallest number of octets possible. That is, the first octet of any multi-octet encoded value shall not be X'00'.

Example: Application-tagged unsigned integer

ASN.1 =	Unsigned
Value =	72
Application Tag = Unsigned Integer (Tag Number = 2)	
Encoded Tag =	X'21'
Encoded Data =	X'48'

20.2.5 Encoding of a Signed Integer Value

The encoding of a signed integer value shall be primitive, with at least one contents octet.

Signed integers shall be encoded in the contents octet(s) as binary numbers using 2's complement notation in the range $-2^{(8*L-1)}$ to $(2^{(8*L-1)} - 1)$ where L is the number of octets used to encode the value and L is at least one. Values encoded into more than one octet shall be conveyed most significant octet first. All signed integers shall be encoded in the smallest number of octets possible. That is, the first octet of any multi-octet encoded value shall not be X'00' if the most significant bit (bit 7) of the second octet is 0, and the first octet shall not be X'FF' if the most significant bit of the second octet is 1.

Example: Application-tagged signed integer

ASN.1 =	INTEGER
Value =	72
Application Tag = Signed Integer (Tag Number = 3)	
Encoded Tag =	X'31'
Encoded Data =	X'48'

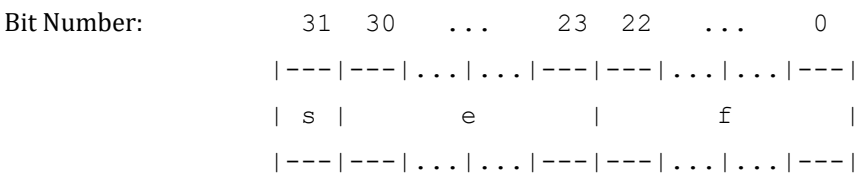
20.2.6 Encoding of a Real Number Value

The encoding of a real number value shall be primitive, with four contents octets. Real numbers shall be encoded using the method specified in ANSI/IEEE Standard 754-1985, "IEEE Standard for Binary Floating-Point Arithmetic." This standard should be consulted for details. The multi-octet value shall be conveyed with the most significant (sign and exponent) octet first.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

For the case of single precision real numbers, the encoding format is:



where the numbers indicate the field widths in bits. Non-zero values shall be represented by the equation $v = (-1)^s 2^{e-127}(1 \bullet f)$ where the symbol "•" signifies the binary point. Zero shall be indicated by setting s, e, and f to zero.

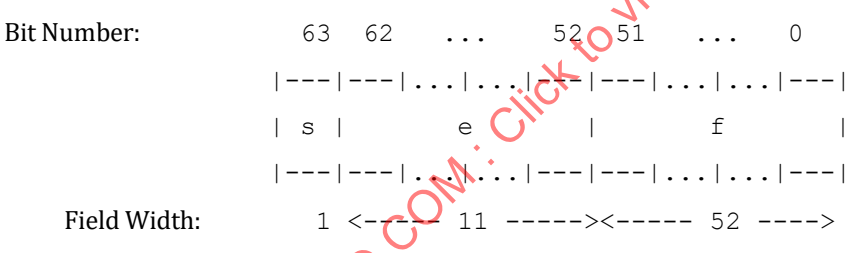
Example: Application-tagged single precision real

ASN.1 =	REAL
Value =	72.0
Application Tag = Real (Tag Number = 4)	
Encoded Tag =	X'44'
Encoded Data =	X'42900000'

20.2.7 Encoding of a Double Precision Real Number Value

The encoding of a double precision real number value shall be primitive with eight contents octets. Double precision real numbers shall be encoded using the method specified in ANSI/IEEE Standard 754-1985, "IEEE Standard for Binary Floating-Point Arithmetic." This standard should be consulted for details. The multi-octet value shall be conveyed most significant (sign and exponent) octet first.

For the case of double precision real numbers, the encoding format is:



where the numbers indicate the field widths in bits. Non-zero values shall be represented by the equation $v = (-1)^s 2^{e-1023}(1 \bullet f)$ where the symbol "•" signifies the binary point. Zero shall be indicated by setting s, e, and f to zero.

Example: Application-tagged double precision real

ASN.1 =	Double
Value =	72.0
Application Tag = Double (Tag Number = 5)	
Encoded Tag =	X'55'
Extended Length =	X'08'
Encoded Data =	X'4052000000000000'

ANSI/IEEE-754 Extended Precision format is not supported by BACnet.

20.2.8 Encoding of an Octet String Value

The encoding of an octet string value shall be primitive.

The encoding shall contain zero, one, or more contents octets equal in value to the octets in the data value, in the order in which they appear in the data value, and with the most significant bit of an octet of the data value aligned with the most significant bit of an octet of the contents octets.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Example: Application-tagged octet string
ASN.1 = OCTET STRING
Value = X'1234FF'
Application Tag = Octet String (Tag Number = 6)
Encoded Tag = X'63'
Encoded Data = X'1234FF'

20.2.9 Encoding of a Character String Value

The encoding of a character string value shall be primitive.

The encoding shall contain an initial contents octet, and zero, one, or more additional contents octets equal in value to the octets in the data value, in the order in which they appear in the data value, i.e. most significant octet first, and with the most significant bit of an octet of the data value aligned with the most significant bit of an octet of the contents octets.

The initial octet shall specify the character set with the following encoding:
X'00' ISO 10646 (UTF-8)
X'01' IBM™/Microsoft™ DBCS
X'02' JIS X 0208
X'03' ISO 10646 (UCS-4)
X'04' ISO 10646 (UCS-2)
X'05' ISO 8859-1

Other values of the initial octet are reserved by ASHRAE.
Example: Application-tagged character string
ASN.1 = CharacterString
Value = "This is a BACnet string!" (ISO 10646 UTF-8)
Application Tag = Character String (Tag Number = 7)
Encoded Tag = X'75'
Length Extension = X'19'
Character Set = X'00' (ISO 10646: UTF-8)
Encoded Data = X'546869732069732061204241436E657420737472696E6721'

Example: Application-tagged character string with non-ANSI character
ASN.1 = CharacterString
Value = "Français" (ISO 10646 UTF-8)
Application Tag = Character String (Tag Number = 7)
Encoded Tag = X'75'
Length Extension = X'0A'
Character Set = X'00' (ISO 10646: UTF-8)
Encoded Data = X'4672616EC3A7616973'

In the case of IBM/Microsoft DBCS (X'01'), the initial octet shall be followed by two additional octets whose value shall represent an unsigned integer, with the most significant octet first, that shall indicate the Code Page to be presumed for the characters that follow.

Example: Application-tagged character string (DBCS)
ASN.1 = CharacterString
Value = "This is a BACnet String!" (IBM/Microsoft DBCS, code page 850)
Application Tag = Character String (Tag Number = 7)
Encoded Tag = X'75'
Length Extension = X'1B'
Encoded Data = X'0103525468697320697320612042414361436E657420737472696E6721'

In the case of ISO 10646 UCS-2 (X'04') and UCS4 (X'03'), each character of the string shall be represented by two or four octets, respectively. The octet order for UCS-2 shall be Row-Cell. The octet order for UCS-4 shall be Group-Plane-Row-Cell.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Example: Application-tagged character string (UCS-2)

ASN.1 =	CharacterString
Value =	"This is a BACnet String!" (ISO 10646 UCS-2)
Application Tag = Character String (Tag Number = 7)	
Encoded Tag =	X'75'
Length Extension =	X'31'
Encoded Data =	X'040054006800690073002000690073002000610020004200410043006E006500740020007300740072006900 6E00670021'

20.2.10 Encoding of a Bit String Value

The encoding of a bit string value shall be primitive.

The contents octets for the primitive encoding shall contain an initial octet and zero or more subsequent octets containing the bit string. The initial octet shall encode, as an unsigned binary integer, the number of unused bits in the final subsequent octet. The number of unused bits shall be in the range zero to seven, inclusive.

Bit strings defined in this standard, e.g., the Status_Flags property, shall be encoded in the order of definition, with the first defined Boolean value in the most significant bit, i.e. bit 7, of the first subsequent octet. The bits in the bitstring shall be placed in bits 7 to 0 of the first subsequent octet, followed by bits 7 to 0 of the second subsequent octet, followed by bits 7 to 0 of each octet in turn, followed by as many bits as are needed of the final subsequent octet, commencing with bit 7. Undefined bits shall be zero.

If the bit string is empty, there shall be no subsequent octets, and the initial octet shall be zero.

Example: Application-tagged bit string

ASN.1 =	BIT STRING
Value =	B'10101'
Application Tag = Bit String (Tag Number = 8)	
Encoded Tag =	X'82'
Encoded Data =	X'03A8'

20.2.11 Encoding of an Enumerated Value

The encoding of an enumerated value shall be primitive, with at least one contents octet.

Enumerated values shall be encoded in the contents octet(s) as binary numbers in the range 0 to $(2^{8*L} - 1)$ where L is the number of octets used to encode the value and L is at least one. Values encoded into more than one octet shall be conveyed most significant octet first. All enumerated values shall be encoded in the smallest number of octets possible. That is, the first octet of any multi-octet encoded value shall not be X'00'.

Example: Application-tagged enumeration

ASN.1 =	BACnetObjectType
Value =	ANALOG-INPUT (0)
Application Tag = Enumerated (Tag Number = 9)	
Encoded Tag =	X '91'
Encoded Data =	X '00'

20.2.12 Encoding of a Date Value

The encoding of a date value shall be primitive, with four contents octets. Unless otherwise specified (e.g., UTC date), a date value generated by a device shall be a local date.

Date values shall be encoded in the contents octets as four binary integers. The first contents octet shall represent the year minus 1900; the second octet shall represent the month, with January = 1; the third octet shall represent the day of the month; and the fourth octet shall represent the day of the week, with Monday = 1. A value of X'FF' = D'255' in any of the four octets shall indicate that the corresponding value is unspecified and shall be considered a wildcard when matching dates. If all four octets = X'FF', the corresponding date may be interpreted as "any" or "don't care."

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Neither an unspecified date nor a date pattern shall be used in date values that convey actual dates, such as in a TimeSynchronization-Request.

The processing of a day of week received in a service that is in the range 1 to 7 and is inconsistent with the values in the other octets shall be a local matter.

A number of special values for the month and day octets have been defined. The following special values shall not be used when conveying an actual date value, such as the Local_Date property of the Device object or in a TimeSynchronization-Request. A value of 13 in the second octet shall indicate odd months. A value of 14 in the second octet shall indicate even months. A value of 32 in the third octet shall indicate the last day of the month. A value of 33 in the third octet shall indicate odd days of the month. A value of 34 in the third octet shall indicate even days of the month.

Example: Application-tagged specific date value

ASN.1 =	Date
Value =	January 24, 1991 (Day of week = Thursday)
Application Tag = Date	(Tag Number = 10)
Encoded Tag =	X'A4'
Encoded Data =	X'5B011804'

Example: Application-tagged date pattern value

ASN.1 =	Date
Value =	year = 1991, month is unspecified, day = 24, day of week is unspecified
Application Tag = Date	(Tag Number = 10)
Encoded Tag =	X'A4'
Encoded Data =	X'5BFF18FF'

20.2.13 Encoding of a Time Value

The encoding of a time value shall be primitive, with four contents octets. Unless otherwise specified (e.g., UTC time), a time value generated by a device shall be a local time.

Time values shall be encoded in the contents octets as four binary integers. The first contents octet shall represent the hour, in the 24-hour system (1 P.M. = D'13'); the second octet shall represent the minute of the hour; the third octet shall represent the second of the minute; and the fourth octet shall represent the fractional part of the second in hundredths of a second. A value of X'FF' = D'255' in any of the four octets shall indicate that the corresponding value is unspecified and shall be considered a wildcard when matching times. If all four octets = X'FF', the corresponding time may be interpreted as "any" or "don't care."

Neither an unspecified time nor a time pattern shall be used in time values that convey actual time, such as those presented by the Local_Time property of the Device object or in a TimeSynchronization-Request.

Example: Application-tagged specific time value

ASN.1 =	Time
Value =	17:35:45.17 (= 5:35:45.17 P.M.)
Application Tag = Time	(Tag Number = 11)
Encoded Tag =	X'B4'
Encoded Data =	X'11232D11'

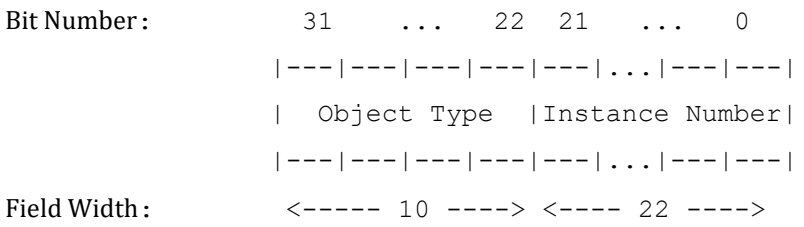
20.2.14 Encoding of an Object Identifier Value

A BACnet Object Identifier value shall consist of two components:

(1) A 10-bit object type, representing the BACnetObjectType of the object, with bit 9 the most significant bit and bit 0 the least significant. For objects defined in this standard, the value for this field shall be determined by the BACnetObjectType enumeration in Clause 21.
(2) A 22-bit object instance number, with bit 21 the most significant bit and bit 0 the least significant.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved



The encoding of an object identifier value shall be primitive, with four contents octets as follows:

Bits 9 through 2 of the object type shall be encoded in bits 7 through 0 of the first contents octet. Bits 1 through 0 of the object type shall be encoded in bits 7 through 6 of the second contents octet.

Bits 21 through 16 of the object instance shall be encoded in bits 5 through 0 of the second contents octet. Bits 15 through 8 of the object instance shall be encoded in bits 7 through 0 of the third contents octet. Bits 7 through 0 of the object instance shall be encoded in bits 7 through 0 of the fourth contents octet.

Example: Application-tagged object identifier value

ASN.1 =	ObjectIdentifier
Value =	(Binary Input, 15)
Application Tag =	ObjectIdentifier (Tag Number = 12)
Encoded Tag =	X'C4'
Encoded Data =	X'00C0000F'

20.2.15 Encoding of a Tagged Value

The encoding of a tagged value shall be derived from the complete encoding of the corresponding data value.

ISO 8824 defines the keywords "IMPLICIT" and "EXPLICIT," with "EXPLICIT" the default. Clause 21 begins with a "DEFINITION IMPLICIT TAGS," which changes the default to IMPLICIT. BACnet ASN.1 definitions are in terms of this default and use EXPLICIT only as an override.

If the "EXPLICIT" keyword is used in the production for the type, the encoding shall be constructed, and the contents octets shall be the complete base encoding, including tags.

If the "EXPLICIT" keyword is not used in the definition of the type, then

a) the encoding shall be constructed if the base encoding is constructed and shall be primitive otherwise, and either

b) the contents octets shall be the same as the contents octets of the base encoding if the base encoding is not primitively tagged application class Boolean or

c) the contents octet shall contain the value B'00000000' to denote a Boolean value of FALSE or B'00000001' to denote a Boolean value of TRUE if the base encoding is primitively tagged application class Boolean.

The context tag numbers shown in the following examples are for illustrative purposes only.

Example: Context-tagged null value

ASN.1 =	[3] NULL
Context Tag =	3
Encoded Tag =	X'38'

Example: Context-tagged Boolean value

ASN.1 =	[6] BOOLEAN
Value =	FALSE
Context Tag =	6
Encoded Tag =	X'69'
Encoded Data =	X'00'

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Example: Context-tagged Boolean value with context tag number greater than 14

ASN.1 =	[27] BOOLEAN
Value =	FALSE
Context Tag =	27
Encoded Tag =	X'F9'
Tag Number Extension =	X'1B'
Encoded Data =	X'00'

Example: Context-tagged unsigned integer

ASN.1 =	[0] Unsigned
Value =	256
Context Tag =	0
Encoded Tag =	X'0A'
Encoded Data =	X'0100'

Example: Context-tagged signed integer

ASN.1 =	[5] INTEGER
Value =	-72
Context Tag =	5
Encoded Tag =	X'59'
Encoded Data =	X'B8'

Example: Context-tagged signed integer with context tag number greater than 14

ASN.1 =	[33] INTEGER
Value =	-72
Context Tag =	33
Encoded Tag =	X'F9'
Tag Number Extension =	X'21'
Encoded Data =	X'B8'

Example: Context-tagged single precision real

ASN.1 =	[0] REAL
Value =	-33.3
Context Tag =	0
Encoded Tag =	X'0C'
Encoded Data =	X'C2053333'

Example: Context-tagged double precision real

ASN.1 =	[1] Double
Value =	-33.3
Context Tag =	1
Encoded Tag =	X'1D'
Length Extension =	X'08'
Encoded Data =	X'C040A66666666666'

Example: Context-tagged double precision real with context tag number greater than 14

ASN.1 =	[85] Double
Value =	-33.3
Context Tag =	85
Encoded Tag =	X'FD'
Tag Number Extension =	X'55'
Length Extension =	X'08'
Encoded Data =	X'C040A66666666666'

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Example: Context-tagged octet string

ASN.1 =	[1] OctetString
Value =	X'4321'
Context Tag =	1
Encoded Tag =	X'1A'
Encoded Data =	X'4321'

Example: Context-tagged character string

ASN.1 =	[5] CharacterString
Value =	"This is a BACnet string!" (ISO 10646 UTF-8)
Context Tag =	5
Encoded Tag =	X'5D'
Length Extension =	X'19'
Character Set =	X'00' (ISO 10646 UTF-8)
Encoded Data =	X'546869732069732061204241436E657420737472696E6721'

Example: Context-tagged character string with context tag number greater than 14

ASN.1 =	[127] CharacterString
Value =	"This is a BACnet string!" (ISO 10646 UTF-8)
Context Tag =	127
Encoded Tag =	X'FD'
Tag Number Extension =	X'7F'
Length Extension =	X'19'
Character Set =	X'00' (ISO 10646 UTF-8)
Encoded Data =	X'546869732069732061204241436E657420737472696E6721'

Example: Context-tagged bit string

ASN.1 =	[0] BIT STRING
Value =	B'10101'
Context Tag =	0
Encoded Tag =	X'0A'
Unused Bits in Last Octet =	X'03'
Encoded Data =	X'A8'

Example: Context-tagged enumeration

ASN.1 =	[9] BACnetObjectType
Value =	ANALOG-INPUT (0)
Context Tag =	9
Encoded Tag =	X'99'
Encoded Data =	X'00'

Example: Context-tagged date value

ASN.1 =	[9] Date
Value =	January 24, 1991 (Day of week = Thursday)
Context Tag =	9
Encoded Tag =	X'9C'
Encoded Data =	X'5B011804'

Example: Context-tagged time value

ASN.1 =	[4] Time
Value =	5:35:45.17 P.M. = 17:35:45.17
Context Tag =	4
Encoded Tag =	X'4C'
Encoded Data =	X'11232D11'

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Example: Context-tagged object identifier value

ASN.1 =	[4] ObjectIdentifier
Value =	(Binary Input, 15)
Context Tag =	4
Encoded Tag =	X'4C'
Encoded Data =	X'00C0000F'

20.2.16 Encoding of a Sequence Value

The encoding of a sequence value shall consist of the complete encoding, including tags, of one data value from each of the types listed in the ASN.1 production for the sequence type, in the order of their appearance in the definition, unless the type was referenced with the keyword "OPTIONAL".

The encoding of a data value may, but need not, be present for a type that was referenced with the keyword "OPTIONAL". If present, it shall appear in the encoding at the point corresponding to the appearance of the type in the ASN.1 definition.

Example: SEQUENCE value

ASN.1 =	BACnetDateTime
Value =	January 24, 1991, 5:35:45.17 P.M.
Application Tag = Date (Tag Number = 10)	
Encoded Tag =	X'A4'
Encoded Data =	X'5B011805'
Application Tag = Time (Tag Number = 11)	
Encoded Tag =	X'B4'
Encoded Data =	X'11232D11'

Example: Context-tagged SEQUENCE value

ASN.1 =	[0] BACnetDateTime
Value =	January 24, 1991, 5:35:45.17 P.M.
Context Tag =	0
Encoded Tag =	X'0E' (opening tag)
Application Tag = Date (Tag Number = 10)	
Encoded Tag =	X'A4'
Encoded Data =	X'5B011805'
Application Tag = Time (Tag Number = 11)	
Encoded Tag =	X'B4'
Encoded Data =	X'11232D11'
Encoded Tag =	X'0F' (closing tag)

Example: Context-tagged SEQUENCE value with context tag number greater than 14

ASN.1 =	[47] BACnetDateTime
Value =	January 24, 1991, 5:35:45.17 P.M.
Context Tag =	47
Encoded Tag =	X'FE' (opening tag)
Tag Number Extension =	X'2F'
Application Tag = Date (Tag Number = 10)	
Encoded Tag =	X'A4'
Encoded Data =	X'5B011805'
Application Tag = Time (Tag Number = 11)	
Encoded Tag =	X'B4'
Encoded Data =	X'11232D11'
Encoded Tag =	X'FF' (closing tag)
Tag Number Extension =	X'2F'

All ASN.1 productions of sequences that contain structured elements shall have distinct tags as necessary to permit unambiguous encoding and decoding of values. The follow example illustrates this requirement.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
(incorrect usage)
Var1 ::= SEQUENCE {
        varn1 SEQUENCE {
                varn2       [1] INTEGER,
                varn3       [2] INTEGER OPTIONAL
                },
        varn4 SEQUENCE {
                varn5       [1] INTEGER OPTIONAL,
                varn6       [2] INTEGER
                }
        }

(correct usage)
Var1 ::= SEQUENCE {
        varn1 SEQUENCE {
                varn2       [1] INTEGER,
                varn3       [2] INTEGER OPTIONAL
                },
        varn4 SEQUENCE {
                varn5       [3] INTEGER OPTIONAL,
                varn6       [4] INTEGER
                }
        }
```

20.2.17 Encoding of a Sequence-Of Value

The encoding of a sequence-of value shall consist of zero, one, or more complete encodings, including tags, of data values from the types listed in the ASN.1 definition.

The use of OPTIONAL components or ABSTRACT-SYNTAX.&Type in datatypes can lead to ambiguous parsing of concatenations. Therefore, the members of a Sequence-Of shall be restricted to datatypes that can be unambiguously parsed when concatenated.

The order of the encodings of the data values shall be the same as the order of the data values in the sequence-of value to be encoded.

Example: SEQUENCE OF primitive data

```
ASN.1 =                 SEQUENCE OF INTEGER
Value =                 1,2,4
Application Tag = Unsigned Integer (Tag Number = 2)
Encoded Tag =           X'21'
Encoded Data =          X'01'
Application Tag = Unsigned Integer (Tag Number = 2)
Encoded Tag =           X'21'
Encoded Data =          X'02'
Application Tag = Unsigned Integer (Tag Number = 2)
Encoded Tag =           X'21'
Encoded Data =          X'04'
```

Example: Context-tagged SEQUENCE OF primitive data

```
ASN.1 =                 [1] SEQUENCE OF INTEGER
Value =                 1,2,4
Encoded Tag =           X'1E' (Opening Tag)
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'01'
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Application Tag = Unsigned Integer (Tag Number = 2)
Encoded Tag = X'21'
Encoded Data = X'02'
Application Tag = Unsigned Integer (Tag Number = 2)
Encoded Tag = X'21'
Encoded Data = X'04'
Encoded Tag = X'1F' (Closing Tag)

Example: SEQUENCE OF constructed data
ASN.1 = SEQUENCE OF BACnetDateTime
Value = (January 24, 1991, 5:00 P.M.),
(January 24, 1991, 6:45 P.M.)
Application Tag = Date (Tag Number = 10)
Encoded Tag = X'A4'
Encoded Data = X'5B011804'
Application Tag = Time (Tag Number = 11)
Encoded Tag = X'B4'
Encoded Data = X'11000000'
Application Tag = Date (Tag Number = 10)
Encoded Tag = X'A4'
Encoded Data = X'5B011804'
Application Tag = Time (Tag Number = 11)
Encoded Tag = X'B4'
Encoded Data = X'122D0000'

20.2.18 Encoding of a Choice Value

The encoding of a CHOICE value shall be the same as the encoding of a value of the chosen type. The encoding may be primitive or constructed depending on the chosen type.

Example: CHOICE of primitive data
ASN.1 = BACnetTimeStamp
Value = 15
Context Tag = 1 (Choice for 'sequence number' in BACnetTimeStamp)
Encoded Tag = X'19'
Encoded Data = X'0F'

Example: CHOICE of constructed data
ASN.1 = BACnetTimeStamp
Value = January 24, 1991, 5:45.17 P.M.
Context Tag = 2 (Choice for 'dateTime' in BACnetTimeStamp)
Encoded Tag = X'2E' (Opening Tag)
Application Tag = Date (Tag Number = 10)
Encoded Tag = X'A4'
Encoded Data = X'5B011804'
Application Tag = Time (Tag Number = 11)
Encoded Tag = X'B4'
Encoded Data = X'11232D11'
Encoded Tag = X'2F' (Closing Tag)

20.2.19 Encoding of a Value of the ANY Type

The encoding of an ANY type shall be the complete encoding specified in this standard for the type of the value substituted for the placeholder ANY. This is represented in ASN.1 by ABSTRACT-SYNTAX.&Type.

20.2.20 Encoding of a BACnetARRAY of BACnetLIST datatype

A BACnetARRAY of BACnetLIST property as a whole is encoded to allow recognition of array elements. Since each list element is of the same datatype, regular Sequence-Of encoding of both the array elements and the contents of the array elements would result in a sequence of values where the array elements cannot be distinguished. Therefore, a

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnetARRAY of BACnetLIST property shall be encoded, when accessed as a whole, as if it had been defined in ASN.1 as follows:

```
Property-Value ::= SEQUENCE OF SEQUENCE {
        array-element  [0] SEQUENCE OF <datatype>
        }
```

In this ASN.1 production, the <datatype> is a placeholder for the datatype defined for the single list elements in the array elements of the BACnetARRAY of BACnetLIST property.

The context tag [0] shall always be encoded as an open/close context tag, creating a frame for list elements and delimiting array elements.

Example: BACnetARRAY of BACnetLIST of primitive data list elements

```
Property =          BACnetARRAY of BACnetLIST of Unsigned
Implied ASN.1:      Property-Value ::= SEQUENCE OF SEQUENCE {
                                           array-element  [0] SEQUENCE OF Unsigned
                                           }
Value =             ((1,2,3),(),(4,5,6))
Encoded Tag =       X'0E' (Opening Tag)
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'01'
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'02'
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'03'
Encoded Tag =       X'0F' (Closing Tag)
Encoded Tag =       X'0E' (Opening Tag)
Encoded Tag =       X'0F' (Closing Tag)
Encoded Tag =       X'0E' (Opening Tag)
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'04'
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'05'
        Application Tag = Unsigned Integer (Tag Number = 2)
        Encoded Tag =           X'21'
        Encoded Data =          X'06'
Encoded Tag =       X'0F' (Closing Tag)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21 FORMAL DESCRIPTION OF APPLICATION PROTOCOL DATA UNITS

This clause consists of an ASN.1 module that defines the BACnet APDUs and all necessary underlying datatypes. Clauses 13 through 17 contain many service parameters that are defined as conditional (C) or user optional (U). Both of these parameter types are designated as OPTIONAL in the ASN.1 productions to indicate that they may or may not be present in the PDU. The use of OPTIONAL in the ASN.1 production shall not supersede the conditional requirements defined in the service specification.

21.1 APDU Definitions

```
BACnetPDU ::= CHOICE {
        confirmed-request-pdu           [0] BACnet-Confirmed-Request-PDU,
        unconfirmed-request-pdu         [1] BACnet-Unconfirmed-Request-PDU,
        simple-ack-pdu                  [2] BACnet-SimpleACK-PDU,
        complex-ack-pdu                 [3] BACnet-ComplexACK-PDU,
        segment-ack-pdu                 [4] BACnet-SegmentACK-PDU,
        error-pdu                       [5] BACnet-Error-PDU,
        reject-pdu                      [6] BACnet-Reject-PDU,
        abort-pdu                       [7] BACnet-Abort-PDU
        }

BACnet-Confirmed-Request-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 0 for this PDU type
        segmented-message               [1] BOOLEAN,
        more-follows                    [2] BOOLEAN,
        segmented-response-accepted     [3] BOOLEAN,
        reserved                        [4] Unsigned (0..3),    -- shall be set to zero
        max-segments-accepted           [5] Unsigned (0..7),    -- as per Clause 20.1.2.4
        max-apdu-length-accepted        [6] Unsigned (0..15),   -- as per Clause 20.1.2.5
        invoke-id                       [7] Unsigned (0..255),
        sequence-number                 [8] Unsigned (0..255) OPTIONAL, -- only if segmented message
        proposed-window-size            [9] Unsigned (1..127) OPTIONAL, -- only if segmented message
        service-choice                  [10] BACnetConfirmedServiceChoice,
        service-request                 [11] BACnet-Confirmed-Service-Request OPTIONAL
-- Context-specific tags 0..11 are NOT used in header encoding
        }

BACnet-Unconfirmed-Request-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 1 for this PDU type
        reserved                        [1] Unsigned (0..15), -- shall be set to zero
        service-choice                  [2] BACnetUnconfirmedServiceChoice,
        service-request                 [3] BACnet-Unconfirmed-Service-Request
-- Context-specific tags 0..3 are NOT used in header encoding
        }

BACnet-SimpleACK-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 2 for this PDU type
        reserved                        [1] Unsigned (0..15), -- shall be set to zero
        invoke-id                       [2] Unsigned (0..255),
        service-ack-choice              [3] BACnetConfirmedServiceChoice
-- Context-specific tags 0..3 are NOT used in header encoding
        }

BACnet-ComplexACK-PDU ::= SEQUENCE {
        pdu-type                        [0] Unsigned (0..15), -- 3 for this PDU type
        segmented-message               [1] BOOLEAN,
        more-follows                    [2] BOOLEAN,
        reserved                        [3] Unsigned (0..3), -- shall be set to zero
        invoke-id                       [4] Unsigned (0..255),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        sequence-number             [5] Unsigned (0..255) OPTIONAL, -- only if segmented message
        proposed-window-size        [6] Unsigned (1..127) OPTIONAL, -- only if segmented message
        service-ack-choice          [7] BACnetConfirmedServiceChoice,
        service-ack                 [8] BACnet-Confirmed-Service-ACK
-- Context-specific tags 0..8 are NOT used in header encoding
        }

BACnet-SegmentACK-PDU ::= SEQUENCE {
        pdu-type                    [0] Unsigned (0..15), -- 4 for this PDU type
        reserved                    [1] Unsigned (0..3), -- shall be set to zero
        negative-ack                [2] BOOLEAN,
        server                      [3] BOOLEAN,
        original-invoke-id          [4] Unsigned (0..255),
        sequence-number             [5] Unsigned (0..255),
        actual-window-size          [6] Unsigned (1..127)
-- Context-specific tags 0..6 are NOT used in header encoding
        }

BACnet-Error-PDU ::= SEQUENCE {
        pdu-type                    [0] Unsigned (0..15), -- 5 for this PDU type
        reserved                    [1] Unsigned (0..15), -- shall be set to zero
        original-invoke-id          [2] Unsigned (0..255),
        error-choice                [3] BACnetConfirmedServiceChoice,
        error                       [4] BACnet-Error
-- Context-specific tags 0..4 are NOT used in header encoding
        }

BACnet-Reject-PDU ::= SEQUENCE {
        pdu-type                    [0] Unsigned (0..15), -- 6 for this PDU type
        reserved                    [1] Unsigned (0..15), -- shall be set to zero
        original-invoke-id          [2] Unsigned (0..255),
        reject-reason               [3] BACnetRejectReason
-- Context-specific tags 0..3 are NOT used in the header encoding
        }

BACnet-Abort-PDU ::= SEQUENCE {
        pdu-type                    [0] Unsigned (0..15), -- 7 for this PDU type
        reserved                    [1] Unsigned (0..7), -- shall be set to zero
        server                      [2] BOOLEAN,
        original-invoke-id          [3] Unsigned (0..255),
        abort-reason                [4] BACnetAbortReason
-- Context-specific tags 0..4 are NOT used in header encoding
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.2 Confirmed Service Productions

BACnetConfirmedServiceChoice ::= ENUMERATED {
-- Alarm and Event Services
acknowledge-alarm (0),
confirmed-audit-notification (32),
confirmed-cov-notification (1),
confirmed-cov-notification-multiple (31),
confirmed-event-notification (2),
get-alarm-summary (3),
get-enrollment-summary (4),
get-event-information (29),
life-safety-operation (27),
subscribe-cov (5),
subscribe-cov-property (28),
subscribe-cov-property-multiple (30),

-- File Access Services
atomic-read-file (6),
atomic-write-file (7),

-- Object Access Services
add-list-element (8),
remove-list-element (9),
create-object (10),
delete-object (11),
read-property (12),
read-property-multiple (14),
read-range (26),
write-property (15),
write-property-multiple (16),
audit-log-query (33),

-- Remote Device Management Services
device-communication-control (17),
confirmed-private-transfer (18),
confirmed-text-message (19),
reinitialize-device (20),

-- Virtual Terminal Services
vt-open (21),
vt-close (22),
vt-data (23)

-- Removed Services
-- formerly: authenticate (24) removed in version 1 revision 11
-- formerly: request-key (25) removed in version 1 revision 11
-- formerly: read-property-conditional (13) removed in version 1 revision 12

-- Services added after 1995
-- read-range (26) see Object Access Services
-- life-safety-operation (27) see Alarm and Event Services
-- subscribe-cov-property (28) see Alarm and Event Services
-- get-event-information (29) see Alarm and Event Services

-- Services added after 2012
-- subscribe-cov-property-multiple (30) see Alarm and Event Services
-- confirmed-cov-notification-multiple (31) see Alarm and Event Services
}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- Services added after 2016
        -- confirmed-audit-notification     (32)  see Alarm and Event Services
        -- audit-log-query                  (33)  see Object Access Services
        }
-- Other services to be added as they are defined. All enumeration values in this production are reserved for definition by
-- ASHRAE. Proprietary extensions are made by using the ConfirmedPrivateTransfer or UnconfirmedPrivateTransfer
-- services. See Clause 23.

BACnet-Confirmed-Service-Request ::= CHOICE {
-- Alarm and Event Services
        acknowledge-alarm                [0]   AcknowledgeAlarm-Request,
        confirmed-cov-notification       [1]   ConfirmedCOVNotification-Request,
        confirmed-cov-notification-multiple [31] ConfirmedCOVNotificationMultiple-Request,
        confirmed-event-notification     [2]   ConfirmedEventNotification-Request,
        -- get-alarm-summary conveys no parameters
        get-enrollment-summary           [4]   GetEnrollmentSummary-Request,
        get-event-information            [29]  GetEventInformation-Request,
        life-safety-operation            [27]  LifeSafetyOperation-Request,
        subscribe-cov                    [5]   SubscribeCOV-Request,
        subscribe-cov-property           [28]  SubscribeCOVProperty-Request,
        subscribe-cov-property-multiple  [30]  SubscribeCOVPropertyMultiple-Request,
        confirmed-audit-notification     [32]  ConfirmedAuditNotification-Request,

-- File Access Services
        atomic-read-file                 [6]   AtomicReadFile-Request,
        atomic-write-file                [7]   AtomicWriteFile-Request,

-- Object Access Services
        add-list-element                 [8]   AddListElement-Request,
        remove-list-element              [9]   RemoveListElement-Request,
        create-object                    [10]  CreateObject-Request,
        delete-object                    [11]  DeleteObject-Request,
        read-property                    [12]  ReadProperty-Request,
        read-property-multiple           [14]  ReadPropertyMultiple-Request,
        read-range                       [26]  ReadRange-Request,
        write-property                   [15]  WriteProperty-Request,
        write-property-multiple          [16]  WritePropertyMultiple-Request,
        audit-log-query                  [33]  AuditLogQuery-Request,

-- Remote Device Management Services
        device-communication-control     [17]  DeviceCommunicationControl-Request,
        confirmed-private-transfer       [18]  ConfirmedPrivateTransfer-Request,
        confirmed-text-message           [19]  ConfirmedTextMessage-Request,
        reinitialize-device              [20]  ReinitializeDevice-Request,

-- Virtual Terminal Services
        vt-open                          [21]  VT-Open-Request,
        vt-close                         [22]  VT-Close-Request,
        vt-data                          [23]  VT-Data-Request

-- Removed Services
        -- formerly: authenticate        [24]  removed in version 1 revision 11
        -- formerly: request-key         [25]  removed in version 1 revision 11
        -- formerly: read-property-conditional  [13]  removed in version 1 revision 12
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- Services added after 1995
        -- read-range                    [26]  see Object Access Services
        -- life-safety-operation         [27]  see Alarm and Event Services
        -- subscribe-cov-property        [28]  see Alarm and Event Services
        -- get-event-information         [29]  see Alarm and Event Services

-- Services added after 2012
        -- subscribe-cov-property-multiple      [30]  see Alarm and Event Services
        -- confirmed-cov-notification-multiple  [31]  see Alarm and Event Services

-- Services added after 2016
        -- confirmed-audit-notifications  [32]  see Alarm and Event Services
        -- audit-log-query                [33]  see Object Access Services
        }
-- Context-specific tags 0..33 are NOT used in the encoding. The tag number is transferred as the service-choice parameter
-- in the BACnet-Confirmed-Request-PDU.
--
-- Other services will be added as they are defined. All choice values in this production are reserved for definition by
-- ASHRAE. Proprietary extensions are made by using the ConfirmedPrivateTransfer service. See Clause 23.

BACnet-Confirmed-Service-ACK ::= CHOICE {
-- This production represents the Result(+) parameters defined for each confirmed service that returns one or more
-- parameters with 'Result(+).

-- Alarm and Event Services
        get-alarm-summary            [3]   GetAlarmSummary-ACK,
        get-enrollment-summary       [4]   GetEnrollmentSummary-ACK,
        get-event-information        [29]  GetEventInformation-ACK,

-- File Access Services
        atomic-read-file             [6]   AtomicReadFile-ACK,
        atomic-write-file            [7]   AtomicWriteFile-ACK,

-- Object Access Services
        create-object                [10]  CreateObject-ACK,
        read-property                [12]  ReadProperty-ACK,
        read-property-multiple       [14]  ReadPropertyMultiple-ACK,
        read-range                   [26]  ReadRange-ACK,
        audit-log-query              [33]  AuditLogQuery-ACK,

-- Remote Device Management Services
        confirmed-private-transfer   [18]  ConfirmedPrivateTransfer-ACK,

-- Virtual Terminal Services
        vt-open                      [21]  VT-Open-ACK,
        vt-data                      [23]  VT-Data-ACK

-- Removed Services
        -- formerly: authenticate                [24] removed in version 1 revision 11
        -- formerly: read-property-conditional   [13] removed in version 1 revision 12

-- Context-specific tags 3..33 are NOT used in the encoding. The tag number is transferred as the service-ack-choice
-- parameter in the BACnet-ComplexACK-PDU.
--
-- Other services to be added as they are defined. All choice values in this production are reserved for definition by
-- ASHRAE. Proprietary extensions are made by using the ConfirmedPrivateTransfer service.
-- See Clause 23.
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.2.1 Confirmed Alarm and Event Services

```
AcknowledgeAlarm-Request ::= SEQUENCE {
        acknowledging-process-identifier  [0] Unsigned32,
        event-object-identifier           [1] BACnetObjectIdentifier,
        event-state-acknowledged          [2] BACnetEventState,
        timestamp                         [3] BACnetTimeStamp,
        acknowledgment-source             [4] CharacterString,
        time-of-acknowledgment            [5] BACnetTimeStamp
        }

ConfirmedCOVNotification-Request ::= SEQUENCE {
        subscriber-process-identifier     [0] Unsigned32,
        initiating-device-identifier      [1] BACnetObjectIdentifier,
        monitored-object-identifier       [2] BACnetObjectIdentifier,
        time-remaining                    [3] Unsigned,
        list-of-values                    [4] SEQUENCE OF BACnetPropertyValue
        }

ConfirmedCOVNotificationMultiple-Request ::= SEQUENCE {
            subscriber-process-identifier       [0] Unsigned32,
            initiating-device-identifier        [1] BACnetObjectIdentifier,
            time-remaining                      [2] Unsigned,
            timestamp                           [3] BACnetDateTime OPTIONAL,
            list-of-cov-notifications                   [4] SEQUENCE OF SEQUENCE {
                    monitored-object-identifier         [0] BACnetObjectIdentifier,
                    list-of-values                      [1] SEQUENCE OF SEQUENCE {
                            property-identifier                 [0] BACnetPropertyIdentifier,
                            property-array-index                [1] Unsigned OPTIONAL, -- used only with array
datatype
                            property-value                      [2] ABSTRACT-SYNTAX.&Type,
                            time-of-change                      [3] Time OPTIONAL
                                                            }
                                                }
        }

ConfirmedEventNotification-Request ::= SEQUENCE {
        process-identifier                [0]  Unsigned32,
        initiating-device-identifier      [1]  BACnetObjectIdentifier,
        event-object-identifier           [2]  BACnetObjectIdentifier,
        timestamp                         [3]  BACnetTimeStamp,
        notification-class                [4]  Unsigned,
        priority                          [5]  Unsigned8,
        event-type                        [6]  BACnetEventType,
        message-text                      [7]  CharacterString OPTIONAL,
        notify-type                       [8]  BACnetNotifyType,
        ack-required                      [9]  BOOLEAN OPTIONAL,
        from-state                        [10] BACnetEventState OPTIONAL,
        to-state                          [11] BACnetEventState,
        event-values                      [12] BACnetNotificationParameters OPTIONAL
        }

GetAlarmSummary-ACK ::= SEQUENCE OF SEQUENCE {
        object-identifier                 BACnetObjectIdentifier,
        alarm-state                       BACnetEventState,
        acknowledged-transitions          BACnetEventTransitionBits
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
GetEnrollmentSummary-Request ::= SEQUENCE {
        acknowledgment-filter           [0] ENUMERATED {
                                                all                 (0),
                                                acked               (1),
                                                not-acked   (2)
                                                },
        enrollment-filter                       [1] BACnetRecipientProcess OPTIONAL,
        event-state-filter                      [2] ENUMERATED {
                                                offnormal   (0),
                                                fault               (1),
                                                normal              (2),
                                                all                 (3),
                                                active              (4)
                                                } OPTIONAL,
        event-type-filter                       [3] BACnetEventType OPTIONAL,
        priority-filter                         [4] SEQUENCE {
                                                min-priority        [0] Unsigned8,
                                                max-priority        [1] Unsigned8
                                                } OPTIONAL,
        notification-class-filter       [5] Unsigned OPTIONAL
        }

GetEnrollmentSummary-ACK ::= SEQUENCE OF SEQUENCE {
        object-identifier               BACnetObjectIdentifier,
        event-type                      BACnetEventType,
        event-state                     BACnetEventState,
        priority                        Unsigned8,
        notification-class              Unsigned OPTIONAL
        }

GetEventInformation-Request ::= SEQUENCE {
        last-received-object-identifier [0] BACnetObjectIdentifier OPTIONAL
        }

GetEventInformation-ACK ::= SEQUENCE {
        list-of-event-summaries         [0] SEQUENCE OF SEQUENCE {
                        object-identifier       [0] BACnetObjectIdentifier,
                        event-state             [1] BACnetEventState,
                        acknowledged-transitions [2] BACnetEventTransitionBits,
                        event-timestamps        [3] SEQUENCE SIZE (3) OF BACnetTimeStamp,
                        notify-type             [4] BACnetNotifyType,
                        event-enable            [5] BACnetEventTransitionBits,
                        event-priorities        [6] SEQUENCE SIZE (3) OF Unsigned
                        },
        more-events                     [1] BOOLEAN
        }

LifeSafetyOperation-Request ::= SEQUENCE {
        requesting-process-identifier   [0]  Unsigned32,
        requesting-source               [1]  CharacterString,
        request                         [2]  BACnetLifeSafetyOperation,
        object-identifier               [3]  BACnetObjectIdentifier OPTIONAL
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
SubscribeCOV-Request ::= SEQUENCE {
        subscriber-process-identifier       [0]   Unsigned32,
        monitored-object-identifier         [1]   BACnetObjectIdentifier,
        issue-confirmed-notifications       [2]   BOOLEAN OPTIONAL,
        lifetime                            [3]   Unsigned OPTIONAL
        }

SubscribeCOVProperty-Request ::= SEQUENCE {
        subscriber-process-identifier       [0]   Unsigned32,
        monitored-object-identifier         [1]   BACnetObjectIdentifier,
        issue-confirmed-notifications       [2]   BOOLEAN OPTIONAL,
        lifetime                            [3]   Unsigned OPTIONAL,
        monitored-property-identifier       [4]   BACnetPropertyReference,
        cov-increment                       [5]   REAL OPTIONAL
        }

SubscribeCOVPropertyMultiple-Request ::= SEQUENCE {
            subscriber-process-identifier                   [0] Unsigned32,
            issue-confirmed-notifications                   [1] BOOLEAN,
            lifetime                                        [2] Unsigned OPTIONAL,
            max-notification-delay                          [3] Unsigned OPTIONAL,
            list-of-cov-subscription-specifications         [4] SEQUENCE OF SEQUENCE {
                    monitored-object-identifier                     [0] BACnetObjectIdentifier,
                    list-of-cov-references                          [1] SEQUENCE OF SEQUENCE {
                            monitored-property                              [0] BACnetPropertyReference,
                            cov-increment                                   [1] REAL OPTIONAL,
                            timestamped                                     [2] BOOLEAN
                            }
                    }
            }

ConfirmedAuditNotification-Request ::= SEQUENCE {
        notifications                       [0]   SEQUENCE OF BACnetAuditNotification
        }
```

21.2.2 Confirmed File Access Services

```
AtomicReadFile-Request ::= SEQUENCE {
        file-identifier     BACnetObjectIdentifier,
        access-method       CHOICE {
                            stream-access       [0] SEQUENCE {
                                                    file-start-position         INTEGER,
                                                    requested-octet-count       Unsigned
                                                    },
                            record-access       [1] SEQUENCE {
                                                    file-start-record           INTEGER,
                                                    requested-record-count      Unsigned
                                                    }
                            }
        }

AtomicReadFile-ACK ::= SEQUENCE {
        end-of-file         BOOLEAN,
        access-method       CHOICE {
                            stream-access       [0] SEQUENCE {
                                                    file-start-position         INTEGER,
                                                    file-data                   OCTET STRING
                                                    },
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                        record-access    [1] SEQUENCE {
                                             file-start-record          INTEGER,
                                             returned-record-count      Unsigned,
                                             file-record-data           SEQUENCE OF OCTET STRING
                                             }
                        }
    }

AtomicWriteFile-Request ::= SEQUENCE {
    file-identifier     BACnetObjectIdentifier,
    access-method       CHOICE {
                        stream-access    [0] SEQUENCE {
                                             file-start-position        INTEGER,
                                             file-data          OCTET STRING
                                             },
                        record-access    [1] SEQUENCE {
                                             file-start-record  INTEGER,
                                             record-count       Unsigned,
                                             file-record-data   SEQUENCE OF OCTET STRING
                                             }
                        }
    }

AtomicWriteFile-ACK ::= CHOICE {
    file-start-position             [0] INTEGER,
    file-start-record               [1] INTEGER
    }
```

21.2.3 Confirmed Object Access Services

```
AddListElement-Request ::= SEQUENCE {
    object-identifier               [0] BACnetObjectIdentifier,
    property-identifier             [1] BACnetPropertyIdentifier,
    property-array-index            [2] Unsigned OPTIONAL, -- used only with array datatype
    list-of-elements                [3] ABSTRACT-SYNTAX.&Type
    }

CreateObject-Request ::= SEQUENCE {
    object-specifier                    [0] CHOICE {
                                            object-type             [0] BACnetObjectType,
                                            object-identifier       [1] BACnetObjectIdentifier
                                            },
    list-of-initial-values          [1] SEQUENCE OF BACnetPropertyValue OPTIONAL
    }

CreateObject-ACK ::= BACnetObjectIdentifier

DeleteObject-Request ::= SEQUENCE {
    object-identifier               BACnetObjectIdentifier
    }

ReadProperty-Request ::= SEQUENCE {
    object-identifier               [0] BACnetObjectIdentifier,
    property-identifier             [1] BACnetPropertyIdentifier,
    property-array-index            [2] Unsigned OPTIONAL   -- used only with array datatype
                                        -- if omitted with an array the entire array is referenced
    }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
ReadProperty-ACK ::= SEQUENCE {
        object-identifier               [0]  BACnetObjectIdentifier,
        property-identifier             [1]  BACnetPropertyIdentifier,
        property-array-index            [2]  Unsigned OPTIONAL,   -- used only with array datatype
                                             -- if omitted with an array the entire array is referenced
        property-value                  [3]  ABSTRACT-SYNTAX.&Type
        }

ReadPropertyMultiple-Request ::= SEQUENCE {
        list-of-read-access-specifications   SEQUENCE OF ReadAccessSpecification
        }

ReadPropertyMultiple-ACK ::= SEQUENCE {
        list-of-read-access-results          SEQUENCE OF ReadAccessResult
        }

ReadRange-Request ::= SEQUENCE {
        object-identifier               [0]  BACnetObjectIdentifier,
        property-identifier             [1]  BACnetPropertyIdentifier,
        property-array-index            [2]  Unsigned OPTIONAL,   -- used only with array datatype
        range                                CHOICE {
                by-position             [3]      SEQUENCE {
                                                 reference-index     Unsigned,
                                                 count               INTEGER16
                                                 },
                -- context tag 4 is deprecated
                -- context tag 5 is deprecated
                by-sequence-number      [6]  SEQUENCE {
                                               reference-sequence-number     Unsigned,
                                               count                         INTEGER16
                                             },
                by-time                 [7]  SEQUENCE {
                                               reference-time      BACnetDateTime,
                                               count               INTEGER16
                                             }
                } OPTIONAL
        }

ReadRange-ACK ::= SEQUENCE {
        object-identifier               [0]  BACnetObjectIdentifier,
        property-identifier             [1]  BACnetPropertyIdentifier,
        property-array-index            [2]  Unsigned OPTIONAL, -- used only with array datatype
        result-flags                    [3]  BACnetResultFlags,
        item-count                      [4]  Unsigned,
        item-data                       [5]  SEQUENCE OF ABSTRACT-SYNTAX.&Type,
        first-sequence-number           [6]  Unsigned32 OPTIONAL – used only if item-count > 0 and the request was
                                             either of
                                               -- type by-sequence-number or by-time
        }

RemoveListElement-Request ::= SEQUENCE {
        object-identifier               [0] BACnetObjectIdentifier,
        property-identifier             [1] BACnetPropertyIdentifier,
        property-array-index            [2] Unsigned OPTIONAL, -- used only with array datatypes
        list-of-elements                [3] ABSTRACT-SYNTAX.&Type
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

WriteProperty-Request ::= SEQUENCE {

object-identifier	[0]	BACnetObjectIdentifier,
property-identifier	[1]	BACnetPropertyIdentifier,
property-array-index	[2]	Unsigned OPTIONAL, -- used only with array datatype -- if omitted with an array the entire -- array is referenced
property-value	[3]	ABSTRACT-SYNTAX.&Type,
priority	[4]	Unsigned (1..16) OPTIONAL – used only when property is commandable

}

WritePropertyMultiple-Request ::= SEQUENCE {

list-of-write-access-specifications		SEQUENCE OF WriteAccessSpecification

}

AuditLogQuery-Request ::= SEQUENCE {

audit-log	[0]	BACnetObjectIdentifier,
query-parameters	[1]	BACnetAuditLogQueryParameters,
start-at-sequence-number	[2]	Unsigned32 OPTIONAL,
requested-count	[3]	Unsigned16

}

AuditLogQuery-ACK ::= SEQUENCE {

audit-log	[0]	BACnetObjectIdentifier,
records	[1]	SEQUENCE OF BACnetAuditLogRecordResult,
no-more-items	[2]	BOOLEAN

}

21.2.4 Confirmed Remote Device Management Services

ConfirmedPrivateTransfer-Request ::= SEQUENCE {

vendor-id	[0]	Unsigned16,
service-number	[1]	Unsigned,
service-parameters	[2]	ABSTRACT-SYNTAX.&Type OPTIONAL

}

ConfirmedPrivateTransfer-ACK ::= SEQUENCE {

vendor-id	[0]	Unsigned16,
service-number	[1]	Unsigned,
result-block	[2]	ABSTRACT-SYNTAX.&Type OPTIONAL

}

ConfirmedTextMessage-Request ::= SEQUENCE {

text-message-source-device	[0]	BACnetObjectIdentifier,	
message-class	[1]	CHOICE {	
		numeric	[0] Unsigned,
		character	[1] CharacterString
		} OPTIONAL,	
message-priority	[2]	ENUMERATED {	
		normal	(0),
		urgent	(1)
		},	
message	[3]	CharacterString	

}

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
DeviceCommunicationControl-Request ::= SEQUENCE {
        time-duration                   [0]   Unsigned16 OPTIONAL,
        enable-disable                  [1]   ENUMERATED {
                                                  enable              (0),
                                                  -- disable - deprecated in version 1 revision 20
                                                  disable-initiation  (2)
                                              },
        password                        [2]   CharacterString (SIZE (1..20)) OPTIONAL
        }

ReinitializeDevice-Request ::= SEQUENCE {
        reinitialized-state-of-device   [0]   ENUMERATED {
                                                  coldstart           (0),
                                                  warmstart           (1),
                                                  start-backup        (2),
                                                  end-backup          (3),
                                                  start-restore       (4),
                                                  end-restore         (5),
                                                  abort-restore       (6),
                                                  activate-changes    (7)
                                              },
        password                        [1]   CharacterString (SIZE (1..20)) OPTIONAL
        }
```

21.2.5 Confirmed Virtual Terminal Services

```
VT-Open-Request ::= SEQUENCE {
        vt-class                        BACnetVTClass,
        local-vt-session-identifier     Unsigned8
        }

VT-Open-ACK ::= SEQUENCE {
        remote-vt-session-identifier    Unsigned8
        }

VT-Close-Request ::= SEQUENCE {
        list-of-remote-vt-session-identifiers     SEQUENCE OF Unsigned8
        }

VT-Data-Request ::= SEQUENCE {
        vt-session-identifier           Unsigned8,
        vt-new-data                     OCTET STRING,
        vt-data-flag                    Unsigned (0..1)
        }

VT-Data-ACK ::= SEQUENCE {
        all-new-data-accepted           [0]   BOOLEAN,
        accepted-octet-count            [1]   Unsigned OPTIONAL – present only if all-new-data-accepted = FALSE
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.3 Unconfirmed Service Productions

```
BACnetUnconfirmedServiceChoice ::= ENUMERATED {
        i-am                                (0),
        i-have                              (1),
        unconfirmed-cov-notification        (2),
        unconfirmed-event-notification      (3),
        unconfirmed-private-transfer        (4),
        unconfirmed-text-message            (5),
        time-synchronization                (6),
        who-has                             (7),
        who-is                              (8),
        utc-time-synchronization            (9),
        write-group                         (10),
        unconfirmed-cov-notification-multiple       (11),
        unconfirmed-audit-notification      (12),
        who-Am-I                            (13),
        you-Are                             (14)
        }
-- Other services to be added as they are defined. All choice values in this production are reserved for definition by
-- ASHRAE. Proprietary extensions are made by using the UnconfirmedPrivateTransfer service. See Clause 23.

BACnet-Unconfirmed-Service-Request ::= CHOICE {
        i-am                                [0]  I-Am-Request,
        i-have                              [1]  I-Have-Request,
        unconfirmed-cov-notification        [2]  UnconfirmedCOVNotification-Request,
        unconfirmed-event-notification      [3]  UnconfirmedEventNotification-Request,
        unconfirmed-private-transfer        [4]  UnconfirmedPrivateTransfer-Request,
        unconfirmed-text-message            [5]  UnconfirmedTextMessage-Request,
        time-synchronization                [6]  TimeSynchronization-Request,
        who-has                             [7]  Who-Has-Request,
        who-is                              [8]  Who-Is-Request,
        utc-time-synchronization            [9]  UTCTimeSynchronization-Request,
        write-group                         [10] WriteGroup-Request,
        unconfirmed-cov-notification-multiple       [11] UnconfirmedCOVNotificationMultiple-Request,
        unconfirmed-audit-notification      [12] UnconfirmedAuditNotification-Request,
        who-Am-I                            [13] Who-Am-I_Request,
        you-Are                             [14] You-Are-Request
        }
-- Context-specific tags 0..14 are NOT used in the encoding. The tag number is transferred as the service-choice parameter
-- in the BACnet-Unconfirmed-Request-PDU.
--
-- Other services to be added as they are defined. All choice values in this production are reserved for definition by
-- ASHRAE. Proprietary extensions are made by using the UnconfirmedPrivateTransfer service.
-- See Clause 23.
```

21.3.1 Unconfirmed Alarm and Event Services

```
UnconfirmedCOVNotification-Request ::= SEQUENCE {
        subscriber-process-identifier       [0]  Unsigned32,
        initiating-device-identifier        [1]  BACnetObjectIdentifier,
        monitored-object-identifier         [2]  BACnetObjectIdentifier,
        time-remaining                      [3]  Unsigned,
        list-of-values                      [4]  SEQUENCE OF BACnetPropertyValue
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
UnconfirmedCOVNotificationMultiple-Request ::= SEQUENCE {
        subscriber-process-identifier       [0]     Unsigned32,
        initiating-device-identifier        [1]     BACnetObjectIdentifier,
        time-remaining                      [2]     Unsigned,
        timestamp                           [3]     BACnetDateTime OPTIONAL,
        list-of-cov-notifications           [4]     SEQUENCE OF SEQUENCE {
                monitored-object-identifier         [0] BACnetObjectIdentifier,
                list-of-values                      [1] SEQUENCE OF SEQUENCE{
                        property-identifier                 [0] BACnetPropertyIdentifier,
                        property-array-index                [1] Unsigned OPTIONAL, -- used only with array datatype
                        property-value                      [2] ABSTRACT-SYNTAX.&Type,
                        time-of-change                      [3] Time OPTIONAL
                                                            }
                }
        }

UnconfirmedEventNotification-Request ::= SEQUENCE {
        process-identifier              [0]     Unsigned32,
        initiating-device-identifier    [1]     BACnetObjectIdentifier,
        event-object-identifier         [2]     BACnetObjectIdentifier,
        timestamp                       [3]     BACnetTimeStamp,
        notification-class              [4]     Unsigned,
        priority                        [5]     Unsigned8,
        event-type                      [6]     BACnetEventType,
        message-text                    [7]     CharacterString OPTIONAL,
        notify-type                     [8]     BACnetNotifyType,
        ack-required                    [9]     BOOLEAN OPTIONAL,
        from-state                      [10]    BACnetEventState OPTIONAL,
        to-state                        [11]    BACnetEventState,
        event-values                    [12]    BACnetNotificationParameters OPTIONAL
        }

UnconfirmedAuditNotification-Request ::= SEQUENCE {
        notifications                   [0] SEQUENCE OF BACnetAuditNotification
        }
```

21.3.2 Unconfirmed Object Access Services

```
WriteGroup-Request ::= SEQUENCE {
        group-number                    [0]     Unsigned32,
        write-priority                  [1]     Unsigned (1..16),
        change-list                     [2]     SEQUENCE OF BACnetGroupChannelValue,
        inhibit-delay                   [3]     BOOLEAN OPTIONAL
        }
```

21.3.3 Unconfirmed Remote Device Management Services

```
I-Am-Request ::= SEQUENCE {
        i-am-device-identifier          BACnetObjectIdentifier,
        max-apdu-length-accepted        Unsigned,
        segmentation-supported          BACnetSegmentation,
        vendor-id                       Unsigned16
        }

I-Have-Request ::= SEQUENCE {
        device-identifier               BACnetObjectIdentifier,
        object-identifier               BACnetObjectIdentifier,
        object-name                     CharacterString
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
UnconfirmedPrivateTransfer-Request ::= SEQUENCE {
        vendor-id                       [0]   Unsigned16,
        service-number                  [1]   Unsigned,
        service-parameters              [2]   ABSTRACT-SYNTAX.&Type OPTIONAL
        }

UnconfirmedTextMessage-Request ::= SEQUENCE {
        text-message-source-device      [0] BACnetObjectIdentifier,
        message-class                   [1] CHOICE {
                                                numeric     [0] Unsigned,
                                                character   [1] CharacterString
                                                } OPTIONAL,
        message-priority                [2] ENUMERATED {
                                                normal      (0),
                                                urgent      (1)
                                                },
        message                         [3] CharacterString
        }

TimeSynchronization-Request ::= SEQUENCE {
        time                            BACnetDateTime
        }

UTCTimeSynchronization-Request ::= SEQUENCE {
        time                            BACnetDateTime
        }

Who-Am-I-Request ::= SEQUENCE {
        vendor-id                       Unsigned16,
        model-name                      CharacterString,
        serial-number                   CharacterString
        }

Who-Has-Request ::= SEQUENCE {
        limits      SEQUENCE {
                                device-instance-range-low-limit     [0] Unsigned (0..4194303),
                                device-instance-range-high-limit    [1] Unsigned (0..4194303)
                                } OPTIONAL,
        object      CHOICE {
                                object-identifier                   [2] BACnetObjectIdentifier,
                                object-name                         [3] CharacterString
                                }
        }

Who-Is-Request ::= SEQUENCE {
        device-instance-range-low-limit     [0] Unsigned (0..4194303) OPTIONAL, -- shall be used as a pair, see 16.10
        device-instance-range-high-limit    [1] Unsigned (0..4194303) OPTIONAL  -- shall be used as a pair, see 16.10
        }

You-Are-Request ::= SEQUENCE {
        vendor-id                       Unsigned16,
        model-name                      CharacterString,
        serial-number                   CharacterString,
        device-identifier               BACnetObjectIdentifier OPTIONAL,
        device-mac-address              OctetString OPTIONAL
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.4 Error Productions

```
BACnetAbortReason ::= ENUMERATED {
        other                                   (0),
        buffer-overflow                         (1),
        invalid-apdu-in-this-state              (2),
        preempted-by-higher-priority-task       (3),
        segmentation-not-supported              (4),
        security-error                          (5),
        insufficient-security                   (6),
        window-size-out-of-range                (7),
        application-exceeded-reply-time         (8),
        out-of-resources                        (9),
        tsm-timeout                             (10),
        apdu-too-long                           (11),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-255
        -- may be used by others subject to the procedures and constraints described in Clause 23.

BACnet-Error ::= CHOICE {
        other                                   [127] Error,
-- The remaining choices in this production represent the Result(-) parameters
-- defined for each confirmed service.

-- Alarm and Event Services
        acknowledge-alarm                       [0]   Error,
        confirmed-cov-notification              [1]   Error,
        confirmed-cov-notification-multiple     [31]  Error,
        confirmed-event-notification            [2]   Error,
        get-alarm-summary                       [3]   Error,
        get-enrollment-summary                  [4]   Error,
        get-event-information                   [29]  Error,
        life-safety-operation                   [27]  Error,
        subscribe-cov                           [5]   Error,
        subscribe-cov-property                  [28]  Error,
        subscribe-cov-property-multiple         [30]  SubscribeCOVPropertyMultiple-Error,
        confirmed-audit-notification            [32]  Error,

-- File Access Services
        atomic-read-file                        [6]   Error,
        atomic-write-file                       [7]   Error,

-- Object Access Services
        add-list-element                        [8]   ChangeList-Error,
        remove-list-element                     [9]   ChangeList-Error,
        create-object                           [10]  CreateObject-Error,
        delete-object                           [11]  Error,
        read-property                           [12]  Error,
        read-property-multiple                  [14]  Error,
        read-range                              [26]  Error,
        write-property                          [15]  Error,
        write-property-multiple                 [16]  WritePropertyMultiple-Error,
        audit-log-query                         [33]  Error,
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- Remote Device Management Services
        device-communication-control    [17] Error,
        confirmed-private-transfer      [18] ConfirmedPrivateTransfer-Error,
        confirmed-text-message          [19] Error,
        reinitialize-device             [20] Error,

-- Virtual Terminal Services
        vt-open                         [21] Error,
        vt-close                        [22] VTClose-Error,
        vt-data                         [23] Error

-- Removed Services
        -- formerly: authenticate       [24] removed in version 1 revision 11
        -- formerly: request-key        [25] removed in version 1 revision 11
        -- formerly: read-property-conditional     [13] removed in version 1 revision 12

-- Services added after 1995
        -- read-range                   [26] see Object Access Services
        -- life-safety-operation        [27] see Alarm and Event Services
        -- subscribe-cov-property       [28] see Alarm and Event Services
        -- get-event-information        [29] see Alarm and Event Services

-- Services added after 2012
        -- subscribe-cov-property-multiple      [30] see Alarm and Event Services
        -- confirmed-cov-notification-multiple  [31] see Alarm and Event Services

-- Services added after 2016
        -- confirmed-audit-notification [32] see Alarm and Event Services
        -- audit-log-query              [33] see Object Access Services
        }
-- Context-specific tags 0..33 and 127 are NOT used in the encoding. The tag number is transferred as the error-choice
-- parameter in the BACnet-Error-PDU.
--
-- Other services to be added as they are defined. All choice values in this production are reserved for definition by
-- ASHRAE. See Clause 23.

BACnetRejectReason ::= ENUMERATED {
        other                           (0),
        buffer-overflow                 (1),
        inconsistent-parameters         (2),
        invalid-parameter-data-type     (3),
        invalid-tag                     (4),
        missing-required-parameter      (5),
        parameter-out-of-range          (6),
        too-many-arguments              (7),
        undefined-enumeration           (8),
        unrecognized-service            (9),
        invalid-data-encoding           (10),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-255
        -- may be used by others subject to the procedures and constraints described in Clause 23.

ChangeList-Error ::= SEQUENCE {
        error-type                      [0] Error,
        first-failed-element-number     [1] Unsigned
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
CreateObject-Error ::= SEQUENCE {
        error-type                      [0]   Error,
        first-failed-element-number     [1]   Unsigned
        }

ConfirmedPrivateTransfer-Error ::= SEQUENCE {
        error-type                      [0] Error,
        vendor-id                       [1] Unsigned16,
        service-number                  [2] Unsigned,
        error-parameters                [3] ABSTRACT-SYNTAX.&Type OPTIONAL
        }

Error ::= SEQUENCE {
-- NOTE: The valid combinations of error-class and error-code are defined in Clause 18.
        error-class     ENUMERATED {
                          device                                (0),
                          object                                (1),
                          property                              (2),
                          resources                             (3),
                          security                              (4),
                          services                              (5),
                          vt                                    (6),
                          communication                         (7),

                        },
                        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
                        -- 64-65535 may be used by others subject to the procedures and constraints described
                        -- in Clause 23.

        error-code      ENUMERATED { -- see below for numerical order
                          abort-apdu-too-long                   (123),
                          abort-application-exceeded-reply-time (124),
                          abort-buffer-overflow                 (51),
                          abort-insufficient-security           (135),
                          abort-invalid-apdu-in-this-state      (52),
                          abort-other                           (56),
                          abort-out-of-resources                (125),
                          abort-preempted-by-higher-priority-task (53),
                          abort-proprietary                     (55),
                          abort-security-error                  (136),
                          abort-segmentation-not-supported      (54),
                          abort-tsm-timeout                     (126),
                          abort-window-size-out-of-range        (127),
                          access-denied                         (85),
                          addressing-error                      (115),
                          bad-destination-address               (86),
                          bad-destination-device-id             (87),
                          bad-signature                         (88),
                          bad-source-address                    (89),
                          busy                                  (82),
                          bvlc-function-unknown                 (143),
                          bvlc-proprietary-function-unknown     (144),
                          character-set-not-supported           (41),
                          communication-disabled                (83),
                          configuration-in-progress             (2),
                          cov-subscription-failed               (43),
                          datatype-not-supported                (47),
                          delete-fdt-entry-failed               (120),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
device-busy                          (3),
distribute-broadcast-failed          (121),
dns-error                            (192),
dns-name-resolution-failed           (190),
dns-resolver-failure                 (191),
dns-unavailable                      (189),
duplicate-entry                      (137),
duplicate-message                    (95),
duplicate-name                       (48),
duplicate-object-id                  (49),
dynamic-creation-not-supported       (4),
encryption-not-configured            (96),
encryption-required                  (97),
file-access-denied                   (5),
file-full                            (128),
header-encoding-error                (145),
header-not-understood                (146),
http-error                           (165),
http-no-upgrade                      (153),
http-not-a-server                    (164),
http-resource-not-local              (154),
http-proxy-authentication-failed     (155),
http-response-timeout                (156),
http-response-syntax-error           (157),
http-response-value-error            (158),
http-response-missing-header         (159),
http-temporary-unavailable           (163),
http-unexpected-response-code        (152),
http-upgrade-required                (161),
http-upgrade-error                   (162),
http-websocket-header-error          (160),
inconsistent-configuration           (129),
inconsistent-object-type             (130),
inconsistent-parameters              (7),
inconsistent-selection-criterion     (8),
internal-error                       (131),
invalid-array-index                  (42),
invalid-configuration-data           (46),
invalid-data-encoding                (142),
invalid-data-type                    (9),
invalid-event-state                  (73),
invalid-file-access-method           (10),
invalid-file-start-position          (11),
invalid-operation-in-this-state      (139),
invalid-parameter-data-type          (13),
invalid-tag                          (57),
invalid-timestamp                    (14),
invalid-value-in-this-state          (138),
ip-address-not-reachable             (198),
ip-error                             (199),
list-element-not-found               (81),
list-item-not-numbered               (140),
list-item-not-timestamped            (141),
log-buffer-full                      (75),
logged-value-purged                  (76),
malformed-message                    (101),
message-too-long                     (113),
message-incomplete                   (147),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

missing-required-parameter (16),
network-down (58),
no-alarm-configured (74),
no-objects-of-specified-type (17),
no-property-specified (77),
no-space-for-object (18),
no-space-to-add-list-element (19),
no-space-to-write-property (20),
no-vt-sessions-available (21),
node-duplicate-vmac (151),
not-a-bacnet-sc-hub (148),
not-configured (132),
not-configured-for-triggered-logging (78),
not-cov-property (44),
not-router-to-dnet (110),
object-deletion-not-permitted (23),
object-identifier-already-exists (24),
operational-problem (25),
optional-functionality-not-supported (45),
other (0),
out-of-memory (133),
parameter-out-of-range (80),
password-failure (26),
payload-expected (149),
property-is-not-a-list (22),
property-is-not-an-array (50),
read-access-denied (27),
read-bdt-failed (117),
read-fdt-failed (119),
register-foreign-device-failed (118),
reject-buffer-overflow (59),
reject-inconsistent-parameters (60),
reject-invalid-parameter-data-type (61),
reject-invalid-tag (62),
reject-missing-required-parameter (63),
reject-other (69),
reject-parameter-out-of-range (64),
reject-proprietary (68),
reject-too-many-arguments (65),
reject-undefined-enumeration (66),
reject-unrecognized-service (67),
router-busy (111),
security-error (114),
security-not-configured (103),
service-request-denied (29),
source-security-required (104),
success (84),
tcp-closed-by-local (195),
tcp-closed-other (196),
tcp-connect-timeout (193),
tcp-connection-refused (194),
tcp-error (197),
timeout (30),
tls-client-authentication-failed (182),
tls-client-certificate-error (180),
tls-client-certificate-expired (184),
tls-client-certificate-revoked (186),
tls-error (188),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        tls-server-authentication-failed        (183),
        tls-server-certificate-error            (181),
        tls-server-certificate-expired          (185),
        tls-server-certificate-revoked          (187),
        unexpected-data                         (150),
        unknown-authentication-type             (106),
        unknown-device                          (70),
        unknown-file-size                       (122),
        unknown-network-message                 (112),
        unknown-object                          (31),
        unknown-property                        (32),
        unknown-route                           (71),
        unknown-subscription                    (79),
        unknown-vt-class                        (34),
        unknown-vt-session                      (35),
        unsupported-object-type                 (36),
        value-not-initialized                   (72),
        value-out-of-range                      (37),
        value-too-long                          (134),
        vt-session-already-closed               (38),
        vt-session-termination-failure          (39),
        websocket-close-error                   (168),
        websocket-closed-abnormally             (173),
        websocket-closed-by-peer                (169),
        websocket-data-against-policy           (175),
        websocket-data-inconsistent             (174),
        websocket-data-not-accepted             (172),
        websocket-endpoint-leaves               (170),
        websocket-error                         (179),
        websocket-extension-missing             (177),
        websocket-frame-too-long                (176),
        websocket-protocol-error                (171),
        websocket-request-unavailable           (178),
        websocket-scheme-not-supported          (166),
        websocket-unknown-control-message       (167),
        write-access-denied                     (40),
        write-bdt-failed                        (116),
-- numerical order reference
        -- see other                            (0),
        -- formerly: authentication-failed      (1), removed version 1 revision 11
        -- see configuration-in-progress        (2),
        -- see device-busy                      (3),
        -- see dynamic-creation-not-supported   (4),
        -- see file-access-denied               (5),
        -- formerly:incompatible-security-levels (6), removed in version 1 revision 11
        -- see inconsistent-parameters          (7),
        -- see inconsistent-selection-criterion (8),
        -- see invalid-data-type                (9),
        -- see invalid-file-access-method       (10),
        -- see invalid-file-start-position      (11),
        -- formerly: invalid-operator-name      (12), removed in version 1 revision 11
        -- see invalid-parameter-data-type      (13),
        -- see invalid-timestamp                (14),
        -- formerly: key-generation-error       (15), removed in version 1 revision 11
        -- see missing-required-parameter       (16),
        -- see no-objects-of-specified-type     (17),
        -- see no-space-for-object              (18),
        -- see no-space-to-add-list-element     (19),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see no-space-to-write-property          (20),
        -- see no-vt-sessions-available            (21),
        -- see property-is-not-a-list              (22),
        -- see object-deletion-not-permitted       (23),
        -- see object-identifier-already-exists    (24),
        -- see operational-problem                 (25),
        -- see password-failure                    (26),
        -- see read-access-denied                  (27),
        -- formerly: security-not-supported        (28), removed in version 1 revision 11
        -- see service-request-denied              (29),
        -- see timeout                             (30),
        -- see unknown-object                      (31),
        -- see unknown-property                    (32),
        -- this enumeration was removed            (33),
        -- see unknown-vt-class                    (34),
        -- see unknown-vt-session                  (35),
        -- see unsupported-object-type             (36),
        -- see value-out-of-range                  (37),
        -- see vt-session-already-closed           (38),
        -- see vt-session-termination-failure      (39),
        -- see write-access-denied                 (40),
        -- see character-set-not-supported         (41),
        -- see invalid-array-index                 (42),
        -- see cov-subscription-failed             (43),
        -- see not-cov-property                    (44),
        -- see optional-functionality-not-supported (45),
        -- see invalid-configuration-data          (46),
        -- see datatype-not-supported              (47),
        -- see duplicate-name                      (48),
        -- see duplicate-object-id                 (49),
        -- see property-is-not-an-array            (50),
        -- see abort-buffer-overflow               (51),
        -- see abort-invalid-apdu-in-this-state    (52),
        -- see abort-preempted-by-higher-priority-task (53),
        -- see abort-segmentation-not-supported    (54),
        -- see abort-proprietary                   (55),
        -- see abort-other                         (56),
        -- see invalid-tag                         (57),
        -- see network-down                        (58),
        -- see reject-buffer-overflow              (59),
        -- see reject-inconsistent-parameters      (60),
        -- see reject-invalid-parameter-data-type  (61),
        -- see reject-invalid-tag                  (62),
        -- see reject-missing-required-parameter   (63),
        -- see reject-parameter-out-of-range       (64),
        -- see reject-too-many-arguments           (65),
        -- see reject-undefined-enumeration        (66),
        -- see reject-unrecognized-service         (67),
        -- see reject-proprietary                  (68),
        -- see reject-other                        (69),
        -- see unknown-device                      (70),
        -- see unknown-route                       (71),
        -- see value-not-initialized               (72),
        -- see invalid-event-state                 (73),
        -- see no-alarm-configured                 (74),
        -- see log-buffer-full                     (75),
        -- see logged-value-purged                 (76),
        -- see no-property-specified               (77),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see not-configured-for-triggered-logging (78),
        -- see unknown-subscription                 (79),
        -- see parameter-out-of-range               (80),
        -- see list-element-not-found               (81),
        -- see busy                                 (82),
        -- see communication-disabled               (83),
        -- see success                              (84),
        -- see access-denied                        (85),
        -- see bad-destination-address              (86),
        -- see bad-destination-device-id            (87),
        -- see bad-signature                        (88),
        -- see bad-source-address                   (89),
        -- formerly: bad-timestamp                  (90), removed in version 1 revision 22
        -- formerly: cannot-use-key                 (91), removed in version 1 revision 22
        -- formerly: cannot-verify-message-id       (92), removed in version 1 revision 22
        -- formerly: correct-key-revision           (93), removed in version 1 revision 22
        -- formerly destination-device-id-required  (94), removed in version 1 revision 22
        -- see duplicate-message                    (95),
        -- see encryption-not-configured            (96),
        -- see encryption-required                  (97),
        -- formerly: incorrect-key                  (98), removed in version 1 revision 22
        -- formerly: invalid-key-data               (99), removed in version 1 revision 22
        -- formerly: key-update-in-progress         (100), removed in version 1 revision 22
        -- see malformed-message                    (101),
        -- formerly: not-key-server                 (102), removed in version 1 revision 22
        -- see security-not-configured              (103),
        -- see source-security-required             (104),
        -- formerly: too-many-keys                  (105), removed in version 1 revision 22
        -- see unknown-authentication-type          (106),
        -- formerly: unknown-key                    (107), removed in version 1 revision 22
        -- formerly: unknown-key-revision           (108), removed in version 1 revision 22
        -- formerly: unknown-source-message         (109), removed in version 1 revision 22
        -- see not-router-to-dnet                   (110),
        -- see router-busy                          (111),
        -- see unknown-network-message              (112),
        -- see message-too-long                     (113),
        -- see security-error                       (114),
        -- see addressing-error                     (115),
        -- see write-bdt-failed                     (116),
        -- see read-bdt-failed                      (117),
        -- see register-foreign-device-failed       (118),
        -- see read-fdt-failed                      (119),
        -- see delete-fdt-entry-failed              (120),
        -- see distribute-broadcast-failed          (121),
        -- see unknown-file-size                    (122),
        -- see abort-apdu-too-long                  (123),
        -- see abort-application-exceeded-reply-time (124),
        -- see abort-out-of-resources               (125),
        -- see abort-tsm-timeout                    (126),
        -- see abort-window-size-out-of-range       (127),
        -- see file-full                            (128),
        -- see inconsistent-configuration           (129),
        -- see inconsistent-object-type             (130),
        -- see internal-error                       (131),
        -- see not-configured                       (132),
        -- see out-of-memory                        (133),
        -- see value-too-long                       (134),
        -- see abort-insufficient-security          (135),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see abort-security-error                (136),
        -- see duplicate-entry                     (137),
        -- see invalid-value-in-this-state         (138),
        -- see invalid-operation-in-this-state     (139),
        -- see list-item-not-numbered              (140),
        -- see list-item-not-timestamped           (141),
        -- see invalid-data-encoding               (142),
        -- see bvlc-function-unknown               (143),
        -- see bvlc-proprietary-function-unknown   (144),
        -- see header-encoding-error               (145),
        -- see header-not-understood               (146),
        -- see message-incomplete                  (147),
        -- see not-a-bacnet-sc-hub                 (148),
        -- see payload-expected                    (149),
        -- see unexpected-data                     (150),
        -- see node-duplicate-vmac                 (151),
        -- see http-unexpected-response-code       (152),
        -- see http-no-upgrade                     (153),
        -- see http-resource-not-local             (154),
        -- see http-proxy-authentication-failed    (155),
        -- see http-response-timeout               (156),
        -- see http-response-syntax-error          (157),
        -- see http-response-value-error           (158),
        -- see http-response-missing-header        (159),
        -- see http-websocket-header-error         (160),
        -- see http-upgrade-required               (161),
        -- see http-upgrade-error                  (162),
        -- see http-temporary-unavailable          (163),
        -- see http-not-a-server                   (164),
        -- see http-error                          (165),
        -- see websocket-scheme-not-supported      (166),
        -- see websocket-unknown-control-message   (167),
        -- see websocket-close-error               (168),
        -- see websocket-closed-by-peer            (169),
        -- see websocket-endpoint-leaves           (170),
        -- see websocket-protocol-error            (171),
        -- see websocket-data-not-accepted         (172),
        -- see websocket-closed-abnormally         (173),
        -- see websocket-data-inconsistent         (174),
        -- see websocket-data-against-policy       (175),
        -- see websocket-frame-too-long            (176),
        -- see websocket-extension-missing         (177),
        -- see websocket-request-unavailable       (178),
        -- see websocket-error                     (179),
        -- see tls-client-certificate-error        (180),
        -- see tls-server-certificate-error        (181),
        -- see tls-client-authentication-failed    (182),
        -- see tls-server-authentication-failed    (183),
        -- see tls-client-certificate-expired      (184),
        -- see tls-server-certificate-expired      (185),
        -- see tls-client-certificate-revoked      (186),
        -- see tls-server-certificate-revoked      (187),
        -- see tls-error                           (188),
        -- see dns-unavailable                     (189),
        -- see dns-name-resolution-failed          (190),
        -- see dns-resolver-failure                (191),
        -- see dns-error                           (192),
        -- see tcp-connect-timeout                 (193),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                -- see tcp-connection-refused          (194),
                -- see tcp-closed-by-local             (195),
                -- see tcp-closed-other                (196),
                -- see tcp-error                       (197),
                -- see ip-address-not-reachable        (198),
                -- see ip-error                        (199),
                ...
            }
            -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
            -- 256-65535 may be used by others subject to the procedures and constraints described
            -- in Clause 23.
    }

SubscribeCOVPropertyMultiple-Error ::= CHOICE {
    error-type                  [0]  Error,
    first-failed-subscription   [1]  SEQUENCE {
            monitored-object-identifier     [0] BACnetObjectIdentifier,
            monitored-property-reference            [1] BACnetPropertyReference,
            error-type                      [2] Error
                    }
    }

WritePropertyMultiple-Error ::= SEQUENCE {
    error-type                  [0]  Error,
    first-failed-write-attempt  [1]  BACnetObjectPropertyReference
    }

VTClose-Error ::= SEQUENCE {
    error-type                          [0] Error,
    list-of-vt-session-identifiers      [1] SEQUENCE OF Unsigned8 OPTIONAL
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.5 Application Types

-- The following productions are the definitions of the Application datatypes.
-- See Clause 20.2.1.4.

-- **NULL** [APPLICATION 0], equivalent to [UNIVERSAL 5]

-- **BOOLEAN** [APPLICATION 1], equivalent to [UNIVERSAL 1]

Unsigned ::= [APPLICATION 2] INTEGER (0..MAX)

Unsigned8 ::= Unsigned (0..255)

Unsigned16 ::= Unsigned (0..65535)

Unsigned32 ::= Unsigned (0..4294967295)

Unsigned64 ::= Unsigned (0.. 18446744073709551615) -- 0 .. 'the 64th power of two'-1

-- **INTEGER**[APPLICATION 3], equivalent to [UNIVERSAL 2]

INTEGER16 ::= INTEGER (-32768..32767)

-- **REAL** APPLICATION 4], equivalent to [UNIVERSAL 9] ANSI/IEEE-754 single precision floating point

Double ::= [APPLICATION 5] OCTET STRING (SIZE(8)) -- ANSI/IEEE-754 double precision floating point

-- **OCTET STRING** [APPLICATION 6], equivalent to [UNIVERSAL 4]

CharacterString ::= [APPLICATION 7] OCTET STRING -- see Clause 20.2.9 for supported types

-- **BIT STRING** [APPLICATION 8], equivalent to [UNIVERSAL 3]

-- **ENUMERATED** [APPLICATION 9], equivalent to [UNIVERSAL 10]

Date ::= [APPLICATION 10] OCTET STRING (SIZE(4)) -- see Clause 20.2.12
-- first octet year minus 1900, X'FF' = unspecified
-- second octet month (1.. 14), 1 = January
-- 13 = odd months
-- 14 = even months
-- X'FF' = unspecified
-- third octet day of month (1..34), 32 = last day of month
-- 33 = odd days of month
-- 34 = even days of month
-- X'FF' = unspecified
-- fourth octet day of week (1..7) 1 = Monday
-- 7 = Sunday
-- X'FF' = unspecified

Time ::= [APPLICATION 11] OCTET STRING (SIZE(4)) -- see Clause 20.2.13
-- first octet hour (0..23), X'FF' = unspecified
-- second octet minute (0..59), X'FF' = unspecified
-- third octet second (0..59), X'FF' = unspecified
-- fourth octet hundredths (0..99) X'FF' = unspecified

BACnetObjectIdentifier ::= [APPLICATION 12] OCTET STRING (SIZE(4)) -- see Clause 20.2.14

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

21.6 Base Types

```
BACnetAccessAuthenticationFactorDisable ::= ENUMERATED {
        none                        (0),
        disabled                    (1),
        disabled-lost               (2),
        disabled-stolen             (3),
        disabled-damaged            (4),
        disabled-destroyed          (5),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccessCredentialDisable ::= ENUMERATED {
        none                        (0),
        disable                     (1),
        disable-manual              (2),
        disable-lockout             (3),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccessCredentialDisableReason ::= ENUMERATED {
        disabled                    (0),
        disabled-needs-provisioning (1),
        disabled-unassigned         (2),
        disabled-not-yet-active     (3),
        disabled-expired            (4),
        disabled-lockout            (5),
        disabled-max-days           (6),
        disabled-max-uses           (7),
        disabled-inactivity         (8),
        disabled-manual             (9),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccessEvent ::= ENUMERATED {
        none                        (0),
        granted                     (1),
        muster                      (2),
        passback-detected           (3),
        duress                      (4),
        trace                       (5),
        lockout-max-attempts        (6),
        lockout-other               (7),
        lockout-relinquished        (8),
        locked-by-higher-priority   (9),
        out-of-service              (10),
        out-of-service-relinquished (11),
        accompaniment-by            (12),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        authentication-factor-read          (13),
        authorization-delayed               (14),
        verification-required               (15),
        no-entry-after-granted              (16),
        -- Enumerated values 128-511 are used for events which indicate that access has been denied.
        denied-deny-all                     (128),
        denied-unknown-credential           (129),
        denied-authentication-unavailable   (130),
        denied-authentication-factor-timeout (131),
        denied-incorrect-authentication-factor (132),
        denied-zone-no-access-rights        (133),
        denied-point-no-access-rights       (134),
        denied-no-access-rights             (135),
        denied-out-of-time-range            (136),
        denied-threat-level                 (137),
        denied-passback                     (138),
        denied-unexpected-location-usage    (139),
        denied-max-attempts                 (140),
        denied-lower-occupancy-limit        (141),
        denied-upper-occupancy-limit        (142),
        denied-authentication-factor-lost   (143),
        denied-authentication-factor-stolen (144),
        denied-authentication-factor-damaged (145),
        denied-authentication-factor-destroyed (146),
        denied-authentication-factor-disabled (147),
        denied-authentication-factor-error  (148),
        denied-credential-unassigned        (149),
        denied-credential-not-provisioned   (150),
        denied-credential-not-yet-active    (151),
        denied-credential-expired           (152),
        denied-credential-manual-disable    (153),
        denied-credential-lockout           (154),
        denied-credential-max-days          (155),
        denied-credential-max-uses          (156),
        denied-credential-inactivity        (157),
        denied-credential-disabled          (158),
        denied-no-accompaniment             (159),
        denied-incorrect-accompaniment      (160),
        denied-lockout                      (161),
        denied-verification-failed          (162),
        denied-verification-timeout         (163),
        denied-other                        (164),
        ...
        }
        -- Enumerated values 0-511 are reserved for definition by ASHRAE. Enumerated values
        -- 512-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccessPassbackMode ::= ENUMERATED {
        passback-off                        (0),
        hard-passback                       (1),
        soft-passback                       (2)
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetAccessRule ::= SEQUENCE {
        time-range-specifier            [0]   ENUMERATED {
                                                  specified       (0),
                                                  always          (1)
                                              },
        time-range                      [1]   BACnetDeviceObjectPropertyReference OPTIONAL,
                                              -- to be present if time-range-specifier has the value "specified"
        location-specifier                    [2]   ENUMERATED {
                                                        specified   (0),
                                                        all         (1)
                                                    },
        location                              [3]   BACnetDeviceObjectReference OPTIONAL,
                                                    -- to be present if location-specifier has the value "specified"
        enable                                [4]   BOOLEAN
        }

BACnetAccessThreatLevel ::= Unsigned(0..100)

BACnetAccessUserType ::= ENUMERATED {
        asset                           (0),
        group                           (1),
        person                          (2),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccessZoneOccupancyState ::= ENUMERATED {
        normal                          (0),
        below-lower-limit               (1),
        at-lower-limit                  (2),
        at-upper-limit                  (3),
        above-upper-limit               (4),
        disabled                        (5),
        not-supported                   (6),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAccumulatorRecord ::= SEQUENCE {
        timestamp                       [0]   BACnetDateTime,
        present-value                   [1]   Unsigned,
        accumulated-value               [2]   Unsigned,
        accumulator-status              [3]    ENUMERATED {
                                                  normal          (0),
                                                  starting        (1),
                                                  recovered       (2),
                                                  abnormal        (3),
                                                  failed                      (4)
                                              }
        }

BACnetAcknowledgeAlarmInfo ::= SEQUENCE {
        event-state-acknowledged        [0] BACnetEventState,
        timestamp                       [1] BACnetTimeStamp
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetAction ::= ENUMERATED {
        direct                          (0),
        reverse                         (1)
        }

BACnetActionCommand ::= SEQUENCE {
        device-identifier               [0]   BACnetObjectIdentifier OPTIONAL,
        object-identifier               [1]   BACnetObjectIdentifier,
        property-identifier             [2]   BACnetPropertyIdentifier,
        property-array-index            [3]   Unsigned OPTIONAL,   -- used only with array datatype
        property-value                  [4]   ABSTRACT-SYNTAX.&Type,
        priority                        [5]   Unsigned (1..16) OPTIONAL, -- used only when property is commandable
        post-delay                      [6]   Unsigned OPTIONAL,
        quit-on-failure                 [7]   BOOLEAN,
        write-successful                [8]   BOOLEAN
        }

BACnetActionList ::= SEQUENCE {
        action                          [0]   SEQUENCE OF BACnetActionCommand
        }

BACnetAddress ::= SEQUENCE {
        network-number                  Unsigned16,                 -- A value of 0 indicates the local network
        mac-address                     OCTET STRING  -- A string of length 0 indicates a broadcast
        }

BACnetAddressBinding ::= SEQUENCE {
        device-identifier               BACnetObjectIdentifier,
        device-address                  BACnetAddress
        }

BACnetAssignedAccessRights ::= SEQUENCE {
        assigned-access-rights          [0]   BACnetDeviceObjectReference,
        enable                          [1] BOOLEAN
        }

BACnetAssignedLandingCalls ::= SEQUENCE {
        landing-calls                   [0]   SEQUENCE OF SEQUENCE {
                                                  floor-number      [0] Unsigned8,
                                                  direction         [1] BACnetLiftCarDirection
                                              }
        }

BACnetAuditLevel ::= ENUMERATED {
        none                            (0),
        audit-all                       (1),
        audit-config                    (2),
        default                         (3),
        ...
        }
-- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values 128-255 may be used
-- by others subject to the procedures and constraints described in Clause 23.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetAuditLogRecord ::= SEQUENCE {
        timestamp                       [0]   BACnetDateTime,
        log-datum                       [1]   CHOICE {
                                                  log-status          [0] BACnetLogStatus,
                                                  audit-notification          [1] BACnetAuditNotification,
                                                  time-change         [2] REAL
                                              }
        }

BACnetAuditLogRecordResult ::= SEQUENCE {
        sequence-number                 [0]   Unsigned64,
        log-record                      [1]   BACnetAuditLogRecord
        }

BACnetAuditLogQueryParameters ::= CHOICE {
        by-target       [0] SEQUENCE {
                                        target-device-identifier    [0] BACnetObjectIdentifier,
                                        target-device-address       [1] BACnetAddress OPTIONAL,
                                        target-object-identifier    [2] BACnetObjectIdentifier OPTIONAL,
                                        target-property-identifier  [3] BACnetPropertyIdentifier OPTIONAL,
                                        target-array-index          [4] Unsigned OPTIONAL,
                                        target-priority             [5] Unsigned(1..16) OPTIONAL,
                                        operations                          [6]     BACnetAuditOperationFlags
OPTIONAL,
                                        successful-actions-only     [7] BOOLEAN
                        },
        by-source       [1] SEQUENCE {
                                        source-device-identifier    [0] BACnetObjectIdentifier,
                                        source-device-address       [1] BACnetAddress OPTIONAL,
                                        source-object-identifier    [2] BACnetObjectIdentifier OPTIONAL,
                                        operations                          [3]     BACnetAuditOperationFlags
OPTIONAL,
                                        successful-actions-only     [4] BOOLEAN
                        }
        }

BACnetAuditNotification ::= SEQUENCE {
        source-timestamp                [0] BACnetTimeStamp OPTIONAL,
        target-timestamp                [1] BACnetTimeStamp OPTIONAL,
        source-device                   [2] BACnetRecipient,
        source-object                   [3] BACnetObjectIdentifier OPTIONAL,
        operation                       [4] BACnetAuditOperation,
        source-comment                  [5] CharacterString OPTIONAL,
        target-comment                  [6] CharacterString OPTIONAL,
        invoke-id                       [7] Unsigned8 OPTIONAL,
        source-user-id                  [8] Unsigned16 OPTIONAL,
        source-user-role                [9] Unsigned8 OPTIONAL,
        target-device                   [10] BACnetRecipient,
        target-object                   [11] BACnetObjectIdentifier OPTIONAL,
        target-property                 [12] BACnetPropertyReference OPTIONAL,
        target-priority                 [13] Unsigned (1..16) OPTIONAL,
        target-value                    [14] ABSTRACT-SYNTAX.&Type OPTIONAL,
        current-value                   [15] ABSTRACT-SYNTAX.&Type OPTIONAL,
        result                          [16] Error OPTIONAL
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetAuditOperation ::= ENUMERATED {
        read                    (0),
        write                   (1),
        create                  (2),
        delete                  (3),
        life-safety             (4),
        acknowledge-alarm       (5),
        device-disable-comm     (6),
        device-enable-comm      (7),
        device-reset            (8),
        device-backup           (9),
        device-restore          (10),
        subscription            (11),
        notification            (12),
        auditing-failure        (13),
        network-changes         (14),
        general                 (15),
        ...
        }
-- Enumerated values 0-31 are reserved for definition by ASHRAE. Enumerated values 32-63 may be used
-- by others subject to the procedures and constraints described in Clause 23.
-- The enumerated values match the bit positions in BACnetAuditOperationFlags.

BACnetAuditOperationFlags ::= BIT STRING {
        read                    (0),
        write                   (1),
        create                  (2),
        delete                  (3),
        life-safety             (4),
        acknowledge-alarm       (5),
        device-disable-comm     (6),
        device-enable-comm      (7),
        device-reset            (8),
        device-backup           (9),
        device-restore          (10),
        subscription            (11),
        notification            (12),
        auditing-failure        (13),
        network-changes         (14),
        general                 (15),
        ...
        }
-- Bit positions correspond one-to-one with the BACnetAuditOperation enumeration values.
-- Bits 0-31 are reserved for definition by ASHRAE. Bits 32-63 may be used by others subject to
-- the procedures and constraints described in Clause 23 for BACnetAuditOperation enumeration values.

BACnetAuthenticationFactor ::= SEQUENCE {
        format-type             [0] BACnetAuthenticationFactorType,
        format-class            [1] Unsigned,
        value                   [2] OCTET STRING -- for encoding of values into this octet string see Annex P.
        }

BACnetAuthenticationFactorFormat ::= SEQUENCE {
        format-type             [0] BACnetAuthenticationFactorType,
        vendor-id               [1] Unsigned16 OPTIONAL,
        vendor-format           [2] Unsigned16 OPTIONAL
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetAuthenticationFactorType ::= ENUMERATED {
        undefined                       (0),
        error                           (1),
        custom                          (2),
        simple-number16                 (3),
        simple-number32                 (4),
        simple-number56                 (5),
        simple-alpha-numeric            (6),
        aba-track2                      (7),
        wiegand26                       (8),
        wiegand37                       (9),
        wiegand37-facility              (10),
        facility16-card32               (11),
        facility32-card32               (12),
        fasc-n                          (13),
        fasc-n-bcd                      (14),
        fasc-n-large                    (15),
        fasc-n-large-bcd                (16),
        gsa75                           (17),
        chuid                           (18),
        chuid-full                      (19),
        guid                            (20),
        cbeff-a                         (21),
        cbeff-b                         (22),
        cbeff-c                         (23),
        user-password                   (24)
        }

BACnetAuthenticationPolicy ::= SEQUENCE {
        policy                          [0] SEQUENCE OF SEQUENCE {
                                            credential-data-input   [0] BACnetDeviceObjectReference,
                                            index                       [1] Unsigned
                                            },
        order-enforced                  [1] BOOLEAN,
        timeout                         [2] Unsigned
        }

BACnetAuthenticationStatus ::= ENUMERATED {
        not-ready                       (0),
        ready                           (1),
        disabled                        (2),
        waiting-for-authentication-factor (3),
        waiting-for-accompaniment       (4),
        waiting-for-verification        (5),
        in-progress                     (6)
        }

BACnetAuthorizationExemption ::= ENUMERATED {
        passback                        (0),
        occupancy-check                 (1),
        access-rights                   (2),
        lockout                         (3),
        deny                            (4),
        verification                    (5),
        authorization-delay             (6),
        ...
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-255 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetAuthorizationMode ::= ENUMERATED {
        authorize                       (0),
        grant-active                    (1),
        deny-all                        (2),
        verification-required           (3),
        authorization-delayed           (4),
        none                            (5),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetBackupState ::= ENUMERATED {
        idle                            (0),
        preparing-for-backup            (1),
        preparing-for-restore           (2),
        performing-a-backup             (3),
        performing-a-restore            (4),
        backup-failure                  (5),
        restore-failure                 (6)
        }

BACnetBDTEntry ::= SEQUENCE {
        bbmd-address    [0] BACnetHostNPort,
        broadcast-mask  [1] OCTET STRING OPTIONAL     -- shall be present if BACnet/IP, and absent for BACnet/IPv6
        }

BACnetBinaryLightingPV ::= ENUMERATED {
        off                             (0),
        on                              (1),
        warn                            (2),
        warn-off                        (3),
        warn-relinquish                 (4),
        stop                            (5),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-255 may be used by
        -- others subject to the procedures and constraints described in Clause 23.

BACnetBinaryPV ::= ENUMERATED {
        inactive                        (0),
        active                          (1)
        }

BACnetCalendarEntry ::= CHOICE {
        date                            [0] Date,
        date-range                      [1] BACnetDateRange,
        weekNDay                        [2] BACnetWeekNDay
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetChannelValue ::= CHOICE {
        null                            NULL,
        real                            REAL,
        enumerated                      ENUMERATED,
        unsigned                        Unsigned,
        boolean                         BOOLEAN,
        integer                         INTEGER,
        double                          Double,
        time                            Time,
        characterstring                 CharacterString,
        octetstring                     OCTET STRING,
        bitstring                       BIT STRING,
        date                            Date,
        objectidentifier                BACnetObjectIdentifier,
        lighting-command                [0] BACnetLightingCommand
        }

BACnetClientCOV ::= CHOICE {
        real-increment                  REAL,
        default-increment               NULL
        }

BACnetCOVMultipleSubscription ::= SEQUENCE {
        recipient                       [0] BACnetRecipientProcess,
        issue-confirmed-notifications   [1] BOOLEAN,
        time-remaining                  [2] Unsigned,
        max-notification-delay          [3] Unsigned,
        list-of-cov-subscription-specifications     [4] SEQUENCE OF SEQUENCE {
                        monitored-object-identifier [0] BACnetObjectIdentifier,
                        list-of-cov-references      [1] SEQUENCE OF SEQUENCE {
                            monitored-property                  [0] BACnetPropertyReference,
                            cov-increment                       [1] REAL OPTIONAL,
                            timestamped                 [2] BOOLEAN
                            }
                        }
        }

BACnetCOVSubscription ::= SEQUENCE {
        recipient                       [0] BACnetRecipientProcess,
        monitored-property-reference    [1] BACnetObjectPropertyReference,
        issue-confirmed-notifications   [2] BOOLEAN,
        time-remaining                  [3] Unsigned,
        cov-increment                   [4] REAL OPTIONAL       -- used only with monitored
                                                                -- properties with a numeric datatype
        }

BACnetCredentialAuthenticationFactor ::= SEQUENCE {
        disable                         [0] BACnetAccessAuthenticationFactorDisable,
        authentication-factor           [1] BACnetAuthenticationFactor
        }

BACnetDailySchedule ::= SEQUENCE {
        day-schedule                    [0] SEQUENCE OF BACnetTimeValue
        }

BACnetDateRange ::= SEQUENCE {  -- see Clause 20.2.12 for restrictions
        start-date                      Date,
        end-date                        Date
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetDateTime ::= SEQUENCE {
        date                            Date, -- see Clause 20.2.12 for restrictions
        time                            Time -- see Clause 20.2.13 for restrictions
        }

BACnetDaysOfWeek ::= BIT STRING {
        monday                          (0),
        tuesday                         (1),
        wednesday                       (2),
        thursday                        (3),
        friday                          (4),
        saturday                        (5),
        sunday                          (6)
        }

BACnetDestination ::= SEQUENCE {
        valid-days                      BACnetDaysOfWeek,
        from-time                       Time,
        to-time                         Time,
        recipient                       BACnetRecipient,
        process-identifier              Unsigned32,
        issue-confirmed-notifications   BOOLEAN,
        transitions                     BACnetEventTransitionBits
        }

BACnetDeviceObjectPropertyReference ::= SEQUENCE {
        object-identifier               [0] BACnetObjectIdentifier,
        property-identifier             [1] BACnetPropertyIdentifier,
        property-array-index            [2] Unsigned OPTIONAL,  -- used only with array datatype
                                            -- if omitted with an array then
                                            -- the entire array is referenced
        device-identifier               [3] BACnetObjectIdentifier OPTIONAL
        }

BACnetDeviceObjectPropertyValue ::= SEQUENCE {
        device-identifier               [0] BACnetObjectIdentifier,
        object-identifier               [1] BACnetObjectIdentifier,
        property-identifier             [2] BACnetPropertyIdentifier,
        property-array-index            [3] Unsigned  OPTIONAL, -- used only with array datatype
        property-value                  [4] ABSTRACT-SYNTAX.&Type
        }

BACnetDeviceObjectReference ::= SEQUENCE {
        device-identifier               [0] BACnetObjectIdentifier OPTIONAL,
        object-identifier               [1] BACnetObjectIdentifier
        }

BACnetDeviceStatus ::= ENUMERATED {
        operational                     (0),
        operational-read-only           (1),
        download-required               (2),
        download-in-progress            (3),
        non-operational                 (4),
        backup-in-progress              (5),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

```
BACnetDoorAlarmState ::= ENUMERATED {
        normal                  (0),
        alarm                   (1),
        door-open-too-long      (2),
        forced-open             (3),
        tamper                  (4),
        door-fault              (5),
        lock-down               (6),
        free-access             (7),
        egress-open             (8),
        ...
        }
        -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
        -- 256-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetDoorSecuredStatus ::= ENUMERATED {
        secured                 (0),
        unsecured               (1),
        unknown                 (2)
        }

BACnetDoorStatus ::= ENUMERATED {
        closed                  (0),
        opened                  (1),
        unknown                 (2),
        door-fault              (3),
        unused                  (4),
        none                    (5),
        closing                 (6),
        opening                 (7),
        safety-locked           (8),
        limited-opened          (9),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetDoorValue ::= ENUMERATED {
        lock                    (0),
        unlock                  (1),
        pulse-unlock            (2),
        extended-pulse-unlock   (3)
        }

BACnetEngineeringUnits ::= ENUMERATED {  -- See below for numerical order
-- Acceleration
        meters-per-second-per-second    (166),

--Area
        square-meters           (0),
        square-centimeters      (116),
        square-feet             (1),
        square-inches           (115),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
--Currency
        currency1                       (105),
        currency2                       (106),
        currency3                       (107),
        currency4                       (108),
        currency5                       (109),
        currency6                       (110),
        currency7                       (111),
        currency8                       (112),
        currency9                       (113),
        currency10                      (114),

--Electrical
        milliamperes                    (2),
        amperes                         (3),
        amperes-per-meter               (167),
        amperes-per-square-meter        (168),
        ampere-square-meters            (169),
        decibels                        (199),
        decibels-millivolt              (200),
        decibels-volt                   (201),
        farads                          (170),
        henrys                          (171),
        ohms                            (4),
        ohm-meter-squared-per-meter     (237),
        ohm-meters                      (172),
        milliohms                       (145),
        kiloohms                        (122),
        megaohms                        (123),
        microsiemens                    (190),
        millisiemens                    (202),
        siemens                         (173),
        siemens-per-meter               (174),
        teslas                          (175),
        volts                           (5),
        millivolts                      (124),
        kilovolts                       (6),
        megavolts                       (7),
        volt-amperes                    (8),
        kilovolt-amperes                (9),
        megavolt-amperes                (10),
        volt-amperes-reactive           (11),
        kilovolt-amperes-reactive       (12),
        megavolt-amperes-reactive       (13),
        volts-per-degree-kelvin         (176),
        volts-per-meter                 (177),
        degrees-phase                   (14),
        power-factor                    (15),
        webers                          (178),

--Energy
        ampere-seconds                  (238),
        volt-ampere-hours               (239), -- i.e. VAh
        kilovolt-ampere-hours           (240),
        megavolt-ampere-hours           (241),
        volt-ampere-hours-reactive      (242), -- i.e. varh
        kilovolt-ampere-hours-reactive  (243),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        megavolt-ampere-hours-reactive  (244),
        volt-square-hours               (245),
        ampere-square-hours             (246),
        joules                          (16),
        kilojoules                      (17),
        kilojoules-per-kilogram         (125),
        megajoules                      (126),
        watt-hours                      (18),
        kilowatt-hours                  (19),
        megawatt-hours                  (146),
        watt-hours-reactive             (203),
        kilowatt-hours-reactive         (204),
        megawatt-hours-reactive         (205),
        btus                            (20),
        kilo-btus                       (147),
        mega-btus                       (148),
        therms                          (21),
        ton-hours                       (22),

--Enthalpy
        joules-per-kilogram-dry-air     (23),
        kilojoules-per-kilogram-dry-air (149),
        megajoules-per-kilogram-dry-air (150),
        btus-per-pound-dry-air          (24),
        btus-per-pound                  (117),

--Entropy
        joules-per-degree-kelvin        (127),
        kilojoules-per-degree-kelvin    (151),
        megajoules-per-degree-kelvin    (152),
        joules-per-kilogram-degree-kelvin(128),

-- Force
        newton                          (153),

--Frequency
        cycles-per-hour                 (25),
        cycles-per-minute               (26),
        hertz                           (27),
        kilohertz                       (129),
        megahertz                       (130),
        per-hour                        (131),

--Humidity
        grams-of-water-per-kilogram-dry-air  (28),
        percent-relative-humidity       (29),

--Length
        micrometers                     (194),
        millimeters                     (30),
        centimeters                     (118),
        kilometers                      (193),
        meters                          (31),
        inches                          (32),
        feet                            (33),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
--Light
        candelas                        (179),
        candelas-per-square-meter       (180),
        watts-per-square-foot           (34),
        watts-per-square-meter          (35),
        lumens                          (36),
        luxes                           (37),
        foot-candles                    (38),

--Mass
        milligrams                      (196),
        grams                           (195),
        kilograms                       (39),
        pounds-mass                     (40),
        tons                            (41),

--Mass Flow
        grams-per-second                (154),
        grams-per-minute                (155),
        kilograms-per-second            (42),
        kilograms-per-minute            (43),
        kilograms-per-hour              (44),
        pounds-mass-per-second          (119),
        pounds-mass-per-minute          (45),
        pounds-mass-per-hour            (46),
        tons-per-hour                   (156),

--Power
        milliwatts                      (132),
        watts                           (47),
        kilowatts                       (48),
        megawatts                       (49),
        btus-per-hour                   (50),
        kilo-btus-per-hour              (157),
        joule-per-hours                 (247),
        horsepower                      (51),
        tons-refrigeration              (52),

--Pressure
        pascals                         (53),
        hectopascals                    (133),
        kilopascals                     (54),
        millibars                       (134),
        bars                            (55),
        pounds-force-per-square-inch    (56),
        millimeters-of-water            206),
        centimeters-of-water            (57),
        inches-of-water                 (58),
        millimeters-of-mercury          (59),
        centimeters-of-mercury          (60),
        inches-of-mercury               (61),

--Temperature
        degrees-celsius                 (62),
        degrees-kelvin                  (63),
        degrees-kelvin-per-hour         (181),
        degrees-kelvin-per-minute       (182),
        degrees-fahrenheit              (64),
        degree-days-celsius             (65),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        degree-days-fahrenheit          (66),
        delta-degrees-fahrenheit        (120),
        delta-degrees-kelvin            (121),

--Time
        years                           (67),
        months                          (68),
        weeks                           (69),
        days                            (70),
        hours                           (71),
        minutes                         (72),
        seconds                         (73),
        hundredths-seconds              (158),
        milliseconds                    (159),

--Torque
        newton-meters                   (160),

--Velocity
        millimeters-per-second          (161),
        millimeters-per-minute          (162),
        meters-per-second               (74),
        meters-per-minute               (163),
        meters-per-hour                 (164),
        kilometers-per-hour             (75),
        feet-per-second                 (76),
        feet-per-minute                 (77),
        miles-per-hour                  (78),

--Volume
        cubic-feet                      (79),
        cubic-meters                    (80),
        imperial-gallons                (81),
        milliliters                     (197),
        liters                          (82),
        us-gallons                      (83),

--Volumetric Flow
        cubic-feet-per-second           (142),
        cubic-feet-per-minute           (84),
        million-standard-cubic-feet-per-minute (254),
        cubic-feet-per-hour             (191),
        cubic-feet-per-day              (248),
        standard-cubic-feet-per-day     (47808),
        million-standard-cubic-feet-per-day (47809),
        thousand-cubic-feet-per-day     (47810),
        thousand-standard-cubic-feet-per-day (47811),
        pounds-mass-per-day             (47812),
        cubic-meters-per-second         (85),
        cubic-meters-per-minute         (165),
        cubic-meters-per-hour           (135),
        cubic-meters-per-day            (249),
        imperial-gallons-per-minute     (86),
        milliliters-per-second          (198),
        liters-per-second               (87),
        liters-per-minute               (88),
        liters-per-hour                 (136),
        us-gallons-per-minute           (89),
        us-gallons-per-hour             (192),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
--Other
        degrees-angular                 (90),
        degrees-celsius-per-hour        (91),
        degrees-celsius-per-minute      (92),
        degrees-fahrenheit-per-hour     (93),
        degrees-fahrenheit-per-minute   (94),
        joule-seconds                   (183),
        kilograms-per-cubic-meter       (186),
        kilowatt-hours-per-square-meter (137),
        kilowatt-hours-per-square-foot  (138),
        watt-hours-per-cubic-meter      (250),
        joules-per-cubic-meter          (251),
        megajoules-per-square-meter     (139),
        megajoules-per-square-foot      (140),
        mole-percent                    (252),
        no-units                        (95),
        newton-seconds                  (187),
        newtons-per-meter               188),
        parts-per-million               (96),
        parts-per-billion               (97),
        pascal-seconds                  (253),
        percent                         (98),
        percent-obscuration-per-foot    (143),
        percent-obscuration-per-meter   (144),
        percent-per-second              (99),
        per-minute                      (100),
        per-second                      (101),
        psi-per-degree-fahrenheit       (102),
        radians                         (103),
        radians-per-second              (184),
        revolutions-per-minute          (104),
        square-meters-per-newton        (185),
        watts-per-meter-per-degree-kelvin (189),
        watts-per-square-meter-degree-kelvin (141),
        per-mille                       (207),
        grams-per-gram                  (208),
        kilograms-per-kilogram          (209),
        grams-per-kilogram              (210),
        milligrams-per-gram             (211),
        milligrams-per-kilogram         (212),
        grams-per-milliliter            (213),
        grams-per-liter                 (214),
        milligrams-per-liter            (215),
        micrograms-per-liter            (216),
        grams-per-cubic-meter           (217),
        milligrams-per-cubic-meter      (218),
        micrograms-per-cubic-meter      (219),
        nanograms-per-cubic-meter       (220),
        grams-per-cubic-centimeter      (221),
        becquerels                      (222),
        kilobecquerels                  (223),
        megabecquerels                  (224),
        gray                            (225),
        milligray                       (226),
        microgray                       (227),
        sieverts                        (228),
        millisieverts                   (229),
        microsieverts                   (230),
        microsieverts-per-hour          (231),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        millirems                       (47814),
        millirems-per-hour              (47815),
        decibels-a                      (232),
        nephelometric-turbidity-unit    (233),
        pH                              (234),
        grams-per-square-meter          (235),
        minutes-per-degree-kelvin       (236),
        degrees-lovibond                (47816),
        alcohol-by-volume               (47817),
        international-bittering-units   (47818),
        european-bitterness-units       (47819),
        degrees-plato                   (47820),
        specific-gravity                (47821),
        european-brewing-convention     (47822)
-- Numerical Order Reference
        -- see square-meters                    (0),
        -- see square-feet                     (1),
        -- see milliamperes                    (2),
        -- see amperes                         (3),
        -- see ohms                            (4),
        -- see volts                           (5),
        -- see kilovolts                       (6),
        -- see megavolts                       (7),
        -- see volt-amperes                    (8),
        -- see kilovolt-amperes                (9),
        -- see megavolt-amperes                (10),
        -- see volt-amperes-reactive           (11),
        -- see kilovolt-amperes-reactive       (12),
        -- see megavolt-amperes-reactive       (13),
        -- see degrees-phase                   (14),
        -- see power-factor                    (15),
        -- see joules                          (16),
        -- see kilojoules                      (17),
        -- see watt-hours                      (18),
        -- see kilowatt-hours                  (19),
        -- see btus                            (20),
        -- see therms                          21),
        -- see ton-hours                       (22),
        -- see joules-per-kilogram-dry-air     (23),
        -- see btus-per-pound-dry-air          (24),
        -- see cycles-per-hour                 (25),
        -- see cycles-per-minute               (26),
        -- see hertz                           (27),
        -- see grams-of-water-per-kilogram-dry-air (28),
        -- see percent-relative-humidity       (29),
        -- see millimeters                     (30),
        -- see meters                          (31),
        -- see inches                          (32),
        -- see feet                            (33),
        -- see watts-per-square-foot           (34),
        -- see watts-per-square-meter          (35),
        -- see lumens                          (36),
        -- see luxes                           (37),
        -- see foot-candles                    (38),
        -- see kilograms                       (39),
        -- see pounds-mass                     (40),
        -- see tons                            (41),
        -- see kilograms-per-second            (42),
        -- see kilograms-per-minute            (43),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see kilograms-per-hour       (44),
        -- see pounds-mass-per-minute   (45),
        -- see pounds-mass-per-hour     (46),
        -- see watts                    (47),
        -- see kilowatts                (48),
        -- see megawatts                (49),
        -- see btus-per-hour            (50),
        -- see horsepower               (51),
        -- see tons-refrigeration       (52),
        -- see pascals                  (53),
        -- see kilopascals              (54),
        -- see bars                     (55),
        -- see pounds-force-per-square-inch     (56),
        -- see centimeters-of-water     (57),
        -- see inches-of-water          (58),
        -- see millimeters-of-mercury   (59),
        -- see centimeters-of-mercury   (60),
        -- see inches-of-mercury        (61),
        -- see degrees-celsius          (62),
        -- see degrees-kelvin           (63),
        -- see degrees-fahrenheit       (64),
        -- see degree-days-celsius      (65),
        -- see degree-days-fahrenheit   (66),
        -- see years                    (67),
        -- see months                   (68),
        -- see weeks                    (69),
        -- see days                     (70),
        -- see hours                    (71),
        -- see minutes                  (72),
        -- see seconds                  (73),
        -- see meters-per-second        (74),
        -- see kilometers-per-hour      (75),
        -- see feet-per-second          (76),
        -- see feet-per-minute          (77),
        -- see miles-per-hour           (78),
        -- see cubic-feet               (79),
        -- see cubic-meters             (80),
        -- see imperial-gallons         (81),
        -- see liters                   (82),
        -- see us-gallons               (83),
        -- see cubic-feet-per-minute    (84),
        -- see cubic-meters-per-second  (85),
        -- see imperial-gallons-per-minute  (86),
        -- see liters-per-second        (87),
        -- see liters-per-minute        (88),
        -- see us-gallons-per-minute    (89),
        -- see degrees-angular          (90),
        -- see degrees-celsius-per-hour (91),
        -- see degrees-celsius-per-minute   (92),
        -- see degrees-fahrenheit-per-hour      (93),
        -- see degrees-fahrenheit-per-minute    (94),
        -- see no-units                 (95),
        -- see parts-per-million        (96),
        -- see parts-per-billion        (97),
        -- see percent                  (98),
        -- see percent-per-second       (99),
        -- see per-minute               (100),
        -- see per-second               (101),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see psi-per-degree-fahrenheit        (102),
        -- see radians                          (103),
        -- see revolutions-per-minute           (104),
        -- see currency1                        (105),
        -- see currency2                        (106),
        -- see currency3                        (107),
        -- see currency4                        (108),
        -- see currency5                        (109),
        -- see currency6                        (110),
        -- see currency7                        (111),
        -- see currency8                        (112),
        -- see currency9                        (113),
        -- see currency10                       (114),
        -- see square-inches                    (115),
        -- see square-centimeters               (116),
        -- see btus-per-pound                   (117),
        -- see centimeters                      (118),
        -- see pounds-mass-per-second           (119),
        -- see delta-degrees-fahrenheit         (120),
        -- see delta-degrees-kelvin             (121),
        -- see kilohms                          (122),
        -- see megohms                          (123),
        -- see millivolts                       (124),
        -- see kilojoules-per-kilogram          (125),
        -- see megajoules                       (126),
        -- see joules-per-degree-kelvin         (127),
        -- see joules-per-kilogram-degree-kelvin (128),
        -- see kilohertz                        (129),
        -- see megahertz                        (130),
        -- see per-hour                         (131),
        -- see milliwatts                       (132),
        -- see hectopascals                     (133),
        -- see millibars                        (134),
        -- see cubic-meters-per-hour            (135),
        -- see liters-per-hour                  (136),
        -- see kilowatt-hours-per-square-meter  (137),
        -- see kilowatt-hours-per-square-foot   (138),
        -- see megajoules-per-square-meter      (139),
        -- see megajoules-per-square-foot       (140),
        -- see watts-per-square-meter-degree-kelvin (141),
        -- see cubic-feet-per-second            (142),
        -- see percent-obscuration-per-foot     (143),
        -- see percent-obscuration-per-meter    (144),
        -- see milliohms                        (145),
        -- see megawatt-hours                   (146),
        -- see kilo-btus                        (147),
        -- see mega-btus                        (148),
        -- see kilojoules-per-kilogram-dry-air  (149),
        -- see megajoules-per-kilogram-dry-air  (150),
        -- see kilojoules-per-degree-kelvin     (151),
        -- see megajoules-per-degree-kelvin     (152),
        -- see newton                           (153),
        -- see grams-per-second                 (154),
        -- see grams-per-minute                 (155),
        -- see tons-per-hour                    (156),
        -- see kilo-btus-per-hour               (157),
        -- see hundredths-seconds               (158),
        -- see milliseconds                     (159),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- see newton-meters                    (160),
-- see millimeters-per-second           (161),
-- see millimeters-per-minute           (162),
-- see meters-per-minute                (163),
-- see meters-per-hour                  (164),
-- see cubic-meters-per-minute          (165),
-- see meters-per-second-per-second     (166),
-- see amperes-per-meter                (167),
-- see amperes-per-square-meter         (168),
-- see ampere-square-meters             (169),
-- see farads                           (170),
-- see henrys                           (171),
-- see ohm-meters                       (172),
-- see siemens                          (173),
-- see siemens-per-meter                (174),
-- see teslas                           (175),
-- see volts-per-degree-kelvin          (176),
-- see volts-per-meter                  (177),
-- see webers                           (178),
-- see candelas                         (179),
-- see candelas-per-square-meter        (180),
-- see degrees-kelvin-per-hour          (181),
-- see degrees-kelvin-per-minute        (182),
-- see joule-seconds                    (183),
-- see radians-per-second               (184),
-- see square-meters-per-newton         (185),
-- see kilograms-per-cubic-meter        (186),
-- see newton-seconds                   (187),
-- see newtons-per-meter                (188),
-- see watts-per-meter-per-degree-kelvin (189),
-- see micro-siemens                    (190),
-- see cubic-feet-per-hour              (191),
-- see us-gallons-per-hour              (192),
-- see kilometers                       (193),
-- see micrometers                      (194),
-- see grams                            (195),
-- see milligrams                       (196),
-- see milliliters                      (197),
-- see milliliters-per-second           (198),
-- see decibels                         (199),
-- see decibels-millivolt               (200),
-- see decibels-volt                    (201),
-- see millisiemens                     (202),
-- see watt-hours-reactive              (203),
-- see kilowatt-hours-reactive          (204),
-- see megawatt-hours-reactive          (205),
-- see millimeters-of-water             (206),
-- see per-mille                        (207),
-- see grams-per-gram                   (208),
-- see kilograms-per-kilogram           (209),
-- see grams-per-kilogram               (210),
-- see milligrams-per-gram              (211),
-- see milligrams-per-kilogram          (212),
-- see grams-per-milliliter             (213),
-- see grams-per-liter                  (214),
-- see milligrams-per-liter             (215),
-- see micrograms-per-liter             (216),
-- see grams-per-cubic-meter            (217),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see milligrams-per-cubic-meter       (218),
        -- see micrograms-per-cubic-meter       (219),
        -- see nanograms-per-cubic-meter        (220),
        -- see grams-per-cubic-centimeter       (221),
        -- see becquerels                       (222),
        -- see kilobecquerels                   (223),
        -- see megabecquerels                   (224),
        -- see gray                             (225),
        -- see milligray                        (226),
        -- see microgray                        (227),
        -- see sieverts                         (228),
        -- see millisieverts                    (229),
        -- see microsieverts                    (230),
        -- see microsieverts-per-hour           (231),
        -- see decibels-a                       (232),
        -- see nephelometric-turbidity-unit     (233),
        -- see pH                               (234),
        -- see grams-per-square-meter           (235),
        -- see minutes-per-degree-kelvin        (236),
        -- see ohm-meter-squared-per-meter      (237),
        -- see ampere-seconds                   (238),
        -- see volt-ampere-hours                (239),
        -- see kilovolt-ampere-hours            (240),
        -- see megavolt-ampere-hours            (241),
        -- see volt-ampere-hours-reactive       (242),
        -- see kilovolt-ampere-hours-reactive   (243),
        -- see megavolt-ampere-hours-reactive   (244),
        -- see volt-square-hours                (245),
        -- see ampere-square-hours              (246),
        -- see joule-per-hours                  (247),
        -- see cubic-feet-per-day               (248),
        -- see cubic-meters-per-day             (249),
        -- see watt-hours-per-cubic-meter       (250),
        -- see joules-per-cubic-meter           (251),
        -- see mole-percent                     (252),
        -- see pascal-seconds                   (253),
        -- see million-standard-cubic-feet-per-minute   (254),
        -- see standard-cubic-feet-per-day      (47808),
        -- see million-standard-cubic-feet-per-day      (47809),
        -- see thousand-cubic-feet-per-day      (47810),
        -- see thousand-standard-cubic-feet-per-day  (47811),
        -- see pounds-mass-per-day              (47812),
        -- see millirems                        (47814),
        -- see millirems-per-hour               (47815),
        -- see degrees-lovibond                 (47816),
        -- see alcohol-by-volume                (47817),
        -- see international-bittering-units    (47818),
        -- see european-bitterness-units        (47819),
        -- see degrees-plato                    (47820),
        -- see specific-gravity                 (47821),
        -- see european-brewing-convention      (47822)
    ...
    }
    -- Enumerated values 0-255 and 47808-49999 are reserved for definition by ASHRAE. Enumerated values
    -- 256-47807 and 50000-65535 may be used by others subject to the procedures and constraints described
    -- in Clause 23.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetEscalatorFault ::= ENUMERATED {
        controller-fault                (0),
        drive-and-motor-fault           (1),
        mechanical-component-fault      (2),
        overspeed-fault                 (3),
        power-supply-fault              (4),
        safety-device-fault             (5),
        controller-supply-fault         (6),
        drive-temperature-exceeded      (7),
        comb-plate-fault                (8),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetEscalatorMode ::= ENUMERATED {
        unknown                         (0),
        stop                            (1),
        up                              (2),
        down                            (3),
        inspection                      (4),
        out-of-service                  (5),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetEscalatorOperationDirection ::= ENUMERATED {
        unknown                         (0),
        stopped                         (1),
        up-rated-speed                  (2),
        up-reduced-speed                (3),
        down-rated-speed                (4),
        down-reduced-speed              (5),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetEventLogRecord ::= SEQUENCE {
        timestamp                       [0] BACnetDateTime,
        log-datum                       [1] CHOICE {
                                                log-status      [0] BACnetLogStatus,
                                                notification    [1] ConfirmedEventNotification-Request,
                                                time-change     [2] REAL
                                                }
        }

BACnetEventNotificationSubscription ::= SEQUENCE {
        recipient                       [0] BACnetRecipient,
        process-identifier              [1] Unsigned32,
        issue-confirmed-notifications   [2] BOOLEAN,
        time-remaining                  [3] Unsigned
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnetEventParameter ::= CHOICE {

-- These choices have a one-to-one correspondence with the Event_Type enumeration with the exception of the
-- complex-event-type, which is used for proprietary event types.

```
        change-of-bitstring     [0] SEQUENCE {
                                    time-delay                  [0] Unsigned,
                                    bitmask                     [1] BIT STRING,
                                    list-of-bitstring-values    [2] SEQUENCE OF BIT STRING
                                    },
        change-of-state         [1] SEQUENCE {
                                    time-delay          [0] Unsigned,
                                    list-of-values      [1] SEQUENCE OF BACnetPropertyStates
                                    },
        change-of-value         [2] SEQUENCE {
                                    time-delay                  [0] Unsigned,
                                    cov-criteria                [1] CHOICE {
                                            bitmask                             [0] BIT STRING,
                                            referenced-property-increment       [1] REAL
                                            }
                                    },
        command-failure             [3] SEQUENCE {
                                    time-delay                      [0] Unsigned,
                                    feedback-property-reference     [1]
BACnetDeviceObjectPropertyReference
                                    },
        floating-limit          [4] SEQUENCE {
                                    time-delay                  [0] Unsigned,
                                    setpoint-reference          [1] BACnetDeviceObjectPropertyReference,
                                    low-diff-limit              [2] REAL,
                                    high-diff-limit             [3] REAL,
                                    deadband                    [4] REAL
                                    },
        out-of-range            [5] SEQUENCE {
                                    time-delay          [0] Unsigned,
                                    low-limit           [1] REAL,
                                    high-limit          [2] REAL,
                                    deadband            [3] REAL
                                    },
        -- context tag 7 is deprecated
        change-of-life-safety   [8] SEQUENCE {
                                    time-delay                      [0] Unsigned,
                                    list-of-life-safety-alarm-values    [1] SEQUENCE OF BACnetLifeSafetyState,
                                    list-of-alarm-values                [2] SEQUENCE OF BACnetLifeSafetyState,
                                    mode-property-reference             [3]
BACnetDeviceObjectPropertyReference
                                    },
        extended                [9] SEQUENCE {
                                    vendor-id               [0] Unsigned16,
                                    extended-event-type     [1] Unsigned,
                                    parameters              [2] SEQUENCE OF CHOICE {
                                            null                    NULL,
                                            real                    REAL,
                                            unsigned                Unsigned,
                                            boolean                 BOOLEAN,
                                            integer                 INTEGER,
                                            double                  Double,
                                            octetstring             OCTET STRING,
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                                        characterstring         CharacterString,
                                        bitstring                       BIT STRING,
                                        enumerated              ENUMERATED,
                                        date                    Date,
                                        time                    Time,
                                        objectidentifier        BACnetObjectIdentifier,
                                        reference               [0]
BACnetDeviceObjectPropertyReference
                                        }
                                },
        buffer-ready            [10] SEQUENCE {
                                        notification-threshold          [0] Unsigned,
                                        previous-notification-count     [1] Unsigned32
                                        },
        unsigned-range          [11] SEQUENCE {
                                        time-delay      [0] Unsigned,
                                        low-limit       [1] Unsigned,
                                        high-limit      [2] Unsigned
                                        },
        -- context tag 12 is reserved for future addenda
        access-event            [13] SEQUENCE {
                                        list-of-access-events           [0] SEQUENCE OF BACnetAccessEvent,
                                        access-event-time-reference     [1]
BACnetDeviceObjectPropertyReference
                                        },
        double-out-of-range     [14] SEQUENCE {
                                        time-delay      [0] Unsigned,
                                        low-limit       [1] Double,
                                        high-limit      [2] Double,
                                        deadband        [3] Double
                                        },
        signed-out-of-range     [15] SEQUENCE {
                                        time-delay      [0] Unsigned,
                                        low-limit       [1] INTEGER,
                                        high-limit      [2] INTEGER,
                                        deadband        [3] Unsigned
                                        },
        unsigned-out-of-range   [16] SEQUENCE {
                                        time-delay      [0] Unsigned,
                                        low-limit       [1] Unsigned,
                                        high-limit      [2] Unsigned,
                                        deadband        [3] Unsigned
                                        },
        change-of-characterstring [17] SEQUENCE {
                                        time-delay              [0] Unsigned,
                                        list-of-alarm-values    [1] SEQUENCE OF CharacterString
                                        },
        change-of-status-flags  [18] SEQUENCE {
                                        time-delay      [0] Unsigned,
                                        selected-flags  [1] BACnetStatusFlags
                                        },
        -- context tag [19] is not used, see note below
        None                    [20] NULL,
        change-of-discrete-value [21] SEQUENCE {
                                        time-delay      [0] Unsigned
                                        },
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        change-of-timer         [22] SEQUENCE {
                                        time-delay              [0] Unsigned,
                                        alarm-values            [1] SEQUENCE OF BACnetTimerState,
                                        update-time-reference   [2] BACnetDeviceObjectPropertyReference
                                        }
        }

-- CHOICE [6] has been intentionally omitted. It parallels the complex-event-type CHOICE [6] of the
-- BACnetNotificationParameters production which was introduced to allow the addition of proprietary event
-- algorithms whose event parameters are not necessarily network-visible.
-- CHOICE [19] has been intentionally omitted. It parallels the change-of-reliability event type CHOICE[19]
-- of the BACnetNotificationParameters production which was introduced for the notification of event state
-- changes to FAULT and from FAULT, which does not have event parameters.

BACnetEventState ::= ENUMERATED {
        normal                  (0),
        fault                   (1),
        offnormal               (2),
        high-limit              (3),
        low-limit               (4),
        life-safety-alarm       (5),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23. The last enumeration used in this version is 5.

BACnetEventTransitionBits ::= BIT STRING {
        to-offnormal            (0),
        to-fault                (1),
        to-normal               (2)
        }

BACnetEventType ::= ENUMERATED {
        change-of-bitstring     (0),
        change-of-state         (1),
        change-of-value         (2),
        command-failure         (3),
        floating-limit          (4),
        out-of-range            (5),
        -- complex-event-type   (6),  see comment below
        -- context tag 7 is deprecated
        change-of-life-safety   (8),
        extended                (9),
        buffer-ready            (10),
        unsigned-range          (11),
        -- enumeration value 12 is reserved for future addenda
        access-event            (13),
        double-out-of-range     (14),
        signed-out-of-range     (15),
        unsigned-out-of-range   (16),
        change-of-characterstring (17),
        change-of-status-flags  (18),
        change-of-reliability   (19),
        none                    (20),
        change-of-discrete-value (21),
        change-of-timer         (22),
        ...
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23. It is expected that these enumerated values will correspond to the use of the
        -- complex-event-type CHOICE [6] of the BACnetNotificationParameters production.

BACnetFaultParameter ::= CHOICE {
        none                    [0] NULL,
        fault-characterstring   [1] SEQUENCE {
                                        list-of-fault-values        [0] SEQUENCE OF CharacterString
                                        },
        fault-extended          [2] SEQUENCE {
                                        vendor-id                   [0] Unsigned16,
                                        extended-fault-type         [1] Unsigned,
                                        parameters                  [2] SEQUENCE OF CHOICE {
                                                null                        NULL,
                                                real                        REAL,
                                                unsigned                    Unsigned,
                                                boolean                     BOOLEAN,
                                                integer                     INTEGER,
                                                double                      Double,
                                                octetstring                 OCTET STRING,
                                                characterstring             CharacterString,
                                                bitstring                           BIT STRING,
                                                enumerated                  ENUMERATED,
                                                date                        Date,
                                                time                        Time,
                                                objectidentifier            BACnetObjectIdentifier,
                                                reference                   [0]
BACnetDeviceObjectPropertyReference
                                                }
                                        },
        fault-life-safety       [3] SEQUENCE {
                                        list-of-fault-values        [0] SEQUENCE OF BACnetLifeSafetyState,
                                        mode-property-reference     [1]
BACnetDeviceObjectPropertyReference
                                        },
        fault-state             [4] SEQUENCE {
                                        list-of-fault-values        [0] SEQUENCE OF BACnetPropertyStates
                                        },
        fault-status-flags      [5] SEQUENCE {
                                        status-flags-reference      [0]  BACnetDeviceObjectPropertyReference
                                        },
        fault-out-of-range              [6] SEQUENCE {
                                        min-normal-value            [0]  CHOICE {
                                                real                        REAL,
                                                unsigned                    Unsigned,
                                                double                      Double,
                                                integer                     INTEGER
                                                },
                                        max-normal-value            [1]  CHOICE {
                                                real                        REAL,
                                                unsigned                    Unsigned,
                                                double                      Double,
                                                integer                     INTEGER
                                                }
                                        },
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        fault-listed            [7] SEQUENCE {
                                        fault-list-reference    [0] BACnetDeviceObjectPropertyReference
                                        }
        }

BACnetFaultType ::= ENUMERATED {
        none                    (0),
        fault-characterstring   (1),
        fault-extended          (2),
        fault-life-safety       (3),
        fault-state             (4),
        fault-status-flags      (5),
        fault-out-of-range      (6),
        fault-listed            (7)
        }

BACnetFDTEntry ::= SEQUENCE {
        bacnetip-address        [0] OCTET STRING,       -- the 6-octet B/IP or 18-octet B/IPv6 address of
the registrant
        time-to-live            [1] Unsigned16,         -- time to live in seconds at the time of registration
        remaining-time-to-live  [2] Unsigned16          -- remaining time to live in seconds, incl. grace period
        }

BACnetFileAccessMethod ::= ENUMERATED {
        record-access           (0),
        stream-access           (1)
        }

BACnetGroupChannelValue ::= SEQUENCE {
        channel                 [0] Unsigned16,
        overriding-priority     [1] Unsigned (1..16) OPTIONAL,
        value                   BACnetChannelValue
        }

BACnetHostAddress ::= CHOICE {
        none                    [0] NULL,
        ip-address              [1] OCTET STRING,       -- 4 octets for B/IP or 16 octets for B/IPv6
        name                    [2] CharacterString     -- Internet host name (see RFC 1123)
        }

BACnetHostNPort ::= SEQUENCE {
        host                    [0] BACnetHostAddress,
        port                    [1] Unsigned16
        }

BACnetIPMode ::= ENUMERATED {
        normal                  (0),
        foreign                 (1),
        bbmd                    (2)
        }

BACnetLandingCallStatus ::= SEQUENCE {
        floor-number    [0] Unsigned8,
        command         CHOICE {
                                direction               [1] BACnetLiftCarDirection,
                                destination     [2] Unsigned8
                                },
        floor-text      [3] CharacterString OPTIONAL
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetLandingDoorStatus ::= SEQUENCE {
        landing-doors    [0] SEQUENCE OF SEQUENCE {
                                 floor-number     [0] Unsigned8,
                                 door-status      [1] BACnetDoorStatus
                                 }
        }

BACnetLifeSafetyMode ::= ENUMERATED {
        off                              (0),
        on                               (1),
        test                             (2),
        manned                           (3),
        unmanned                         (4),
        armed                            (5),
        disarmed                         (6),
        prearmed                         (7),
        slow                             (8),
        fast                             (9),
        disconnected                     (10),
        enabled                          (11),
        disabled                         (12),
        automatic-release-disabled       (13),
        default                          (14),
        activated-oeo-alarm              (15),
        activated-oeo-evacuate           (16),
        activated-oeo-phase1-recall      (17),
        activated-oeo-unavailable        (18),
        deactivated                      (19),
        ...
        }
        -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
        -- 256-65535 may be used by others subject to procedures and constraints described in Clause 23.

BACnetLifeSafetyOperation ::= ENUMERATED {
        none                             (0),
        silence                          (1),
        silence-audible                  (2),
        silence-visual                   (3),
        reset                            (4),
        reset-alarm                      (5),
        reset-fault                      (6),
        unsilence                        (7),
        unsilence-audible                (8),
        unsilence-visual                 (9),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to procedures and constraints described in
        -- Clause 23.

BACnetLifeSafetyOperationInfo ::= SEQUENCE {
        requesting-process-identifier    [0] Unsigned32,
        request                          [1] BACnetLifeSafetyOperation
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetLifeSafetyState ::= ENUMERATED {
        quiet                           (0),
        pre-alarm                       (1),
        alarm                           (2),
        fault                           (3),
        fault-pre-alarm                 (4),
        fault-alarm                     (5),
        not-ready                       (6),
        active                          (7),
        tamper                          (8),
        test-alarm                      (9),
        test-active                     (10),
        test-fault                      (11),
        test-fault-alarm                (12),
        holdup                          (13),
        duress                          (14),
        tamper-alarm                    (15),
        abnormal                        (16),
        emergency-power                 (17),
        delayed                         (18),
        blocked                         (19),
        local-alarm                     (20),
        general-alarm                   (21),
        supervisory                     (22),
        test-supervisory                (23),
        non-default-mode                (24),
        oeo-unavailable                 (25),
        oeo-alarm                       (26),
        oeo-phase1-recall               (27),
        oeo-evacuate                    (28),
        oeo-unaffected                  (29),
        test-oeo-unavailable            (30),
        test-oeo-alarm                  (31),
        test-oeo-phase1-recall          (32),
        test-oeo-evacuate               (33),
        test-oeo-unaffected             (34),
        ...
        }
        -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
        -- 256-65535 may be used by others subject to procedures and constraints described in Clause 23.

BACnetLiftCarCallList ::= SEQUENCE {
        floor-numbers                   [0] SEQUENCE OF Unsigned8
        }

BACnetLiftCarDirection ::= ENUMERATED {
        unknown                         (0),
        none                            (1),
        stopped                         (2),
        up                              (3),
        down                            (4),
        up-and-down                     (5),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetLiftCarDoorCommand ::= ENUMERATED {
        none                            (0),
        open                            (1),
        close                           (2)
        }

BACnetLiftCarDriveStatus ::= ENUMERATED {
        unknown                         (0),
        stationary                      (1),
        braking                         (2),
        accelerate                      (3),
        decelerate                      (4),
        rated-speed                     (5),
        single-floor-jump               (6),
        two-floor-jump                  (7),
        three-floor-jump                (8),
        multi-floor-jump                (9),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetLiftCarMode ::= ENUMERATED {
        unknown                         (0),
        normal                          (1), -- in service
        vip                             (2),
        homing                          (3),
        parking                         (4),
        attendant-control               (5),
        firefighter-control             (6),
        emergency-power                 (7),
        inspection                      (8),
        cabinet-recall                  (9),
        earthquake-operation            (10),
        fire-operation                  (11),
        out-of-service                  (12),
        occupant-evacuation             (13),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetLiftFault ::= ENUMERATED {
        controller-fault                (0),
        drive-and-motor-fault           (1),
        governor-and-safety-gear-fault  (2),
        lift-shaft-device-fault         (3),
        power-supply-fault              (4),
        safety-interlock-fault          (5),
        door-closing-fault              (6),
        door-opening-fault              (7),
        car-stopped-outside-landing-zone (8),
        call-button-stuck               (9),
        start-failure                   (10),
        controller-supply-fault         (11),
        self-test-failure               (12),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        runtime-limit-exceeded          (13),
        position-lost                   (14),
        drive-temperature-exceeded      (15),
        load-measurement-fault          (16),
        ...
        }
-- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values
-- 1024-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetLiftGroupMode ::= ENUMERATED {
        unknown                         (0),
        normal                          (1),
        down-peak                       (2),
        two-way                         (3),
        four-way                        (4),
        emergency-power                 (5),
        up-peak                         (6)
        }

BACnetLightingCommand ::= SEQUENCE {
        operation                       [0] BACnetLightingOperation,
        target-level                    [1] REAL (0.0..100.0) OPTIONAL,
        ramp-rate                       [2] REAL (0.1..100.0)  OPTIONAL,
        step-increment                  [3] REAL (0.1..100.0)  OPTIONAL,
        fade-time                       [4] Unsigned (100..86400000)  OPTIONAL,
        priority                        [5] Unsigned (1..16) OPTIONAL
        }
        -- Note that the combination of level, ramp-rate, step-increment, and fade-time fields is
        -- dependent on the specific lighting operation. See Table 12-67.

BACnetLightingInProgress ::= ENUMERATED {
        idle                            (0),
        fade-active                     (1),
        ramp-active                     (2),
        not-controlled                  (3),
        other                           (4)
        }

BACnetLightingOperation ::= ENUMERATED {
        none                            (0),
        fade-to                         (1),
        ramp-to                         (2),
        step-up                         (3),
        step-down                       (4),
        step-on                         (5),
        step-off                        (6),
        warn                            (7),
        warn-off                        (8),
        warn-relinquish                 (9),
        stop                            (10),
        ...
        }
        -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values 256-65535 may be used
by
        -- others subject to the procedures and constraints described in Clause 23.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetLightingTransition ::= ENUMERATED {
        none                            (0),
        fade                            (1),
        ramp                            (2),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-255 may be used by
        -- others subject to the procedures and constraints described in Clause 23.

BACnetLimitEnable ::= BIT STRING {
        low-limit-enable                (0),
        high-limit-enable               (1)
        }

BACnetLockStatus ::= ENUMERATED {
        locked                          (0),
        unlocked                        (1),
        lock-fault                      (2),
        unused                          (3),
        unknown                         (4)
        }

BACnetLogData ::=  CHOICE {
        log-status          [0] BACnetLogStatus,
        log-data            [1] SEQUENCE OF CHOICE {
                                    boolean-value       [0] BOOLEAN,
                                    real-value          [1] REAL,
                                    enumerated-value            [2] ENUMERATED,         -- Optionally limited to 32
bits
                                    unsigned-value      [3] Unsigned,           -- Optionally limited to 32 bits
                                    integer-value       [4] INTEGER,            -- Optionally limited to 32 bits
                                    bitstring-value     [5] BIT STRING,         -- Optionally limited to 32 bits
                                    null-value          [6] NULL,
                                    failure             [7] Error,
                                    any-value           [8] ABSTRACT-SYNTAX.&Type  -- Optional
                                    },
        time-change         [2] REAL
        }

BACnetLoggingType ::= ENUMERATED {
        polled                          (0),
        cov                             (1),
        triggered                       (2),
        ...
        }
-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-255 may be used by others subject to the procedures and constraints described
-- in Clause 23.

BACnetLogMultipleRecord ::=  SEQUENCE {
        timestamp                       [0] BACnetDateTime,
        log-data                        [1] BACnetLogData
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetLogRecord ::=  SEQUENCE {
        timestamp       [0] BACnetDateTime,
        log-datum       [1] CHOICE {
                                log-status              [0] BACnetLogStatus,
                                boolean-value           [1] BOOLEAN,
                                real-value              [2] REAL,
                                enumerated-value                [3] ENUMERATED,         -- Optionally limited to 32
bits
                                unsigned-value          [4] Unsigned,           -- Optionally limited to 32 bits
                                integer-value           [5] INTEGER,            -- Optionally limited to 32 bits
                                bitstring-value         [6] BIT STRING,         -- Optionally limited to 32 bits
                                null-value              [7] NULL,
                                failure                 [8] Error,
                                time-change             [9] REAL,
                                any-value               [10] ABSTRACT-SYNTAX.&Type  -- Optional
                        },
        status-flags    [2] BACnetStatusFlags OPTIONAL
        }

BACnetLogStatus ::= BIT STRING {
        log-disabled                    (0),
        buffer-purged                   (1),
        log-interrupted                 (2)
        }

BACnetMaintenance ::= ENUMERATED {
        none                            (0),
        periodic-test                   (1),
        need-service-operational        (2),
        need-service-inoperative        (3),
        ...
        }
        -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
        -- 256-65535 may be used by others subject to procedures and constraints described in
        -- Clause 23.

BACnetNameValue ::= SEQUENCE {
        name                            [0] CharacterString,
        value                           ABSTRACT-SYNTAX.&Type OPTIONAL
                                        -- value is limited to primitive datatypes and BACnetDateTime
        }

BACnetNameValueCollection ::= SEQUENCE {
        members                         [0]    SEQUENCE OF BACnetNameValue
        }

BACnetNetworkNumberQuality ::= ENUMERATED {
        unknown                         (0),
        learned                         (1),
        learned-configured              (2),
        configured                      (3)
        }

BACnetNetworkPortCommand ::= ENUMERATED {
        idle                            (0),
        discard-changes                 (1),
        renew-fd-registration           (2),
        restart-slave-discovery         (3),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        renew-dhcp                  (4),
        restart-autonegotiation     (5),
        disconnect                  (6),
        restart-port                (7),
        ...
        }
        -- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values
        -- 128-255 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetNetworkType ::= ENUMERATED {
        ethernet                    (0),
        arcnet                      (1),
        mstp                        (2),
        ptp                         (3),
        lontalk                     (4),
        ipv4                        (5),
        zigbee                      (6),
        virtual                     (7),
        -- formerly: non-bacnet     (8), removed in version 1 revision 18
        ipv6                        (9),
        serial                      (10),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-255 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetNodeType ::= ENUMERATED {
        unknown                     (0),
        system                      (1),
        network                     (2),
        device                      (3),
        organizational              (4),
        area                        (5),
        equipment                   (6),
        point                       (7),
        collection                  (8),
        property                    (9),
        functional                  (10),
        other                       (11),
        subsystem                   (12),
        building                    (13),
        floor                       (14),
        section                     (15),
        module                      (16),
        tree                        (17),
        member                      (18),
        protocol                    (19),
        room                        (20),
        zone                        (21)
        }

BACnetNotificationParameters ::= CHOICE {
-- These choices have a one-to-one correspondence with the Event_Type enumeration with the exception of the
-- complex-event-type, which is used for proprietary event types.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

```
change-of-bitstring     [0] SEQUENCE {
                            referenced-bitstring    [0] BIT STRING,
                            status-flags            [1] BACnetStatusFlags
                            },
change-of-state         [1] SEQUENCE {
                            new-state               [0] BACnetPropertyStates,
                            status-flags            [1] BACnetStatusFlags
                            },
change-of-value         [2] SEQUENCE {
                            new-value               [0] CHOICE {
                                                            changed-bits    [0] BIT STRING,
                                                            changed-value   [1] REAL
                                                        },
                            status-flags            [1] BACnetStatusFlags
                            },
command-failure             [3] SEQUENCE {
                            command-value           [0] ABSTRACT-SYNTAX.&Type,
                                                    -- depends on ref property
                            status-flags            [1] BACnetStatusFlags,
                            feedback-value          [2] ABSTRACT-SYNTAX.&Type
                                                    -- depends on ref property
                            },
floating-limit          [4] SEQUENCE {
                            reference-value         [0] REAL,
                            status-flags            [1] BACnetStatusFlags,
                            setpoint-value          [2] REAL,
                            error-limit             [3] REAL
                            },
out-of-range            [5] SEQUENCE {
                            exceeding-value         [0] REAL,
                            status-flags            [1] BACnetStatusFlags,
                            deadband                [2] REAL,
                            exceeded-limit          [3] REAL
                            },
complex-event-type      [6] SEQUENCE OF BACnetPropertyValue,
-- complex tag 7 is deprecated
change-of-life-safety   [8] SEQUENCE {
                            new-state               [0] BACnetLifeSafetyState,
                            new-mode                [1] BACnetLifeSafetyMode,
                            status-flags            [2] BACnetStatusFlags,
                            operation-expected      [3] BACnetLifeSafetyOperation
                            },
extended                [9] SEQUENCE {
                            vendor-id               [0] Unsigned16,
                            extended-event-type     [1] Unsigned,
                            parameters              [2] SEQUENCE OF CHOICE {
                                                        null            NULL,
                                                        real            REAL,
                                                        unsigned        Unsigned,
                                                        boolean         BOOLEAN,
                                                        integer         INTEGER,
                                                        double          Double,
                                                        octetstring     OCTET STRING,
                                                        characterstring CharacterString,
                                                        bitstring       BIT STRING,
                                                        enumerated      ENUMERATED,
                                                        date            Date,
                                                        time            Time,
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                                        objectidentifier  BACnetObjectIdentifier,
                                        property-value    [0] BACnetDeviceObjectPropertyValue
                                        }
                                },
buffer-ready            [10] SEQUENCE {
                                buffer-property         [0] BACnetDeviceObjectPropertyReference,
                                previous-notification   [1] Unsigned32,
                                current-notification    [2] Unsigned32
                                },
unsigned-range          [11] SEQUENCE {
                                exceeding-value         [0] Unsigned,
                                status-flags            [1] BACnetStatusFlags,
                                exceeded-limit          [2] Unsigned
                                },
-- context tag 12 is reserved for future addenda
access-event            [13] SEQUENCE {
                                access-event            [0] BACnetAccessEvent,
                                status-flags            [1] BACnetStatusFlags,
                                access-event-tag        [2] Unsigned,
                                access-event-time               [3] BACnetTimeStamp,
                                access-credential               [4] BACnetDeviceObjectReference,
                                authentication-factor   [5] BACnetAuthenticationFactor OPTIONAL
                                },
double-out-of-range     [14] SEQUENCE {
                                exceeding-value         [0] Double,
                                status-flags            [1] BACnetStatusFlags,
                                deadband                [2] Double,
                                exceeded-limit          [3] Double
                                },
signed-out-of-range     [15] SEQUENCE {
                                exceeding-value         [0] INTEGER,
                                status-flags            [1] BACnetStatusFlags,
                                deadband                [2] Unsigned,
                                exceeded-limit          [3] INTEGER
                                },
unsigned-out-of-range   [16] SEQUENCE {
                                exceeding-value         [0] Unsigned,
                                status-flags            [1] BACnetStatusFlags,
                                deadband                [2] Unsigned,
                                exceeded-limit          [3] Unsigned
                                },
change-of-characterstring       [17] SEQUENCE {
                                changed-value           [0] CharacterString,
                                status-flags            [1] BACnetStatusFlags,
                                alarm-value             [2] CharacterString
                                },
change-of-status-flags  [18] SEQUENCE {
                                present-value           [0] ABSTRACT-SYNTAX.&Type OPTIONAL,
                                                        -- depends on referenced property
                                referenced-flags        [1] BACnetStatusFlags
                                },
change-of-reliability   [19] SEQUENCE {
                                reliability             [0] BACnetReliability,
                                status-flags            [1] BACnetStatusFlags,
                                property-values         [2] SEQUENCE OF BACnetPropertyValue
                                },
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- context tag [20] is not used, see note below
        change-of-discrete-value [21] SEQUENCE {
                new-value               [0] CHOICE {
                        boolean             BOOLEAN,
                        unsigned            Unsigned,
                        integer             INTEGER,
                        enumerated          ENUMERATED,
                        characterstring     CharacterString,
                        octetstring         OCTET STRING,
                        date                Date,
                        time                Time,
                        objectidentifier    BACnetObjectIdentifier,
                        datetime                [0]
BACnetDateTime
                },
                status-flags            [1] BACnetStatusFlags
                },
        change-of-timer         [22] SEQUENCE {
                new-state               [0] BACnetTimerState,
                status-flags            [1] BACnetStatusFlags,
                update-time             [2] BACnetDateTime,
                last-state-change           [3] BACnetTimerTransition OPTIONAL,
                initial-timeout         [4] Unsigned OPTIONAL,
                expiration-time         [5] BACnetDateTime OPTIONAL
                }
        }
            -- CHOICE [20] has been intentionally omitted. It parallels the 'none' event type CHOICE[20] of
            -- the BACnetEventParameter production which was introduced for the case an object
            -- does not apply an event algorithm

BACnetNotifyType ::= ENUMERATED {
        alarm                   (0),
        event                   (1),
        ack-notification        (2)
        }

BACnetObjectPropertyReference ::= SEQUENCE {
        object-identifier       [0] BACnetObjectIdentifier,
        property-identifier     [1] BACnetPropertyIdentifier,
        property-array-index    [2] Unsigned OPTIONAL -- used only with array datatype
                                    -- if omitted with an array the entire array is referenced
        }

BACnetObjectPropertyValue ::= SEQUENCE {
        Object-identifier       [0] BACnetObjectIdentifier,
        property-identifier     [1] BACnetPropertyIdentifier,
        property-array-index    [2] Unsigned OPTIONAL, -- used only with array datatype
                                    -- if omitted with an array the entire array is referenced
        property-value          [3] ABSTRACT-SYNTAX.&Type, --any datatype appropriate for the specified property
        priority                [4] Unsigned (1..16) OPTIONAL
        }

BACnetObjectSelector ::= CHOICE {
        none                    NULL,
        object                  BACnetObjectIdentifier,
        object-type             BACnetObjectType
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnetObjectType ::= ENUMERATED { -- see below for numerical order

access-credential	(32),
access-door	(30),
access-point	(33),
access-rights	(34),
access-user	(35),
access-zone	(36),
accumulator	(23),
alert-enrollment	(52),
analog-input	(0),
analog-output	(1),
analog-value	(2),
audit-log	(61),
audit-reporter	(62),
averaging	(18),
binary-input	(3),
binary-lighting-output	(55),
binary-output	(4),
binary-value	(5),
bitstring-value	(39),
calendar	(6),
channel	(53),
characterstring-value	(40),
command	(7),
credential-data-input	(37),
datepattern-value	(41),
date-value	(42),
datetimepattern-value	(43),
datetime-value	(44),
device	(8),
elevator-group	(57),
escalator	(58),
event-enrollment	(9),
event-log	(25),
file	(10),
global-group	(26),
group	(11),
integer-value	(45),
large-analog-value	(46),
life-safety-point	(21),
life-safety-zone	(22),
lift	(59),
lighting-output	(54),
load-control	(28),
loop	(12),
multi-state-input	(13),
multi-state-output	(14),
multi-state-value	(19),
network-port	(56),
notification-class	(15),
notification-forwarder	(51),
octetstring-value	(47),
positive-integer-value	(48),
program	(16),
pulse-converter	(24),
schedule	(17),
staging	(60),
structured-view	(29),
timepattern-value	(49),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        time-value                  (50),
        timer                       (31),
        trend-log                   (20),
        trend-log-multiple          (27),
-- numerical order reference
        -- see analog-input         (0),
        -- see analog-output        (1),
        -- see analog-value         (2),
        -- see binary-input         (3),
        -- see binary-output        (4),
        -- see binary-value         (5),
        -- see calendar             (6),
        -- see command              (7),
        -- see device               (8),
        -- see event-enrollment     (9),
        -- see file                 (10),
        -- see group                (11),
        -- see loop                 (12),
        -- see multi-state-input    (13),
        -- see multi-state-output   (14),
        -- see notification-class   (15),
        -- see program              (16),
        -- see schedule             (17),
        -- see averaging            (18),
        -- see multi-state-value    (19),
        -- see trend-log            (20),
        -- see life-safety-point    (21),
        -- see life-safety-zone     (22),
        -- see accumulator          (23),
        -- see pulse-converter      (24),
        -- see event-log            (25),
        -- see global-group         (26),
        -- see trend-log-multiple   (27),
        -- see load-control         (28),
        -- see structured-view      (29),
        -- see access-door          (30),
        -- see timer                (31),
        -- see access-credential    (32),
        -- see access-point         (33),
        -- see access-rights        (34),
        -- see access-user          (35),
        -- see access-zone          (36),
        -- see credential-data-input (37),
        --                          (38), removed
        -- see bitstring-value      (39),
        -- see characterstring-value (40),
        -- see datepattern-value    (41),
        -- see date-value           (42),
        -- see datetimepattern-value (43),
        -- see datetime-value       (44),
        -- see integer-value        (45),
        -- see large-analog-value   (46),
        -- see octetstring-value    (47),
        -- see positive-integer-value (48),
        -- see timepattern-value    (49),
        -- see time-value           (50),
        -- see notification-forwarder (51),
        -- see alert-enrollment     (52),
        -- see channel              (53),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see lighting-output          (54),
        -- see binary-lighting-output   (55),
        -- see network-port             (56),
        -- see elevator-group           (57),
        -- see escalator                (58),
        -- see lift                     (59),
        -- see staging                  (60),
        -- see audit-log                (61),
        -- see audit-reporter           (62)
        ...
        }
        -- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values
        -- 128-1023 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetObjectTypesSupported ::= BIT STRING {
        analog-input                (0),
        analog-output               (1),
        analog-value                (2),
        binary-input                (3),
        binary-output               (4),
        binary-value                (5),
        calendar                    (6),
        command                     (7),
        device                      (8),
        event-enrollment            (9),
        file                        (10),
        group                       (11),
        loop                        (12),
        multi-state-input           (13),
        multi-state-output          (14),
        notification-class          (15),
        program                     (16),
        schedule                    (17),
        averaging                   (18),
        multi-state-value           (19),
        trend-log                   (20),
        life-safety-point           (21),
        life-safety-zone            (22),
        accumulator                 (23),
        pulse-converter             (24),
        event-log                   (25),
        global-group                (26),
        trend-log-multiple          (27),
        load-control                (28),
        structured-view             (29),
        access-door                 (30),
        timer                       (31),
        access-credential           (32),
        access-point                (33),
        access-rights               (34),
        access-user                 (35),
        access-zone                 (36),
        credential-data-input       (37),
        --                          (38), removed
        bitstring-value             (39),
        characterstring-value       (40),
        datepattern-value           (41),
        date-value                  (42),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        datetimepattern-value       (43),
        datetime-value              (44),
        integer-value               (45),
        large-analog-value          (46),
        octetstring-value           (47),
        positive-integer-value      (48),
        timepattern-value           (49),
        time-value                  (50),
        notification-forwarder      (51),
        alert-enrollment            (52),
        channel                     (53),
        lighting-output             (54),
        binary-lighting-output      (55),
        network-port                (56),
        elevator-group              (57),
        escalator                   (58),
        lift                        (59),
        staging                     (60),
        audit-log                   (61),
        audit-reporter              (62)
        }

BACnetOptionalBinaryPV ::= CHOICE {
        null                        NULL,
        binary-pv                   BACnetBinaryPV
        }

BACnetOptionalCharacterString ::= CHOICE {
        null                        NULL,
        characterstring             CharacterString
        }

BACnetOptionalPriorityFilter ::= CHOICE {
        default                     NULL,
        filter                      BACnetPriorityFilter
        }

BACnetOptionalREAL ::= CHOICE {
        null                        NULL,
        real                        REAL
        }

BACnetOptionalUnsigned ::= CHOICE {
        null                        NULL,
        unsigned                    Unsigned
        }

BACnetPolarity ::= ENUMERATED {
        normal                      (0),
        reverse                     (1)
        }

BACnetPortPermission ::= SEQUENCE {
        port-id                     [0] Unsigned8,
        enabled                     [1] BOOLEAN
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetPrescale ::= SEQUENCE {
        multiplier                  [0] Unsigned,
        modulo-divide               [1] Unsigned
        }

BACnetPriorityArray ::= SEQUENCE SIZE (16) OF BACnetPriorityValue
        -- accessed as a BACnetARRAY

BACnetPriorityFilter ::= BIT STRING {
        manual-life-safety          (0),
        automatic-life-safety       (1),
        priority-3                  (2),
        priority-4                  (3),
        critical-equipment-controls (4),
        minimum-on-off              (5),
        priority-7                  (6),
        manual-operator             (7),
        priority-9                  (8),
        priority-10                 (9),
        priority-11                 (10),
        priority-12                 (11),
        priority-13                 (12),
        priority-14                 (13),
        priority-15                 (14),
        priority-16                 (15)
        }

BACnetPriorityValue ::= CHOICE {
        null                        NULL,
        real                        REAL,
        enumerated                  ENUMERATED,
        unsigned                    Unsigned,
        boolean                     BOOLEAN,
        integer                     INTEGER,
        double                      Double,
        time                        Time,
        characterstring             CharacterString,
        octetstring                 OCTET STRING,
        bitstring                   BIT STRING,
        date                        Date,
        objectidentifier            BACnetObjectIdentifier,
        constructed-value           [0] ABSTRACT-SYNTAX.&Type,
        datetime                    [1]    BACnetDateTime
        }

BACnetProcessIdSelection ::= CHOICE {
        process-identifier          Unsigned32,
        null-value                  NULL
        }

BACnetProgramError ::= ENUMERATED {
        normal                      (0),
        load-failed                 (1),
        internal                    (2),
        program                     (3),
        other                       (4),
        ...
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetProgramRequest ::= ENUMERATED {
        ready                   (0),
        load                    (1),
        run                     (2),
        halt                    (3),
        restart                 (4),
        unload                  (5)
        }

BACnetProgramState ::= ENUMERATED {
        idle                    (0),
        loading                 (1),
        running                 (2),
        waiting                 (3),
        halted                  (4),
        unloading               (5)
        }

BACnetPropertyAccessResult ::= SEQUENCE {
        object-identifier       [0] BACnetObjectIdentifier,
        property-identifier     [1] BACnetPropertyIdentifier,
        property-array-index    [2] Unsigned OPTIONAL, -- used only with array datatype
                                                       -- if omitted with an array then
                                                       -- the entire array is referenced
        device-identifier       [3] BACnetObjectIdentifier OPTIONAL,
        access-result           CHOICE {
                                        property-value          [4] ABSTRACT-SYNTAX.&Type,
                                        property-access-error   [5] Error
                                        }
        }

BACnetPropertyIdentifier ::= ENUMERATED {  -- see below for numerical order
        absentee-limit                          (244),
        accepted-modes                          (175),
        access-alarm-events                     (245),
        access-doors                            (246),
        access-event                            (247),
        access-event-authentication-factor      (248),
        access-event-credential                 (249),
        access-event-tag                        (322),
        access-event-time                       (250),
        access-transaction-events               (251),
        accompaniment                           (252),
        accompaniment-time                      (253),
        ack-required                            (1),
        acked-transitions                       (0),
        action                                  (2),
        action-text                             (3),
        activation-time                         (254),
        active-authentication-policy            (255),
        active-cov-multiple-subscriptions       (481),
        active-cov-subscriptions                (152),
        active-text                             (4),
        active-vt-sessions                      (5),
        actual-shed-level                       (212),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

adjust-value (176),
alarm-value (6),
alarm-values (7),
align-intervals (193),
all (8),
all-writes-successful (9),
allow-group-delay-inhibit (365),
apdu-length (399),
apdu-segment-timeout (10),
apdu-timeout (11),
application-software-version (12),
archive (13),
assigned-access-rights (256),
assigned-landing-calls (447),
attempted-samples (124),
audit-source-reporter (497),
audit-level (498),
audit-notification-recipient (499),
auditable-priority-filter (500),
auditable-operations (501),
authentication-factors (257),
authentication-policy-list (258),
authentication-policy-names (259),
authentication-status (260),
authorization-exemptions (364),
authorization-mode (261),
auto-slave-discovery (169),
average-value (125),
backup-and-restore-state (338),
backup-failure-timeout (153),
backup-preparation-time (339),
bacnet-ip-global-address (407),
bacnet-ip-mode (408),
bacnet-ip-multicast-address (409),
bacnet-ip-nat-traversal (410),
bacnet-ip-udp-port (412),
bacnet-ipv6-mode (435),
bacnet-ipv6-udp-port (438),
bacnet-ipv6-multicast-address (440),
bbmd-accept-fd-registrations (413),
bbmd-broadcast-distribution-table (414),
bbmd-foreign-device-table (415),
belongs-to (262),
bias (14),
bit-mask (342),
bit-text (343),
blink-warn-enable (373),
buffer-size (126),
car-assigned-direction (448),
car-door-command (449),
car-door-status (450),
car-door-text (451),
car-door-zone (452),
car-drive-status (453),
car-load (454),
car-load-units (455),
car-mode (456),
car-moving-direction (457),
car-position (458),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

change-of-state-count (15),
change-of-state-time (16),
changes-pending (416),
channel-number (366),
client-cov-increment (127),
command (417),
command-time-array (430),
configuration-files (154),
control-groups (367),
controlled-variable-reference (19),
controlled-variable-units (20),
controlled-variable-value (21),
count (177),
count-before-change (178),
count-change-time (179),
cov-increment (22),
cov-period (180),
cov-resubscription-interval (128),
covu-period (349),
covu-recipients (350),
credential-disable (263),
credential-status (264),
credentials (265),
credentials-in-zone (266),
current-command-priority (431),
database-revision (155),
date-list (23),
daylight-savings-status (24),
days-remaining (267),
deadband (25),
default-fade-time (374),
default-present-value (492),
default-ramp-rate (375),
default-step-increment (376),
default-subordinate-relationship (490),
default-timeout (393),
delete-on-forward (502),
deployed-profile-location (484),
derivative-constant (26),
derivative-constant-units (27),
description (28),
description-of-halt (29),
device-address-binding (30),
device-type (31),
device-uuid (507),
direct-reading (156),
door-alarm-state (226),
door-extended-pulse-time (227),
door-members (228),
door-open-too-long-time (229),
door-pulse-time (230),
door-status (231),
door-unlock-delay-time (232),
duty-window (213),
effective-period (32),
egress-active (386),
egress-time (377),
elapsed-active-time (33),
elevator-group (459),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
enable                          (133),    -- renamed from previous version
energy-meter                    (460),
energy-meter-ref                (461),
entry-points                    (268),
error-limit                     (34),
escalator-mode                  (462),
event-algorithm-inhibit         (354),
event-algorithm-inhibit-ref     (355),
event-detection-enable          (353),
event-enable                    (35),
event-message-texts             (351),
event-message-texts-config      (352),
event-parameters                (83),
event-state                     (36),
event-time-stamps               (130),
event-type                      (37),
exception-schedule              (38),
execution-delay                 (368),
exit-points                     (269),
expected-shed-level             (214),
expiration-time                 (270),
extended-time-enable            (271),
failed-attempt-events           (272),
failed-attempts                 (273),
failed-attempts-time            (274),
fault-high-limit                (388),
fault-low-limit                 (389),
fault-parameters                (358),
fault-signals                   (463),
fault-type                      (359),
fault-values                    (39),
fd-bbmd-address                 (418),
fd-subscription-lifetime        (419),
feedback-value                  (40),
file-access-method              (41),
file-size                       (42),
file-type                       (43),
firmware-revision               (44),
floor-number                    (506),
floor-text                      (464),
full-duty-baseline              (215),
global-identifier               (323),
group-id                        (465),
group-member-names              (346),
group-members                   (345),
group-mode                      (467),
high-limit                      (45),
higher-deck                     (468),
in-process                      (47),
in-progress                     (378),
inactive-text                   (46),
initial-timeout                 (394),
input-reference                 (181),
installation-id                 (469),
instance-of                     (48),
instantaneous-power             (379),
integral-constant               (49),
integral-constant-units         (50),
interface-value                 (387),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

interval-offset (195),
ip-address (400),
ip-default-gateway (401),
ip-dhcp-enable (402),
ip-dhcp-lease-time (403),
ip-dhcp-lease-time-remaining (404),
ip-dhcp-server (405),
ip-dns-server (406),
ip-subnet-mask (411),
ipv6-address (436),
ipv6-auto-addressing-enable (442),
ipv6-default-gateway (439),
ipv6-dhcp-lease-time (443),
ipv6-dhcp-lease-time-remaining (444),
ipv6-dhcp-server (445),
ipv6-dns-server (441),
ipv6-prefix-length (437),
ipv6-zone-index (446),
issue-confirmed-notifications (51),
is-utc (344),
landing-call-control (471),
landing-calls (470),
landing-door-status (472),
last-access-event (275),
last-access-point (276),
last-command-time (432),
last-credential-added (277),
last-credential-added-time (278),
last-credential-removed (279),
last-credential-removed-time (280),
last-notify-record (173),
last-priority (369),
last-restart-reason (196),
last-restore-time (157),
last-state-change (395),
last-use-time (281),
life-safety-alarm-values (166),
lighting-command (380),
lighting-command-default-priority (381),
limit-enable (52),
limit-monitoring-interval (182),
link-speed (420),
link-speed-autonegotiate (422),
link-speeds (421),
list-of-group-members (53),
list-of-object-property-references (54),
local-date (56),
local-forwarding-only (360),
local-time (57),
location (58),
lock-status (233),
lockout (282),
lockout-relinquish-time (283),
log-buffer (131),
log-device-object-property (132),
log-interval (134),
logging-object (183),
logging-record (184),
logging-type (197),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        low-diff-limit                      (390),
        low-limit                           (59),
        lower-deck                          (473),
        mac-address                         (423),
        machine-room-id                     (474),
        maintenance-required                (158),
        making-car-call                     (475),
        manipulated-variable-reference      (60),
        manual-slave-address-binding        (170),
        masked-alarm-values                 (234),
        max-actual-value                    (382),
        max-apdu-length-accepted            (62),
        max-failed-attempts                 (285),
        max-info-frames                     (63),
        max-master                          (64),
        max-pres-value                      (65),
        max-segments-accepted               (167),
        maximum-output                      (61),
        maximum-send-delay                  (503),
        maximum-value                       (135),
        maximum-value-timestamp             (149),
        member-of                           (159),
        member-status-flags                 (347),
        members                             (286),
        min-actual-value                    (383),
        min-pres-value                      (69),
        minimum-off-time                    (66),
        minimum-on-time                     (67),
        minimum-output                      (68),
        minimum-value                       (136),
        minimum-value-timestamp             (150),
        mode                                (160),
        model-name                          (70),
        modification-date                   (71),
        monitored-objects                   (504),
        muster-point                        (287),
        negative-access-rules               (288),
        network-interface-name              (424),
        network-number                      (425),
        network-number-quality              (426),
        network-type                        (427),
        next-stopping-floor                 (476),
        node-subtype                        (207),
        node-type                           (208),
        notification-class                  (17),  -- renamed from previous version
        notification-threshold              (137),
        notify-type                         (72),
        number-of-apdu-retries              (73),
        number-of-authentication-policies   (289),
        number-of-states                    (74),
        object-identifier                   (75),
        object-list                         (76),
        object-name                         (77),
        object-property-reference           (78),
        object-type                         (79),
        occupancy-count                     (290),
        occupancy-count-adjust              (291),
        occupancy-count-enable              (292),
        occupancy-lower-limit               (294),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

occupancy-lower-limit-enforced (295),
occupancy-state (296),
occupancy-upper-limit (297),
occupancy-upper-limit-enforced (298),
operation-direction (477),
operation-expected (161),
optional (80),
out-of-service (81),
output-units (82),
passback-mode (300),
passback-timeout (301),
passenger-alarm (478),
polarity (84),
port-filter (363),
positive-access-rules (302),
power (384),
power-mode (479),
prescale (185),
present-stage (493),
present-value (85),
priority (86),
priority-array (87),
priority-for-writing (88),
process-identifier (89),
process-identifier-filter (361),
profile-location (485),
profile-name (168),
program-change (90),
program-location (91),
program-state (92),
property-list (371),
proportional-constant (93),
proportional-constant-units (94),
protocol-level (482),
protocol-object-types-supported (96),
protocol-revision (139),
protocol-services-supported (97),
protocol-version (98),
pulse-rate (186),
read-only (99),
reason-for-disable (303),
reason-for-halt (100),
recipient-list (102),
record-count (141),
records-since-notification (140),
reference-port (483),
registered-car-call (480),
reliability (103),
reliability-evaluation-inhibit (357),
relinquish-default (104),
represents (491),
requested-shed-level (218),
requested-update-interval (348),
required (105),
resolution (106),
restart-notification-recipients (202),
restore-completion-time (340),
restore-preparation-time (341),
routing-table (428),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

scale (187),
scale-factor (188),
schedule-default (174),
secured-status (235),
segmentation-supported (107),
send-now (505),
serial-number (372),
setpoint (108),
setpoint-reference (109),
setting (162),
shed-duration (219),
shed-level-descriptions (220),
shed-levels (221),
silenced (163),
slave-address-binding (171),
slave-proxy-enable (172),
stages (494),
stage-names (495),
start-time (142),
state-change-values (396),
state-description (222),
state-text (110),
status-flags (111),
stop-time (143),
stop-when-full (144),
strike-count (391),
structured-object-list (209),
subordinate-annotations (210),
subordinate-list (211),
subordinate-node-types (487),
subordinate-relationships (489),
subordinate-tags (488),
subscribed-recipients (362),
supported-format-classes (305),
supported-formats (304),
system-status (112),
tags (486),
target-references (496),
threat-authority (306),
threat-level (307),
time-delay (113),
time-delay-normal (356),
time-of-active-time-reset (114),
time-of-device-restart (203),
time-of-state-count-reset (115),
time-of-strike-count-reset (392),
time-synchronization-interval (204),
time-synchronization-recipients (116),
timer-running (397),
timer-state (398),
total-record-count (145),
trace-flag (308),
tracking-value (164),
transaction-notification-class (309),
transition (385),
trigger (205),
units (117),
update-interval (118),
update-time (189),

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        user-external-identifier                (310),
        user-information-reference              (311),
        user-name                               (317),
        user-type                               (318),
        uses-remaining                          (319),
        utc-offset                              (119),
        utc-time-synchronization-recipients     (206),
        valid-samples                           (146),
        value-before-change                     (190),
        value-change-time                       (192),
        value-set                               (191),
        value-source                            (433),
        value-source-array                      (434),
        variance-value                          (151),
        vendor-identifier                       (120),
        vendor-name                             (121),
        verification-time                       (326),
        virtual-mac-address-table               (429),
        vt-classes-supported                    (122),
        weekly-schedule                         (123),
        window-interval                         (147),
        window-samples                          (148),
        write-status                            (370),
        zone-from                               (320),
        zone-members                            (165),
        zone-to                                 (321),
-- -numerical order reference
        -- see acked-transitions                (0),
        -- see ack-required                     (1),
        -- see action                           (2),
        -- see action-text                      (3),
        -- see active-text                      (4),
        -- see active-vt-sessions               (5),
        -- see alarm-value                      (6),
        -- see alarm-values                     (7),
        -- see all                              (8),
        -- see all-writes-successful            (9),
        -- see apdu-segment-timeout             (10),
        -- see apdu-timeout                     (11),
        -- see application-software-version     (12),
        -- see archive                          (13),
        -- see bias                             (14),
        -- see change-of-state-count            (15),
        -- see change-of-state-time             (16),
        -- see notification-class               (17),
        -- this property  deleted               (18),
        -- see controlled-variable-reference    (19),
        -- see controlled-variable-units        (20),
        -- see controlled-variable-value        (21),
        -- see cov-increment                    (22),
        -- see date-list                        (23),
        -- see daylight-savings-status          (24),
        -- see deadband                         (25),
        -- see derivative-constant              (26),
        -- see derivative-constant-units        (27),
        -- see description                      (28),
        -- see description-of-halt              (29),
        -- see device-address-binding           (30),
        -- see device-type                      (31),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see effective-period             (32),
        -- see elapsed-active-time          (33),
        -- see error-limit                  (34),
        -- see event-enable                 (35),
        -- see event-state                  (36),
        -- see event-type                   (37),
        -- see exception-schedule           (38),
        -- see fault-values                 (39),
        -- see feedback-value               (40),
        -- see file-access-method           (41),
        -- see file-size                    (42),
        -- see file-type                    (43),
        -- see firmware-revision            (44),
        -- see high-limit                   (45),
        -- see inactive-text                (46),
        -- see in-process                   (47),
        -- see instance-of                  (48),
        -- see integral-constant            (49),
        -- see integral-constant-units      (50),
        -- see issue-confirmed-notifications (51), removed in version 1 revision 4, added back in version 1 revision 20
        -- see limit-enable                 (52),
        -- see list-of-group-members        (53),
        -- see list-of-object-property-references (54),
        -- formerly: list-of-session-keys   (55), removed in version 1 revision 11
        -- see local-date                   (56),
        -- see local-time                   (57),
        -- see location                     (58),
        -- see low-limit                    (59),
        -- see manipulated-variable-reference (60),
        -- see maximum-output               (61),
        -- see max-apdu-length-accepted     (62),
        -- see max-info-frames              (63),
        -- see max-master                   (64),
        -- see max-pres-value               (65),
        -- see minimum-off-time             (66),
        -- see minimum-on-time              (67),
        -- see minimum-output               (68),
        -- see min-pres-value               (69),
        -- see model-name                   (70),
        -- see modification-date            (71),
        -- see notify-type                  (72),
        -- see number-of-apdu-retries       (73),
        -- see number-of-states             (74),
        -- see object identifier            (75),
        -- see object-list                  (76),
        -- see object-name                  (77),
        -- see object-property-reference    (78),
        -- see object-type                  (79),
        -- see optional                     (80),
        -- see out-of-service               (81),
        -- see output-units                 (82),
        -- see event-parameters             (83),
        -- see polarity                     (84),
        -- see present value                (85),
        -- see priority                     (86),
        -- see priority-array               (87),
        -- see priority-for-writing         (88),
        -- see process-identifier           (89),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see program-change                    (90),
        -- see program-location                  (91),
        -- see program-state                     (92),
        -- see proportional-constant             (93),
        -- see proportional-constant-units       (94),
        -- formerly: protocol-conformance-class  (95), removed in version 1 revision 2
        -- see protocol-object-types-supported   (96),
        -- see protocol-services-supported       (97),
        -- see protocol-version                  (98),
        -- see read-only                         (99),
        -- see reason-for-halt                   (100),
        -- formerly: recipient                   (101), removed in version 1 revision 4
        -- see recipient-list                    (102),
        -- see reliability                       (103),
        -- see relinquish-default                (104),
        -- see required                          (105),
        -- see resolution                        (106),
        -- see segmentation-supported            (107),
        -- see setpoint                          (108),
        -- see setpoint-reference                (109),
        -- see state-text                        (110),
        -- see status-flags                      (111),
        -- see system-status                     (112),
        -- see time-delay                        (113),
        -- see time-of-active-time-reset         (114),
        -- see time-of-state-count-reset         (115),
        -- see time-synchronization-recipients   (116),
        -- see units                             (117),
        -- see update-interval                   (118),
        -- see utc-offset                        (119),
        -- see vendor-identifier                 (120),
        -- see vendor-name                       (121),
        -- see vt-classes-supported              (122),
        -- see weekly-schedule                   (123),
        -- see attempted-samples                 (124),
        -- see average-value                     (125),
        -- see buffer-size                       (126),
        -- see client-cov-increment              (127),
        -- see cov-resubscription-interval       (128),
        -- formerly: current-notify-time         (129), removed in version 1 revision 3
        -- see event-time-stamps                 (130),
        -- see log-buffer                        (131),
        -- see log-device-object-property        (132),
        -- see enable                            (133), log-enable was renamed to enable in version 1 revision 5
        -- see log-interval                      (134),
        -- see maximum-value                     (135),
        -- see minimum-value                     (136),
        -- see notification-threshold            (137),
        -- formerly: previous-notify-time        (138), removed in version 1 revision 3
        -- see protocol-revision                 (139),
        -- see records-since-notification        (140),
        -- see record-count                      (141),
        -- see start-time                        (142),
        -- see stop-time                         (143),
        -- see stop-when-full                    (144),
        -- see total-record-count                (145),
        -- see valid-samples                     (146),
        -- see window-interval                   (147),
        -- see window-samples                    (148),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see maximum-value-timestamp      (149),
        -- see minimum-value-timestamp      (150),
        -- see variance-value               (151),
        -- see active-cov-subscriptions     (152),
        -- see backup-failure-timeout       (153),
        -- see configuration-files          (154),
        -- see database-revision            (155),
        -- see direct-reading               (156),
        -- see last-restore-time,           (157),
        -- see maintenance-required         (158),
        -- see member-of                    (159),
        -- see mode                         (160),
        -- see operation-expected           (161),
        -- see setting                      (162),
        -- see silenced                     (163),
        -- see tracking-value               (164),
        -- see zone-members                 (165),
        -- see life-safety-alarm-values     (166),
        -- see max-segments-accepted        (167),
        -- see profile-name                 (168),
        -- see auto-slave-discovery         (169),
        -- see manual-slave-address-binding (170),
        -- see slave-address-binding        (171),
        -- see slave-proxy-enable           (172),
        -- see last-notify-record           (173),
        -- see schedule-default             (174),
        -- see accepted-modes               (175),
        -- see adjust-value                 (176),
        -- see count                        (177),
        -- see count-before-change          (178),
        -- see count-change-time            (179),
        -- see cov-period                   (180),
        -- see input-reference              (181),
        -- see limit-monitoring-interval    (182),
        -- see logging-object               (183),
        -- see logging-record               (184),
        -- see prescale                     (185),
        -- see pulse-rate                   (186),
        -- see scale                        (187),
        -- see scale-factor                 (188),
        -- see update-time                  (189),
        -- see value-before-change          (190),
        -- see value-set                    (191),
        -- see value-change-time            (192),
        -- see align-intervals              (193),
        -- enumeration value 194 is unassigned
        -- see interval-offset              (195),
        -- see last-restart-reason          (196),
        -- see logging-type                 (197),
        -- enumeration values 198-201 are unassigned
        -- see restart-notification-recipients (202),
        -- see time-of-device-restart       (203),
        -- see time-synchronization-interval (204),
        -- see trigger                      (205),
        -- see utc-time-synchronization-recipients (206),
        -- see node-subtype                 (207),
        -- see node-type                    (208),
        -- see structured-object-list       (209),
        -- see subordinate-annotations      (210),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see subordinate-list                 (211),
        -- see actual-shed-level                (212),
        -- see duty-window                      (213),
        -- see expected-shed-level              (214),
        -- see full-duty-baseline               (215),
        -- enumeration values 216-217 are unassigned
        -- see requested-shed-level             (218),
        -- see shed-duration                    (219),
        -- see shed-level-descriptions          (220),
        -- see shed-levels                      (221),
        -- see state-description                (222),
        -- enumeration values 223-225 are unassigned
        -- see door-alarm-state                 (226),
        -- see door-extended-pulse-time         (227),
        -- see door-members                     (228),
        -- see door-open-too-long-time          (229),
        -- see door-pulse-time                  (230),
        -- see door-status                      (231),
        -- see door-unlock-delay-time           (232),
        -- see lock-status                      (233),
        -- see masked-alarm-values              (234),
        -- see secured-status                   (235),
        -- enumeration values 236-243 are unassigned
        -- see absentee-limit                   (244),
        -- see access-alarm-events              (245),
        -- see access-doors                     (246),
        -- see access-event                     (247),
        -- see access-event-authentication-factor       (248),
        -- see access-event-credential          (249),
        -- see access-event-time                (250),
        -- see access-transaction-events        (251),
        -- see accompaniment                    (252),
        -- see accompaniment-time               (253),
        -- see activation-time                  (254),
        -- see active-authentication-policy     (255),
        -- see assigned-access-rights           (256),
        -- see authentication-factors           (257),
        -- see authentication-policy-list       (258),
        -- see authentication-policy-names              (259),
        -- see authentication-status            (260),
        -- see authorization-mode               (261),
        -- see belongs-to                       (262),
        -- see credential-disable               (263),
        -- see credential-status                (264),
        -- see credentials                      (265),
        -- see credentials-in-zone              (266),
        -- see days-remaining                   (267),
        -- see entry-points                     (268),
        -- see exit-points                      (269),
        -- see expiration-time                  (270),
        -- see extended-time-enable             (271),
        -- see failed-attempt-events            (272),
        -- see failed-attempts                  (273),
        -- see failed-attempts-time             (274),
        -- see last-access-event                (275),
        -- see last-access-point                (276),
        -- see last-credential-added            (277),
        -- see last-credential-added-time       (278),
        -- see last-credential-removed          (279),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- see last-credential-removed-time          (280),
-- see last-use-time                         (281),
-- see lockout                               (282),
-- see lockout-relinquish-time               (283),
-- formerly: master-exemption                (284), removed in version 1 revision 13
-- see max-failed-attempts                   (285),
-- see members                               (286),
-- see muster-point                          (287),
-- see negative-access-rules                 (288),
-- see number-of-authentication-policies     (289),
-- see occupancy-count                       (290),
-- see occupancy-count-adjust                (291),
-- see occupancy-count-enable                (292),
-- formerly: occupancy-exemption             (293), removed in version 1 revision 13
-- see occupancy-lower-limit                 (294),
-- see occupancy-lower-limit-enforced        (295),
-- see occupancy-state                       (296),
-- see occupancy-upper-limit                 (297),
-- see occupancy-upper-limit-enforced        (298),
-- formerly: passback-exemption              (299), removed in version 1 revision 13
-- see passback-mode                         (300),
-- see passback-timeout                      (301),
-- see positive-access-rules                 (302),
-- see reason-for-disable                    (303),
-- see supported-formats                     (304),
-- see supported-format-classes              (305),
-- see threat-authority                      (306),
-- see threat-level                          (307),
-- see trace-flag                            (308),
-- see transaction-notification-class        (309),
-- see user-external-identifier              (310),
-- see user-information-reference            (311),
-- enumeration values 312-316 are unassigned
-- see user-name                             (317),
-- see user-type                             (318),
-- see uses-remaining                        (319),
-- see zone-from                             (320),
-- see zone-to                               (321),
-- see access-event-tag                      (322),
-- see global-identifier                     (323),
-- enumeration values 324-325 are unassigned
-- see verification-time                     (326),
-- enumeration values 327-337 are removed
-- see backup-and-restore-state              (338),
-- see backup-preparation-time               (339),
-- see restore-completion-time               (340),
-- see restore-preparation-time              (341),
-- see bit-mask                              (342),
-- see bit-text                              (343),
-- see is-utc                                (344),
-- see group-members                         (345),
-- see group-member-names                    (346),
-- see member-status-flags                   (347),
-- see requested-update-interval             (348),
-- see covu-period                           (349),
-- see covu-recipients                       (350),
-- see event-message-texts                   (351),
-- see event-message-texts-config            (352),
-- see event-detection-enable                (353),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see event-algorithm-inhibit          (354),
        -- see event-algorithm-inhibit-ref      (355),
        -- see time-delay-normal                (356),
        -- see reliability-evaluation-inhibit   (357),
        -- see fault-parameters                 (358),
        -- see fault-type                       (359),
        -- see local-forwarding-only            (360),
        -- see process-identifier-filter        (361),
        -- see subscribed-recipients            (362),
        -- see port-filter                      (363),
        -- see authorization-exemptions         (364),
        -- see allow-group-delay-inhibit        (365),
        -- see channel-number                   (366),
        -- see control-groups                   (367),
        -- see execution-delay                  (368),
        -- see last-priority                    (369),
        -- see write-status                     (370),
        -- see property-list                    (371),
        -- see serial-number                    (372),
        -- see blink-warn-enable                (373),
        -- see default-fade-time                (374),
        -- see default-ramp-rate                (375),
        -- see default-step-increment           (376),
        -- see egress-time                      (377),
        -- see in-progress                      (378),
        -- see instantaneous-power              (379),
        -- see lighting-command                 (380),
        -- see lighting-command-default-priority    (381),
        -- see max-actual-value                 (382),
        -- see min-actual-value                 (383),
        -- see power                            (384),
        -- see transition                       (385),
        -- see egress-active                    (386),
        -- see interface-value                  (387),
        -- see fault-high-limit                 (388),
        -- see fault-low-limit                  (389),
        -- see low-diff-limit                   (390),
        -- see strike-count                     (391),
        -- see time-of-strike-count-reset       (392),
        -- see default-timeout                  (393),
        -- see initial-timeout                  (394),
        -- see last-state-change                (395),
        -- see state-change-values              (396),
        -- see timer-running                    (397),
        -- see timer-state                      (398),
        -- see apdu-length                      (399),
        -- see ip-address                       (400),
        -- see ip-default-gateway               (401),
        -- see ip-dhcp-enable                   (402),
        -- see ip-dhcp-lease-time               (403),
        -- see ip-dhcp-lease-time-remaining     (404),
        -- see ip-dhcp-server                   (405),
        -- see ip-dns-server                    (406),
        -- see bacnet-ip-global-address         (407),
        -- see bacnet-ip-mode                   (408),
        -- see bacnet-ip-multicast-address      (409),
        -- see bacnet-ip-nat-traversal          (410),
        -- see ip-subnet-mask                   (411),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see bacnet-ip-udp-port               (412),
        -- see bbmd-accept-fd-registrations     (413),
        -- see bbmd-broadcast-distribution-table (414),
        -- see bbmd-foreign-device-table        (415),
        -- see changes-pending                  (416),
        -- see command                          (417),
        -- see fd-bbmd-address                  (418),
        -- see fd-subscription-lifetime         (419),
        -- see link-speed                       (420),
        -- see link-speeds                      (421),
        -- see link-speed-autonegotiate         (422),
        -- see mac-address                      (423),
        -- see network-interface-name           (424),
        -- see network-number                   (425),
        -- see network-number-quality           (426),
        -- see network-type                     (427),
        -- see routing-table                    (428),
        -- see virtual-mac-address-table        (429),
        -- see command-time-array               (430),
        -- see current-command-priority         (431),
        -- see last-command-time                (432),
        -- see value-source                     (433),
        -- see value-source-array               (434),
        -- see bacnet-ipv6-mode                 (435),
        -- see ipv6-address                     (436),
        -- see ipv6-prefix-length               (437),
        -- see bacnet-ipv6-udp-port             (438),
        -- see ipv6-default-gateway             (439),
        -- see bacnet-ipv6-multicast-address    (440),
        -- see ipv6-dns-server                  (441),
        -- see ipv6-auto-addressing-enable      (442),
        -- see ipv6-dhcp-lease-time             (443),
        -- see ipv6-dhcp-lease-time-remaining   (444),
        -- see ipv6-dhcp-server                 (445),
        -- see ipv6-zone-index                  (446),
        -- see assigned-landing-calls           (447),
        -- see car-assigned-direction           (448),
        -- see car-door-command                 (449),
        -- see car-door-status                  (450),
        -- see car-door-text                    (451),
        -- see car-door-zone                    (452),
        -- see car-drive-status                 (453),
        -- see car-load                         (454),
        -- see car-load-units                   (455),
        -- see car-mode                         (456),
        -- see car-moving-direction             (457),
        -- see car-position                     (458),
        -- see elevator-group                   (459),
        -- see energy-meter                     (460),
        -- see energy-meter-ref                 (461),
        -- see escalator-mode                   (462),
        -- see fault-signals                    (463),
        -- see floor-text                       (464),
        -- see group-id                         (465),
        -- enumeration value 466 is unassigned
        -- see group-mode                       (467),
        -- see higher-deck                      (468),
        -- see installation-id                  (469),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        -- see landing-calls                          (470),
        -- see landing-call-control                   (471),
        -- see landing-door-status                    (472),
        -- see lower-deck                             (473),
        -- see machine-room-id                        (474),
        -- see making-car-call                        (475),
        -- see next-stopping-floor                    (476),
        -- see operation-direction                    (477),
        -- see passenger-alarm                        (478),
        -- see power-mode                             (479),
        -- see registered-car-call                    (480),
        -- see active-cov-multiple-subscriptions      (481),
        -- see protocol-level                         (482),
        -- see reference-port                         (483),
        -- see deployed-profile-location              (484),
        -- see profile-location                       (485),
        -- see tags                                   (486),
        -- see subordinate-node-types                 (487),
        -- see subordinate-tags                       (488),
        -- see subordinate-relationships              (489),
        -- see default-subordinate-relationship       (490),
        -- see represents                             (491),
        -- see default-present-value                  (492),
        -- see present-stage                          (493),
        -- see stages                                 (494),
        -- see stage-names                            (495),
        -- see target-references                      (496),
        -- see audit-source-level                     (497),
        -- see audit-level                            (498),
        -- see audit-notification-recipient           (499),
        -- see audit-priority-filter                  (500),
        -- see auditable-operations                   (501),
        -- see delete-on-forward                      (502),
        -- see maximum-send-delay                     (503),
        -- see monitored-objects                      (504),
        -- see send-now                               (505),
        -- see floor-number                           (506),
        -- see device-uuid                            (507),
        ...
        }
        -- The special property identifiers all, optional, and required are reserved for use in the
        -- ReadPropertyMultiple service or services not defined in this standard.
        --
        -- Enumerated values 0-511 and enumerated values 4194304 and up are reserved for definition by ASHRAE.
        -- Enumerated values 512-4194303 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetPropertyReference ::= SEQUENCE {
        property-identifier         [0] BACnetPropertyIdentifier,
        property-array-index        [1] Unsigned OPTIONAL  -- used only with array datatype
                                                           -- if omitted with an array the entire array is referenced
        }

BACnetPropertyStates ::= CHOICE {
-- This production represents the possible datatypes for properties that
-- have discrete or enumerated values. The choice shall be consistent with the
-- datatype of the property referenced in the Event Enrollment Object.
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
boolean-value                [0]  BOOLEAN,
binary-value                 [1]  BACnetBinaryPV,
event-type                   [2]  BACnetEventType,
polarity                     [3]  BACnetPolarity,
program-change               [4]  BACnetProgramRequest,
program-state                [5]  BACnetProgramState,
reason-for-halt              [6]  BACnetProgramError,
reliability                  [7]  BACnetReliability,
state                        [8]  BACnetEventState,
system-status                [9]  BACnetDeviceStatus,
units                        [10] BACnetEngineeringUnits,
unsigned-value               [11] Unsigned,
life-safety-mode             [12] BACnetLifeSafetyMode,
life-safety-state            [13] BACnetLifeSafetyState,
restart-reason               [14] BACnetRestartReason,
door-alarm-state             [15] BACnetDoorAlarmState,
action                       [16] BACnetAction,
door-secured-status          [17] BACnetDoorSecuredStatus,
door-status                  [18] BACnetDoorStatus,
door-value                   [19] BACnetDoorValue,
file-access-method           [20] BACnetFileAccessMethod,
lock-status                  [21] BACnetLockStatus,
life-safety-operation        [22] BACnetLifeSafetyOperation,
maintenance                  [23] BACnetMaintenance,
node-type                    [24] BACnetNodeType,
notify-type                  [25] BACnetNotifyType,
--                           [26] removed,
shed-state                   [27] BACnetShedState,
silenced-state               [28] BACnetSilencedState,
--                           [29] reserved for future addenda
access-event                 [30] BACnetAccessEvent,
zone-occupancy-state         [31] BACnetAccessZoneOccupancyState,
access-credential-disable-reason [32] BACnetAccessCredentialDisableReason,
access-credential-disable    [33] BACnetAccessCredentialDisable,
authentication-status        [34] BACnetAuthenticationStatus,
backup-state                 [36] BACnetBackupState,
write-status                 [37] BACnetWriteStatus,
lighting-in-progress         [38] BACnetLightingInProgress,
lighting-operation           [39] BACnetLightingOperation,
lighting-transition          [40] BACnetLightingTransition,
integer-value                [41] INTEGER,
binary-lighting-value        [42] BACnetBinaryLightingPV,
timer-state                  [43] BACnetTimerState,
timer-transition             [44] BACnetTimerTransition,
bacnet-ip-mode               [45] BACnetIPMode,
network-port-command         [46] BACnetNetworkPortCommand,
network-type                 [47] BACnetNetworkType,
network-number-quality       [48] BACnetNetworkNumberQuality,
escalator-operation-direction [49] BACnetEscalatorOperationDirection,
escalator-fault              [50] BACnetEscalatorFault,
escalator-mode               [51] BACnetEscalatorMode,
lift-car-direction           [52] BACnetLiftCarDirection,
lift-car-door-command        [53] BACnetLiftCarDoorCommand,
lift-car-drive-status        [54] BACnetLiftCarDriveStatus,
lift-car-mode                [55] BACnetLiftCarMode,
lift-group-mode              [56] BACnetLiftGroupMode,
lift-fault                   [57] BACnetLiftFault,
protocol-level               [58] BACnetProtocolLevel,
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        audit-level                 [59]  BACnetAuditLevel,
        audit-operation             [60]  BACnetAuditOperation,
        extended-value              [63]  Unsigned32,
        -- example-one              [256]       BACnetExampleTypeOne,
        -- example-two              [257]       BACnetExampleTypeTwo,
        ...
        }
        -- Tag values greater than 254 are not encoded as ASN context tags. In these cases, the tag value is multiplied
        -- by 100000 and is added to the enumeration value and the sum is encoded using context tag 63,
        -- the extended-value choice.
        -- Tag values 0-63 are reserved for definition by ASHRAE. Tag values of 64-254 may be used by others to
        -- accommodate vendor specific properties that have discrete or enumerated values, subject to the constraints
        -- described in Clause 23.

BACnetPropertyValue ::= SEQUENCE {
        property-identifier         [0] BACnetPropertyIdentifier,
        property-array-index        [1] Unsigned OPTIONAL,  -- used only with array datatypes
                                                            -- if omitted with an array the entire array is referenced
        property-value              [2] ABSTRACT-SYNTAX.&Type, -- any datatype appropriate for the specified property
        priority                    [3] Unsigned (1..16) OPTIONAL  -- used only when property is commandable
        }

BACnetProtocolLevel ::= ENUMERATED {
        physical                    (0),
        protocol                    (1),
        bacnet-application          (2),
        non-bacnet-application      (3)
        }

BACnetRecipient ::= CHOICE {
        device                      [0] BACnetObjectIdentifier,
        address                     [1] BACnetAddress
        }

BACnetRecipientProcess ::= SEQUENCE {
        recipient                   [0] BACnetRecipient,
        process-identifier          [1] Unsigned32
        }

BACnetRelationship ::= ENUMERATED {
        unknown                     (0),
        default                     (1),
        -- Note that all of the following are in even/odd pairs indicating forward and reverse relationships.
        -- Proprietary extensions shall also follow the same even/odd pairing so that consumers can determine, for
        -- any given relationship, what the opposite relationship is.
        contains                    (2),
        contained-by                (3),
        uses                        (4),
        used-by                     (5),
        commands                    (6),
        commanded-by                (7),
        adjusts                     (8),
        adjusted-by                 (9),
        ingress                     (10),
        egress                      (11),
        supplies-air                (12),
        receives-air                (13),
        supplies-hot-air            (14),
        receives-hot-air            (15),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        supplies-cool-air               (16),
        receives-cool-air               (17),
        supplies-power                  (18),
        receives-power                  (19),
        supplies-gas                    (20),
        receives-gas                    (21),
        supplies-water                  (22),
        receives-water                  (23),
        supplies-hot-water              (24),
        receives-hot-water              (25),
        supplies-cool-water             (26),
        receives-cool-water             (27),
        supplies-steam                  (28),
        receives-steam                  (29),
        ...
        }
        -- Enumerated values 0-1023 are reserved for definition by ASHRAE. Enumerated values 1024-65535
        -- may be used by others subject to the procedures and constraints described in Clause 23.

BACnetReliability ::= ENUMERATED {
        no-fault-detected               (0),
        no-sensor                       (1),
        over-range                      (2),
        under-range                     (3),
        open-loop                       (4),
        shorted-loop                    (5),
        no-output                       (6),
        unreliable-other                (7),
        process-error                   (8),
        multi-state-fault               (9),
        configuration-error             (10),
        -- enumeration value 11 is reserved for a future addendum
        communication-failure           (12),
        member-fault                    (13),
        monitored-object-fault          (14),
        tripped                         (15),
        lamp-failure                    (16),
        activation-failure              (17),
        renew-dhcp-failure              (18),
        renew-fd-registration-failure   (19),
        restart-auto-negotiation-failure (20),
        restart-failure                 (21),
        proprietary-command-failure     (22),
        faults-listed                   (23),
        referenced-object-fault         (24),
        multi-state-out-of-range        (25),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetRestartReason ::= ENUMERATED {
        unknown                         (0),
        coldstart                       (1),
        warmstart                       (2),
        detected-power-lost             (3),
        detected-powered-off            (4),
        hardware-watchdog               (5),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        software-watchdog               (6),
        suspended                       (7),
        activate-changes                (8),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values 64-255
        -- may be used by others subject to the procedures and constraints described in Clause 23.

BACnetResultFlags ::= BIT STRING {
        first-item                      (0),
        last-item                       (1),
        more-items                      (2)
        }

BACnetRouterEntry ::= SEQUENCE {
        network-number                  [0]     Unsigned16,
        mac-address             [1]     OCTET STRING,
        status                  [2]     ENUMERATED {
                                                available       (0),
                                                busy            (1),
                                                disconnected    (2)
                                        },
        performance-index       [3]     Unsigned8 OPTIONAL
        }

BACnetScale ::= CHOICE {
        float-scale                     [0] REAL,
        integer-scale                   [1] INTEGER
}

BACnetSegmentation ::= ENUMERATED {
        segmented-both                  (0),
        segmented-transmit              (1),
        segmented-receive               (2),
        no-segmentation                 (3)
        }

BACnetServicesSupported ::= BIT STRING {
-- Alarm and Event Services
        acknowledge-alarm               (0),
        confirmed-cov-notification      (1),
        -- confirmed-cov-notification-multiple  (42),
        confirmed-event-notification    (2),
        get-alarm-summary               (3),
        get-enrollment-summary          (4),
        -- get-event-information        (39),
        -- life-safety-operation        (37),
        subscribe-cov                   (5),
        -- subscribe-cov-property       (38),
        -- subscribe-cov-property-multiple  (41),
        -- confirmed-audit-notification (44),

-- File Access Services
        atomic-read-file                (6),
        atomic-write-file               (7),
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- Object Access Services
        add-list-element                (8),
        remove-list-element             (9),
        create-object                   (10),
        delete-object                   (11),
        read-property                   (12),
        read-property-multiple          (14),
        -- read-range                   (35),
        -- write-group                  (40),
        write-property                  (15),
        write-property-multiple         (16),
        -- audit-log-query              (45),

-- Remote Device Management Services
        device-communication-control    (17),
        confirmed-private-transfer      (18),
        confirmed-text-message          (19),
        reinitialize-device             (20),
        -- who-Am-I                     (47),
        -- you-Are                      (48),

-- Virtual Terminal Services
        vt-open                         (21),
        vt-close                        (22),
        vt-data                         (23),

-- Removed Services
        -- formerly: read-property-conditional      (13), removed in version 1 revision 12
        -- formerly: authenticate                   (24), removed in version 1 revision 11
        -- formerly: request-key                    (25), removed in version 1 revision 11

-- Unconfirmed Services
        i-am                            (26),
        i-have                          (27),
        unconfirmed-cov-notification    (28),
        -- unconfirmed-cov-notification-multiple    (43),
        unconfirmed-event-notification  (29),
        unconfirmed-private-transfer    (30),
        unconfirmed-text-message        (31),
        time-synchronization            (32),
        -- utc-time-synchronization     (36),
        who-has                         (33),
        who-is                          (34),
        -- unconfirmed-audit-notification (46)

-- Services added after 1995
        read-range                      (35), -- Object Access Service
        utc-time-synchronization        (36), -- Remote Device Management Service
        life-safety-operation           (37), -- Alarm and Event Service
        subscribe-cov-property          (38), -- Alarm and Event Service
        get-event-information           (39), -- Alarm and Event Service
        write-group                     (40), -- Object Access Services

-- Services added after 2012
        subscribe-cov-property-multiple (41), -- Alarm and Event Service
        confirmed-cov-notification-multiple (42), -- Alarm and Event Service
        unconfirmed-cov-notification-multiple (43), -- Alarm and Event Service
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
-- Services added after 2016
        confirmed-audit-notification    (44), -- Alarm and Event Service
        audit-log-query                 (45), -- Object Access Service
        unconfirmed-audit-notification  (46), -- Alarm and Event Service
        who-Am-I                        (47), -- Remote Device Management Service
        you-Are                         (48) -- Remote Device Management Service
        }

BACnetSetpointReference ::= SEQUENCE {
        setpoint-reference [0] BACnetObjectPropertyReference OPTIONAL
        }

BACnetShedLevel ::= CHOICE {
        percent                 [0]  Unsigned,
        level                   [1]  Unsigned,
        amount                  [2]  REAL
        }

BACnetShedState ::= ENUMERATED {
        shed-inactive           (0),
        shed-request-pending    (1),
        shed-compliant          (2),
        shed-non-compliant      (3)
        }

BACnetSilencedState ::= ENUMERATED {
        unsilenced              (0),
        audible-silenced        (1),
        visible-silenced        (2),
        all-silenced            (3),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to procedures and constraints described in
        -- Clause 23.

BACnetSpecialEvent ::= SEQUENCE {
        period                  CHOICE {
                                    calendar-entry      [0] BACnetCalendarEntry,
                                    calendar-reference  [1] BACnetObjectIdentifier
                                    },
        list-of-time-values     [2] SEQUENCE OF BACnetTimeValue,
        event-priority          [3] Unsigned (1..16)
        }

BACnetStageLimitValue ::= SEQUENCE {
        limit                   REAL,
        values                  BITSTRING,      -- length is Nreferences bits, see Clause 12.62.6
        deadband                REAL
        }

BACnetStatusFlags ::= BIT STRING {
        in-alarm                (0),
        fault                   (1),
        overridden              (2),
        out-of-service          (3)
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetSuccessFilter ::= ENUMERATED {
        all                             (0),
        successes-only                  (1),
        failures-only                   (2)
        }

BACnetTimerState ::= ENUMERATED {
        idle                            (0),
        running                         (1),
        expired                         (2)
        }

BACnetTimerStateChangeValue ::= CHOICE {
        null                            NULL,
        boolean                         BOOLEAN,
        unsigned                        Unsigned,
        integer                         INTEGER,
        real                            REAL,
        double                          Double,
        octetstring                     OCTET STRING,
        characterstring                 CharacterString,
        bitstring                       BIT STRING,
        enumerated                      ENUMERATED,
        date                            Date,
        time                            Time,
        objectidentifier                BACnetObjectIdentifier,
        no-value                        [0] NULL,
        constructed-value               [1] ABSTRACT-SYNTAX.&Type,
        datetime                        [2] BACnetDateTime,
        lighting-command                [3] BACnetLightingCommand
        }
        -- This choice may be extended in the future with additional context
        -- tagged options for supporting other standardized complex data types

BACnetTimerTransition ::= ENUMERATED {
        none                            (0),
        idle-to-running                 (1),
        running-to-idle                 (2),
        running-to-running              (3),
        running-to-expired              (4),
        forced-to-expired               (5),
        expired-to-idle                 (6),
        expired-to-running              (7)
        }

BACnetTimeStamp ::= CHOICE {
        time                            [0]  Time, -- this form was deprecated in version 1 revision 21
        sequence-number                 [1]  Unsigned (0..65535),
        datetime                        [2]  BACnetDateTime
        }

BACnetTimeValue ::= SEQUENCE {
        time            Time,
        value           ABSTRACT-SYNTAX.&Type  -- any primitive datatype; complex types cannot be decoded
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
BACnetValueSource ::= CHOICE {
        none                    [0] NULL,
        object                  [1] BACnetDeviceObjectReference,
        address                 [2] BACnetAddress
        }

BACnetVMACEntry ::= SEQUENCE {
        virtual-mac-address     [0]   OCTET STRING, -- maximum size 6 octets
        native-mac-address      [1]   OCTET STRING
        }

BACnetVTClass ::= ENUMERATED {
        default-terminal        (0),
        ansi-x3-64              (1),
        dec-vt52                (2),
        dec-vt100               (3),
        dec-vt220               (4),
        hp-700-94               (5),
        ibm-3130                (6),
        ...
        }
        -- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
        -- 64-65535 may be used by others subject to the procedures and constraints described
        -- in Clause 23.

BACnetVTSession ::= SEQUENCE {
        local-vt-session-id     Unsigned8,
        remote-vt-session-id    Unsigned8,
        remote-vt-address       BACnetAddress
        }

BACnetWeekNDay ::= OCTET STRING (SIZE (3))
--      first octet     month (1..14)           where:  1 =January
--                                                      13 = odd months
--                                                      14 = even months
--                                                      X'FF' = any month
--      second octet    week-of-month (1..9)    where:  1 = days numbered 1-7
--                                                      2 = days numbered 8-14
--                                                      3 = days numbered 15-21
--                                                      4 = days numbered 22-28
--                                                      5 = days numbered 29-31
--                                                      6 = last 7 days of this month
--                                                      7 = any of the 7 days prior to the last 7 days of this month
--                                                      8 = any of the 7 days prior to the last 14 days of this month
--                                                      9 = any of the 7 days prior to the last 21 days of this month
--                                                      X'FF' = any week of this month
--      third octet     day-of-week (1..7)      where:  1 = Monday
--                                                      7 = Sunday
--                                                      X'FF' = any day of week

BACnetWriteStatus ::= ENUMERATED {
        idle                    (0),
        in-progress             (1),
        successful              (2),
        failed                  (3)
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
ReadAccessResult ::= SEQUENCE {
        object-identifier           [0] BACnetObjectIdentifier,
        list-of-results             [1] SEQUENCE OF SEQUENCE {
                        property-identifier     [2] BACnetPropertyIdentifier,
                        property-array-index    [3] Unsigned OPTIONAL, -- used only with array datatype
                                                    -- if omitted with an array the entire
                                                    -- array is referenced
                        read-result     CHOICE {
                                            property-value          [4] ABSTRACT-SYNTAX.&Type,
                                            property-access-error   [5] Error
                                            }
                        }
        }

ReadAccessSpecification ::= SEQUENCE {
        object-identifier               [0] BACnetObjectIdentifier,
        list-of-property-references     [1] SEQUENCE OF BACnetPropertyReference
        }

WriteAccessSpecification ::= SEQUENCE {
        object-identifier               [0] BACnetObjectIdentifier,
        list-of-properties              [1] SEQUENCE OF BACnetPropertyValue
        }
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

22 CONFORMANCE AND INTEROPERABILITY

BACnet defines a comprehensive set of object types and application services in the sense that communication requirements among all levels of control in a distributed, hierarchical building automation system are addressed. There is a need to account for the reality that not all devices in a building automation system need to support the full functionality of BACnet in order to perform their tasks.

To reach the overarching goal of this standard - communication between disparate building automation and control devices, possibly from different manufacturers - two distinct conditions must be met: 1) each implemented BACnet capability must precisely conform to the requirements of this standard; and 2) devices that seek to interoperate must implement precisely complementary BACnet capabilities appropriate to the desired form of interoperation. This clause defines how these conditions are to be met and what it means to conform to BACnet.

22.1 Conformance to BACnet

This clause specifies the requirements that shall be met in order to conform with BACnet.

22.1.1 Protocol Implementation Conformance Statement (PICS)

All devices conforming to the BACnet protocol shall have a Protocol Implementation Conformance Statement (PICS) that identifies all of the portions of BACnet that are implemented. This PICS shall contain all of the information described in Clause 22.1.1.1 and shall be in the format found in Annex A.

22.1.1.1 PICS Contents

A PICS is a written document, created by the manufacturer of a device, that identifies the particular options specified by BACnet that are implemented in the device. A BACnet PICS is considered a public document that is available for use by any interested party. At a minimum, a BACnet PICS shall convey the following information.

(a) Basic information identifying the vendor and describing the BACnet device.
(b) The BACnet Interoperability Building Blocks supported by the device (see Annex K).
(c) The standardized BACnet device profile to which the device conforms (see Annex L).
(d) All non-standard application services that are supported along with an indication for each service of whether the device can initiate the service request, respond to a service request, or both.
(e) A list of all standard and proprietary object types that are supported.
(f) For each object type supported,
    a. any optional properties that are supported,
    b. which properties can be written-to using BACnet services,
    c. If the objects can be dynamically created or deleted using BACnet services,
    d. any restrictions on the range of data values for properties.
(g) The data link layer option options, both real and virtual, supported. (See Annexes H and J).
(h) Whether segmented requests are supported.
(i) Whether segmented responses are supported.

22.1.2 Conformance Test

In order to conform to the BACnet protocol, all devices shall pass a conformance test that verifies the correct implementation of the standard object types and services indicated in the PICS. This conformance test shall consist of a collection of test cases drawn from a standard test suite in such a way as to test each object type and service for which support is claimed (positive test) and to test for an appropriate response to errors and standard services and objects that are not implemented to ensure the absence of detrimental behavior (negative test). The details of these tests are prescribed in the companion standard, "Testing Conformance to BACnet," ASHRAE 135.1.

22.1.3 Data Link and Physical Layers

To conform to the BACnet protocol, all devices shall support one of the five data link layer options, defined in Clauses 7 through 11, and one of the physical layers compatible with that data link layer, except as indicated in Clause 22.1.4.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

22.1.4 Conformance with Non-Standard Data Link Layer

Special circumstances may require that a device support a data link and physical layer technology that is not one of the BACnet options in order to interoperate with other networked devices in a particular situation. Such a device may be said to conform to BACnet with a non-standard data link layer, provided that the criteria in Clause 22.1.1 through 22.1.2 are met.

A device conforming to the BACnet protocol under the provisions of this clause may use non-standard protocol layers other than the data link and physical layers, provided that the non-standard protocol is used to convey a standard BACnet LSDU that contains application and network layer information defined by this standard and encoded according to the rules of Clause 20 and Clause 6. Segmentation of the BACnet LSDU is permitted. Annex H provides examples of this for the Department of Defense Internet protocols and the Novell Internetwork Datagram Protocol.

22.1.5 Minimum Device Requirements

A device that conforms to the BACnet protocol and contains an application layer shall:

(a) contain exactly one Device object,
(b) execute the ReadProperty service,
(c) execute the Who-Has and Who-Is services (and thus initiate the I-Have and I-Am services) unless the device is an MS/TP slave device,
(d) execute the WriteProperty service if the device executes the WritePropertyMultiple, AddListElement or RemoveListElement services,
(e) allow the WriteProperty service to modify any properties that are modifiable by the AddListElement or RemoveListElement services,
(f) execute the WriteProperty service if the device contains any objects with properties that are required to be writable,
(g) have a configurable device instance that can take on any value across the range 0 .. 4194302, and
(h) contain a Network Port object for each configured network port.

22.2 BACnet Interoperability

BACnet is intended to provide a single, uniform standard for building control systems, the ultimate goal of which is "interoperability." Interoperability means the ability of disparate control system devices to work together toward a common objective through the digital exchange of relevant information. Although interoperability is often thought of in terms of interconnecting equipment from multiple manufacturers, it is also possible to envision interoperating systems from a single vendor, possibly equipment of different vintages. Thus, while BACnet enables multi-vendor interoperability, it in no way requires it.

22.2.1 Interoperability Areas

"Interoperability areas" (IAs) are intended to describe the functionality that is important in practical automation and control systems to achieve specific operational objectives. The five IAs delineated in this standard are data sharing, alarm and event management, scheduling, trending, and device and network management. Each IA implies a set of capabilities. Each capability, in turn, requires that specific elements of BACnet be implemented in a particular device to enable interoperability in a known and predictable manner with a minimum of field engineering. The selection of which BACnet elements are required for a particular type of device is indicated in the device profiles presented in Annex L. This section describes the specific capabilities associated with each IA.

22.2.1.1 Data Sharing

"Data sharing" is the exchange of information between BACnet devices. It may be uni-directional or bi-directional. Interoperability in this area permits the collection of data for archival storage, graphics, and reports, the sharing of common sensor or calculated values between devices, carrying out interlocked control strategies, and the modification of setpoints or other operational parameters of BACnet objects.

22.2.1.2 Alarm and Event Management

"Alarm and event management" is the exchange of data between BACnet devices related to the occurrence of a predefined condition that meets specific criteria. Such conditions are called "events" and may be the basis for the initiation of a particular control action in response or the simple logging of the event's occurrence. The event may also

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

be deemed to represent a condition that constitutes an "alarm", requiring human acknowledgment and intervention. Interoperability in this area permits the annunciation and acknowledgment of alarms; the display of data indicating the basis for the alarm annunciation; the sharing of events for the purpose of logging or distributed control applications; modification of alarm limits and routing; and the production of summaries of the occurrence of such alarms and events.

BACnet defines two different mechanisms for generating alarms and events. One is called "intrinsic reporting" because it relies on the use of properties that are part of or "intrinsic" to the object that is being monitored for alarms or events. The other mechanism is called "algorithmic change reporting." Algorithmic change reporting is more general but it also requires the overhead of an additional object called the Event Enrollment object. The intrinsic reporting method is preferred under circumstances where it meets the objectives of the intended application. See Clause 13.

22.2.1.3 Scheduling

"Scheduling" is the exchange of data between BACnet devices related to the establishment and maintenance of dates and times at which specified output actions are to be taken. Interoperability in this area permits the use of date and time schedules or timers for starting and stopping equipment and changing control setpoints as well as other analog or binary parameters.

22.2.1.4 Trending

"Trending" is the accumulation of records consisting of a timestamp and a set of one or more logged data values. These records are collected at specified rates for a specified duration. The values are those of specific properties of specific objects. "Trending" is distinguished from the real-time plotting of data in that the data are usually destined for long-term storage and the sampling intervals are usually longer. Interoperability in this area permits the establishment of logging parameters and the subsequent retrieval and storage of logged data.

22.2.1.5 Device and Network Management

"Device and network management" is the exchange of data between BACnet devices concerning the operation and status of the devices comprising the BACnet internetwork. Interoperability in this area permits determining which devices are present on a given network and some of their operational capabilities, including which objects they maintain; the ability to start up and shut down communication from a particular device; the ability to synchronize the time in those devices that maintain clocks; the ability to reinitialize the operation of a device's computer; the ability to establish connections as needed; and the ability to change the connection configuration.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

23 EXTENDING BACnet TO ACCOMMODATE VENDOR PROPRIETARY INFORMATION

The objective of BACnet is to provide the mechanisms by which building automation equipment may exchange information. To aid in interoperability, BACnet defines a standardized set of data structures, called objects, which contain information that is common to most building systems. BACnet may also be used to exchange non-standardized information between devices that understand this information. There are four independent areas where BACnet may be extended to exchange non-standard information:

(a) A vendor may define proprietary extended values for enumerations used in BACnet.
(b) A vendor may invoke a proprietary service using the PrivateTransfer services.
(c) A vendor may add new proprietary properties to a standard object.
(d) A vendor may define new proprietary object types.
(e) A vendor may include standard items from future revisions of this standard, limited to:
    a. Error classes and error codes
    b. BACnetEngineeringUnits enumeration values
    c. BIBBs
    d. Device Profiles
    e. Data links that do not require any object, property, or service changes

In each of these cases, the BACnet messages implicitly reference a vendor identification code that serves to unambiguously specify which vendor's proprietary enumerations, services, properties, or objects were intended. Vendor identification codes are administrated by ASHRAE and are assigned one per vendor. The special Vendor_Identifier of zero is permanently assigned to ASHRAE. The Vendor_Identifier for a given device may be determined by reading the Vendor_Identifier property of the Device object. A list of vendor identification codes may be obtained from the ASHRAE Manager of Standards.

23.1 Extending Enumeration Values

There may be instances when it is necessary for a vendor to extend BACnet by including additional possible values to an enumeration. This is accomplished by using enumeration values that are greater than the range reserved for BACnet for a given enumeration type. Table 23-1 defines those enumerations that may be extended and the range of enumerated values reserved for BACnet use. All other enumerations, which do not appear in Table 23-1, shall not be extended.

Table 23-1. Extensible Enumerations

Enumeration Name	Reserved Range	Maximum Value
error-class	0...63	65535
error-code	0...255	65535
BACnetAbortReason	0...63	255
BACnetAccessAuthenticationFactorDisable	0...63	65535
BACnetAccessCredentialDisable	0...63	65535
BACnetAccessCredentialDisableReason	0...63	65535
BACnetAccessEvent	0...511	65535
BACnetAccessUserType	0...63	65535
BACnetAccessZoneOccupancyState	0...63	65535
BACnetAuditLevel	0...127	255
BACnetAuditOperation	0...31	63
BACnetAuthorizationExemption	0...63	255
BACnetAuthorizationMode	0...63	65535
BACnetBinaryLightingPV	0...63	255
BACnetDeviceStatus	0...63	65535
BACnetDoorAlarmState	0...255	65535
BACnetDoorStatus	0...1023	65535
BACnetEngineeringUnits	0...255, 47808...49999	65535
BACnetEscalatorFault	0...1023	65535
BACnetEscalatorMode	0...1023	65535
BACnetEscalatorOperationDirection	0...1023	65535
BACnetEventState	0...63	65535
BACnetEventType	0...63	65535

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Enumeration Name	Reserved Range	Maximum Value
BACnetLifeSafetyMode	0...255	65535
BACnetLifeSafetyOperation	0...63	65535
BACnetLifeSafetyState	0...255	65535
BACnetLiftCarDirection	0...1023	65535
BACnetLiftCarDriveStatus	0...1023	65535
BACnetLiftCarMode	0...1023	65535
BACnetLiftFault	0...1023	65535
BACnetLightingOperation	0...255	65535
BACnetLightingTransition	0...63	255
BACnetLoggingType	0...63	255
BACnetMaintenance	0...255	65535
BACnetNetworkPortCommand	0...127	255
BACnetNetworkType	0...63	255
BACnetObjectType	0...127	1023
BACnetProgramError	0...63	65535
BACnetPropertyIdentifier	0...511, 4194304...(2^{32} - 1)	(2^{32} - 1)
BACnetPropertyStates	0...63	254
BACnetRejectReason	0...63	255
BACnetRelationship	0...1023	65535
BACnetReliability	0...63	65535
BACnetRestartReason	0...63	255
BACnetSilencedState	0...63	65535
BACnetVTClass	0...63	65535

23.2 Using the PrivateTransfer Services to Invoke Non-Standardized Services

BACnet defines a set of application layer services that are specifically tailored to integrating building control systems. While this standard prescribes a set of application layer services that is intended to be comprehensive, vendors are free to create additional services. Standard services shall be used when possible.

Vendors may add proprietary services to BACnet using the PrivateTransfer services to invoke them. The service types and arguments are not restricted by BACnet, but they shall be conveyed using the Confirmed or Unconfirmed PrivateTransfer services. The protocol mechanisms used in the handling of these APDUs shall perform as specified in this standard.

The format of proprietary application layer services, invoked using PrivateTransfer, shall follow the encoding rules of this standard.

When using the PrivateTransfer service, it is important to note that segmentation is not permitted for Error APDUs. The implementor shall ensure that the parameters in the Error APDU do not expand to the point where segmentation is required.

23.3 Adding Proprietary Properties to a Standardized Object

BACnet defines a set of standard objects, each with a set of standard properties that can be accessed and manipulated with BACnet services. BACnet allows a vendor to add proprietary properties to extend the capabilities of a standard object. Proprietary properties receive the same support from BACnet services as standard properties and therefore can be accessed and manipulated in a manner identical to standard properties.

Objects may indicate conformance to an object profile by use of the Profile_Name property.

If a proprietary property is to be a commandable property, additional properties that fulfill the role of the standard Priority_Array and Relinquish_Default properties shall be provided for each commandable property. The priority arbitration mechanism described in Clause 19 shall apply.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Vendors may add proprietary properties to a standard object by modifying the object definition within a device. Proprietary properties are enumerated with Property_Identifier values of 512 and above. These property identifiers can be used in any BACnet service that uses a Property_Identifier as a parameter.

Proprietary property identifiers implicitly reference the Vendor_Identifier property of the Device object in the device where they reside. It is entirely possible, and expected, that different vendors will use the same enumeration values to represent completely different properties.

23.4 Adding Proprietary Object Types to BACnet

To accommodate building applications where the defined set of standardized objects is not adequate, BACnet allows a vendor to add proprietary object types. Standard object types shall be used when possible. To enhance extensibility, BACnet provides the same support for proprietary objects as for standard objects.

Objects may indicate conformance to an object profile by use of the Profile_Name property.

23.4.1 Proprietary Object_Type Enumerations

Vendors may add proprietary object types to BACnet by extending the BACnetObjectType enumeration. Proprietary object types are enumerated with Object_Type values of 128 and above. These Object_Type values may be used in any BACnet service that uses an Object_Type as a parameter.

23.4.2 Proprietary Property Datatypes

The properties of vendor proprietary objects may include both standard and proprietary datatypes. Proprietary datatypes may only be constructed from application datatypes defined in Clause 20.2.1.4.

23.4.3 Required Properties in Proprietary Object Types

Non-standard object types shall support the following properties:

Object_Identifier
Object_Name
Object_Type
Property_List

These properties shall be implemented to behave as they would in standard BACnet objects. This means that the Object_Identifier and Object_Name properties shall be unique within the BACnet device that maintains them. The Object_Name string shall be at least one character in length and shall consist of only printable characters. The Property_List property was added in Protocol Revision 14.

23.5 Restrictions on Extending BACnet

The following restrictions to extending BACnet apply:

(a) APDU types 8 through 15 are reserved for future ASHRAE use.

(b) Services may be added only via the Confirmed- and UnconfirmedPrivateTransfer services. That is, the enumerations BACnetConfirmedServiceChoice and BACnetUnconfirmedServiceChoice may not be extended.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

24 DELETED CLAUSE

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

25 REFERENCES

ANSI/EIA/CEA-709.1-B (2002), Control Network Protocol Specification.

ANSI/IEEE Standard 754 (1985), IEEE Standard for Binary Floating-Point Arithmetic.

ANSI/INCITS 92-1981 (R1998), (formerly ANSI X3.92-1981), Data Encryption Algorithm.

ANSI/INCITS X3.4-1986 (R1997), Information Processing - Coded Character Sets - 7-Bit American National Standard Code for Information Interchange (7-bit ASCII).

ANSI/TIA/EIA-232-F-1997 (R2002), Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange.

ANSI/TIA/EIA-485-A-1998 (R2003), Standard for Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems.

ANSI X3.41-1974 (R1990), American National Standard Code Extension Techniques for Use with the 7-bit Coded Character Set of American National Standard Code for Information Interchange.

ATA 878.1 (1999), ARCNET Local Area Network Standard.

DDN Protocol Handbook, Volumes 1-3, NIC 50004, 50005, and 50006.

Echelon, LonMark™ Layer 1-6 Interoperability Guidelines Version 3.3.

FIPS 46-2 (1993), Federal Information Processing Standards - Data Encryption Standard.

FIPS 180-2 (2002), Federal Information Processing Standards - Secure Hash Standard

FIPS 197 (2002), Federal Information Processing Standards - Advanced Encryption Standard

IEEE Transactions on Dependable and Secure Computing, Vol. 6, No. 1, January – March 2009, The Effectiveness of Checksums for Embedded Control Networks , T. Maxino and P. Koopman

IEEE/ACM TRANSACTIONS ON NETWORKING, VOL.7, NO.2, APRIL, 1999, Consistent Overhead Byte Stuffing, S. Cheshire and M. Baker.

IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2002), 32-Bit Cyclic Redundancy Codes for Internet Applications, P. Koopman.

IETF RFC 768, User Datagram Protocol, Internet Engineering Task Force

IETF RFC 791, Internet Protocol, Internet Engineering Task Force

IETF RFC 1055, Transmission of IP Datagrams over Serial Lines: SLIP, Internet Engineering Task Force

IETF RFC 1112, Host Extensions for IP Multicasting, Internet Engineering Task Force

IETF RFC 1123, Requirements for Internet Hosts, Internet Engineering Task Force

IETF RFC 1321, The MD5 Message-Digest Algorithm, Internet Engineering Task Force

IETF RFC 1661, The Point-to-Point Protocol (PPP), Internet Engineering Task Force

IETF RFC 1738, Uniform Resource Locators (URL), Internet Engineering Task Force

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

IETF RFC 2246, The TLS Protocol Version 1.0, Internet Engineering Task Force

IETF RFC 2373, IP Version 6 Addressing Architecture, Internet Engineering Task Force

IETF RFC 2460, Internet Protocol, Version 6 (IPv6) Specification, Internet Engineering Task Force

IETF RFC 2616, Hypertext Transfer Protocol - HTTP/1.1, Internet Engineering Task Force

IETF RFC 2663, IP Network Address Translator (NAT), Internet Engineering Task Force

IETF RFC 3041, Privacy Extensions for Stateless Address Autoconfiguration in IPv6, Internet Engineering Task Force

IETF RFC 3066, Tags for the Identification of Languages, Internet Engineering Task Force

IETF RFC 3629, UTF-8, a transformation format of ISO 10646, Internet Engineering Task Force

IETF RFC 3986, Uniform Resource Identifier (URI), Internet Engineering Task Force

IETF RFC 3987, Internationalized Resource Identifiers (IRIs), Internet Engineering Task Force

IETF RFC 4007, IPv6 Scoped Address Architecture, Internet Engineering Task Force

IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace, Internet Engineering Task Force

IETF RFC 4180, Common Format and MIME Type for Comma-Separated Values (CSV) Files, Internet Engineering Task Force

IETF RFC 4193, Unique Local IPv6 Unicast Addresses, Internet Engineering Task Force

IETF RFC 4291, IP Version 6 Addressing Architecture, Internet Engineering Task Force

IETF RFC 4294, IPv6 Node Requirements, Internet Engineering Task Force

IETF RFC 4627, The application/json Media Type for JavaScript Object Notation (JSON), Internet Engineering Task Force

IETF RFC 4862, IPv6 Stateless Address Autoconfiguration

IETF RFC 5280, X.509 PKI Certificate and Certificate Revocation List (CRL) Profile, Internet Engineering Task Force

IETF RFC 5289 (2008), TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM), Internet Engineering Task Force

IETF RFC 5785, Defining Well-Known Uniform Resource Identifiers (URIs), Internet Engineering Task Force

IETF RFC 5952, A Recommendation for IPv6 Address Text Representation, Internet Engineering Task Force

IETF RFC 5958, Asymmetric Key Packages, Internet Engineering Task Force

IETF RFC 5988, Web Linking, Internet Engineering Task Force

IETF RFC 6455 (2011), The WebSocket Protocol, Internet Engineering Task Force

IETF RFC 6724, Default Address Selection for Internet Protocol Version 6 (IPv6), Internet Engineering Task Force

IETF RFC 6749, The OAuth 2.0 Authorization Framework, Internet Engineering Task Force

IETF RFC 6750, The OAuth 2.0 Authorization Framework: Bearer Token Usage, Internet Engineering Task Force

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

IETF RFC 6762 (2013), Multicast DNS, Internet Engineering Task Force

IETF RFC 6838, Media Type Specifications and Registration Procedures, Internet Engineering Task Force

IETF RFC 7009, OAuth 2.0 Token Revocation, Internet Engineering Task Force

IETF RFC 7159, The JavaScript Object Notation (JSON) Data Interchange Format, Internet Engineering Task Force

IETF RFC 7251 (2014), AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS, Internet Engineering Task Force

IETF RFC 7468 (2015), Textual Encodings of PKIX, PKCS, and CMS Structures, Internet Engineering Task Force

IETF RFC 7519, JSON Web Token (JWT), Internet Engineering Task Force

IETF RFC 8163 (2017), Transmission of IPv6 over Master-Slave/Token-Passing (MS/TP) Networks, Internet Engineering Task Force

IETF RFC 8446 (2018), The Transport Layer Security (TLS) Protocol Version 1.3, Internet Engineering Task Force

ISO 7498 (1984), Information processing systems - Open Systems Interconnection - Basic Reference Model.

ISO TR 8509 (1987), Information processing systems - Open Systems Interconnection - service conventions.

ISO 8649 (1988), Information processing systems - Open Systems Interconnection - Service definition for the Association Control Service Element.

ISO 8802-2 (1998), Information processing systems - Local area networks - Part 2: Logical link control.

ISO/IEC 8802-3 (2000), Information processing systems - Local area networks - Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.

ISO 8822 (1994), Information processing systems - Open Systems Interconnection - Connection-oriented presentation service definition.

ISO/IEC 8824 (1990), Information technology - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).

ISO/IEC 8825 (1990), Information technology - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).

ISO 9545 (1994), Information processing systems - Open Systems Interconnection - Application Layer Structure (ALS).

ISO/IEC 10646-1 (2000), IT - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane.

JIS C 6226 (1983), Code of the Japanese Graphic Character Set for Information Interchange. Japan Institute for Standardization.

Konnex Association, Konnex Handbook Volume 3: System Specifications.

Konnex Association, Konnex Handbook Volume 3: System Specifications, Part 7: Interworking, Chapter 2: Datapoint Types.

Konnex Association, Konnex Handbook Volume 3: System Specifications, Part 7: Interworking, Chapter 3: Standard Identifier Tables, Annex 1 - Property Identifiers.

Konnex Association, Konnex Handbook Volume 7: Applications Descriptions.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

NETSCAPE SSL3 DRAFT302 (1996), The SSL Protocol Version 3.0, Netscape Communications

UNICODE Technical Report# 17-5: Character Encoding Model. The Unicode Consortium.

W3C (2000), Simple Object Access Protocol (SOAP) 1.1, World Wide Web Consortium

W3C (2001), XML Schema Part 0: Primer, World Wide Web Consortium

W3C (2001), XML Schema Part 1: Structures, World Wide Web Consortium

W3C (2001), XML Schema Part 2: Datatypes, World Wide Web Consortium

W3C (2003), Extensible Markup Language (XML) 1.0 (Second Edition), World Wide Web Consortium

WS-I (2004), WS-I Basic Profile 1.0, Web Services Interoperability Organization

Sources for Reference Material

ANSI: American National Standards Institute, 25 West 43rd St., 4th Floor, New York , NY 10036.

Cheshire: http://www.stuartcheshire.org/papers/COBSforToN.pdf

DDN: Available from the Defense Data Network Information Center, SRI International, 333 Ravenswood Ave., Room EJ291, Menlo Park, CA 94025.

Echelon: Echelon Corporation, 550 Meridian Ave., San Jose, CA 95126.

EIA: Electronics Industries Alliance, 2500 Wilson Blvd. Arington, VA 22201.

EIBA: EIB Association (EIBA) s.c.r.l., Neerveldstraat / Rue de Neerveld 105, B-1200 Brussels, Belgium

FIPS: National Institute of Standards and Technology, Gaithersburg, MD 20899.

IEEE: The Institute of Electrical and Electronics Engineers, Inc., 3 Park Ave., 17th Floor, New York, NY 10016.

Internet Engineering Task Force, www.ietf.org.

INCITS: The International Committee for Information Technology Standards (INCITS) is sponsored by the Information Technology Industry Council (ITI), 1250 Eye St. NW, Suite 200, Washington, DC 20005.

ISO: Available from ANSI.

JIS: Available from ANSI.

Konnex Association: Neerveldstraat / Rue de Neerveld 105, B-1200 Brussels, Belgium

Koopman: http://www.ece.cmu.edu/~koopman/projects.html#crc

LonMark International, 550 Meridian Avenue, San Jose, CA 95126.

Netscape Communications, www.netscape.com

The Unicode Consortium. P.O. Box 391476, Mountain View, CA 94039-1476, USA.

W3C: World Wide Web Consortium, www.w3.org

WS-I: Web Services Interoperability Organization, www.ws-i.org

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX A - PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT (NORMATIVE)

(This annex is part of this standard and is required for its use.)

BACnet Protocol Implementation Conformance Statement

Date: _____
Vendor Name:
Product Name:
Product Model Number:
Application Software Version: _____ **Firmware Revision:** _____ **BACnet Protocol Revision:** _____

Product Description:

_____
_____
_____
_____
_____
_____

BACnet Standardized Device Profiles Supported (Annex L):

☐ BACnet Cross-Domain Advanced Operator Workstation (B-XAWS)
☐ BACnet Advanced Operator Workstation (B-AWS)
☐ BACnet Operator Workstation (B-OWS)
☐ BACnet Operator Display (B-OD)
☐ BACnet Advanced Lighting Workstations (B-ALWS)
☐ BACnet Lighting Operator Display (B-LOD)
☐ BACnet Advanced Life Safety Workstation (B-ALSWS)
☐ BACnet Life Safety Workstation (B-LSWS)
☐ BACnet Life Safety Annunciator Panel (B-LSAP)
☐ BACnet Advanced Access Control Workstation (B-AACWS)
☐ BACnet Access Control Workstation (B-ACWS)
☐ BACnet Access Control Security Display (B-ACSD)
☐ BACnet Advanced Elevator Workstation (B-AEWS)
☐ BACnet Elevator Workstation (B-EWS)
☐ BACnet Elevator Display (B-ED)
☐ BACnet Advanced Lighting Control Station (B-ALCS)
☐ BACnet Lighting Control Station (B-LCS)
☐ BACnet Building Controller (B-BC)
☐ BACnet Advanced Application Controller (B-AAC)
☐ BACnet Application Specific Controller (B-ASC)
☐ BACnet Smart Actuator (B-SA)
☐ BACnet Smart Sensor (B-SS)
☐ BACnet Lighting Supervisor (B-LS)
☐ BACnet Lighting Device (B-LD)
☐ BACnet Advanced Life Safety Controller (B-ALSC)
☐ BACnet Life Safety Controller (B-LSC)
☐ BACnet Advanced Access Control Controller (B-AACC)
☐ BACnet Access Control Controller (B-ACC)
☐ BACnet Advanced Elevator Controller (B-AEC)
☐ BACnet Elevator Controller (B-EC)
☐ BACnet Elevator Monitor (B-EM)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

☐ BACnet Router (B-RTR)
☐ BACnet Gateway (B-GW)
☐ BACnet Broadcast Management Device (B-BBMD)
☐ BACnet Access Control Door Controller (B-ACDC)
☐ BACnet Access Control Credential Reader (B-ACCR)
☐ BACnet Secure Connect Hub (B-SCHUB)

☐ BACnet General (B-GENERAL)

BACnet Interoperability Building Blocks Supported (Annex K): _____
_____
_____
_____

Segmentation Capability:

☐Able to transmit segmented messages Window Size _____
☐Able to receive segmented messages Window Size _____

Standard Object Types Supported:

An object type is supported if it may be present in the device. For each standard Object Type supported provide the following data:

1) Whether objects of this type are dynamically creatable using the CreateObject service
2) Whether objects of this type are dynamically deletable using the DeleteObject service
3) List of the optional properties supported
4) List of all properties that are writable where not otherwise required by this standard
5) List of all properties that are conditionally writable where not otherwise required by this standard
6) List of proprietary properties and for each its property identifier, datatype, and meaning
7) List of any property range restrictions

BACnet Data Link Layer Options:

☐ ARCNET (ATA 878.1), 2.5 Mb. (Clause 8)
☐ ARCNET (ATA 878.1), EIA-485 (Clause 8), baud rate(s) _____
☐ BACnet IP, (Annex J)
☐ BACnet IP, (Annex J), BACnet Broadcast Management Device (BBMD)
☐ BACnet IP, (Annex J), Network Address Translation (NAT Traversal)
☐ BACnet IPv6, (Annex U)
☐ BACnet IPv6, (Annex U), BACnet Broadcast Management Device (BBMD)
☐ BACnet/ZigBee (Annex O) _____
☐ Ethernet, ISO 8802-3 (Clause 7)
☐ LonTalk, ISO/IEC 14908.1 (Clause 11), medium: _____
☐ MS/TP master (Clause 9)
  ☐ Master ☐ Slave
  ☐ Non-isolated transceiver ☐ Isolated transceiver
  ☐ Local 47K ohms bias resistors ☐ None ☐ Other: _____
  Transceiver unit loading: ☐ 1 ☐ ½ ☐ ¼ ☐ ⅛
  Data rates: ☐ 9600 ☐ 19200 ☐ 38400 ☐ 57600 ☐ 76800 ☐ 115200
☐ Point-To-Point, EIA 232 (Clause 10), baud rate(s):_____
☐ Point-To-Point, modem, (Clause 10), baud rate(s): _____

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

☐ BACnet Secure Connect (Annex AB)

☐ BACnet Secure Connect Node

If direct connections are supported:

Maximum number of simultaneous direct connections initiated: _____

Maximum number of simultaneous direct connections accepted: _____

☐ BACnet Secure Connect Hub Function

Maximum number of simultaneous hub connections accepted: _____

☐ HTTPS Proxy Support

List the types of HTTPS proxies supported: _____

☐ Additional cipher suites supported beyond those required for TLS V1.3

The additional cipher suites supported using the cipher suite names as of the TLS Cipher Suite Registry

at IANA (See RFC 8446):

_____

_____

☐ Additional Transport Layer Security versions other than V1.3 supported

The TLS versions other than V1.3 that are supported, including the supported cipher suites for the

version beyond those required, using the cipher suite names as defined by the TLS version supported:

_____

_____

☐ Generates private keys internally, and provides matching certificate signing requests.

☐ DNS host name resolution supported (RFC 1123)

☐ mDNS host name resolution supported (RFC 6762)

☐ Other:

Device Address Binding:

Is static device binding supported? (This is currently necessary for two-way communication with MS/TP slaves and certain other devices.) ☐ Yes ☐ No

Networking Options:

☐ Router, Clause 6 - List all routing configurations, e.g., ARCNET-Ethernet, Ethernet-MS/TP, etc.

☐ Annex H, BACnet Tunneling Router over IP

Character Sets Supported:

Indicating support for multiple character sets does not imply that they can all be supported simultaneously.

☐ ISO 10646 (UTF-8)	☐ IBM™/Microsoft™ DBCS	☐ ISO 8859-1
☐ ISO 10646 (UCS-2)	☐ ISO 10646 (UCS-4)	☐ JIS X 0208

Gateway Options:

If this product is a communication gateway, describe the types of non-BACnet equipment/networks(s) that the gateway supports:

_____

_____

_____

If this product is a communication gateway which presents a network of virtual BACnet devices, a separate PICS shall be provided that describes the functionality of the virtual BACnet devices. That PICS shall describe a superset of the functionality of all types of virtual BACnet devices that can be presented by the gateway.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX B - GUIDE TO SPECIFYING BACnet DEVICES (INFORMATIVE)

(This annex is not part of this standard but is included for informative purposes only.)

The BIBBs (Annex K) and standardized BACnet device profiles (Annex L) are intended to be a useful tool for people who design, specify, or operate building automation systems that contain BACnet devices. This classification approach is a compromise between two conflicting goals. The first goal is to promote interoperability by limiting the various combinations of BACnet object types and services that can be supported and still conform to this standard. The other goal is to avoid unnecessarily restricting manufacturers of BACnet devices in the sense that they would be required to provide BACnet functionality that would never be used by a device except to meet a conformance requirement. Maximum interoperability would be achieved by requiring all BACnet devices to support exactly the same combination of standard object types and application services. On the other hand, complete flexibility for manufacturers would inevitably lead to such widespread variation in the particular object types and application services that are supported that many devices would only partially interoperate. Interoperability would be limited to the intersection of the application services and object types supported by the devices.

The idea behind the BIBB and device profile model is to combine the portions of the BACnet protocol that are needed to perform particular functions, and to identify those functions that a system designer would expect in a certain type of device. When designing or specifying BACnet devices for an automation system, it is appropriate to specify the device profile that best meets the needs of the application and any additional BIBBs that are also required. Devices can be expected to interoperate with respect to a given BIBB so long as one device implements the A-side functionality and the other device implements the B-side functionality.

A particular manufacturer may decide to build a product that supports more BIBBs than required by its device profile. This can be determined from the PICS.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX C - Removed

(This annex has been removed from the standard.)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX D - Removed

(This annex has been removed from the standard.)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX E - EXAMPLES OF BACnet APPLICATION SERVICES (INFORMATIVE)

(This annex is not part of this standard but is included for informative purposes only.)

This annex provides examples of the use of application services defined in Clauses 13 to 17. In these examples, the names of properties are spelled out in Mixed Case, the values of properties with a datatype of CharacterString are enclosed in double quotes ("), and the values of enumerated types are spelled out in UPPER CASE. No encoding of parameters is contained in the examples; only their unencoded, symbolic values are shown. The encoding for these examples may be found in Annex F.

E.1 Alarm and Event Services

E.1.1 Examples of the AcknowledgeAlarm Service

See section E.1.4 for the use of the AcknowledgeAlarm service in conjunction with the ConfirmedEventNotification service and section E.1.5 for use of the AcknowledgeAlarm service in conjunction with the UnconfirmedEventNotification service.

E.1.2 Example of the ConfirmedCOVNotification Service

The following example illustrates a notification that the present value of a monitored Analog Input object has changed.

Service =	ConfirmedCOVNotification
'Subscriber Process Identifier' =	18
'Initiating Device Identifier' =	(Device, Instance 4)
'Monitored Object Identifier' =	(Analog Input, Instance 10)
'Time Remaining' =	0
'List of Values' =	((Present_Value, 65.0), (Status_Flags, (FALSE, FALSE, FALSE, FALSE)))

E.1.3 Example of the UnconfirmedCOVNotification Service

The following example illustrates a notification that the present value of a monitored Analog Input object has changed.

Service =	UnconfirmedCOVNotification
'Subscriber Process Identifier' =	18
'Initiating Device Identifier' =	(Device, Instance 4)
'Monitored Object Identifier' =	(Analog Input, Instance 10)
'Time Remaining' =	0
'List of Values' =	((Present_Value, 65.0), (Status_Flags, (FALSE, FALSE, FALSE, FALSE)))

E.1.4 Example of the ConfirmedEventNotification Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT

Consider an Analog Input called "Zone1_Temp" that supports intrinsic event reporting. Assume that the temperature in zone one has just risen to the point that it exceeds the high-limit value, causing the alarm to become active and the TO_OFFNORMAL bit in the Acked_Transitions property to be cleared. Further assume that there is no associated message text. The device will issue a ConfirmedEventNotification service request primitive with the following parameters:

Service =	ConfirmedEventNotification
'Process Identifier' =	1
'Initiating Device Identifier' =	(Device, Instance 4)
'Event Object Identifier' =	(Analog Input, Instance 2)
'Time Stamp' =	16
'Notification Class' =	4
'Priority' =	100
'Event Type' =	OUT_OF_RANGE
'Notify Type' =	ALARM

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'AckRequired' = TRUE
'From State' = NORMAL
'To State' = HIGH_LIMIT
'Event Values' = ((Exceeding_Value, 80.1), (Status_Flags, (TRUE, FALSE, FALSE, FALSE)), (Deadband, 1.0), (Exceeded_Limit, 80.0))

This PDU will be sent to each device specified by the Notification Class object associated with this Analog Input. Assuming that the PDU is correctly received, a 'Result(+)' confirm primitive will be received from each recipient as a confirmation that the message was received. Because the 'AckRequired' parameter has a value of TRUE, it is expected that a human operator will be notified and will acknowledge the alarm. After this happens, an AcknowledgeAlarm indication primitive will be received. The parameters in this primitive will be:

Service = AcknowledgeAlarm
'Acknowledging Process Identifier' = 1
'Event Object Identifier' = (Analog Input, Instance 2)
'Event State Acknowledged' = HIGH_LIMIT
'Time Stamp' = 16
'Acknowledgment Source' = "MDL"
'Time Of Acknowledgment' = (21-Jun-1992, 13:03:41.9)

The local BACnet device will locate the appropriate Analog Input object instance, set the TO_OFFNORMAL bit in the Acked_Transitions property, and issue a 'Result(+)' response primitive. Note that the 'Time Stamp' parameter is a sequence number in this example, and it has the value that was sent with the ConfirmedEventNotification.

E.1.5 Example of the UnconfirmedEventNotification Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT

Consider an Analog Input called "Zone1_Temp" that supports intrinsic event reporting. Assume that the temperature in zone one has just risen to the point that it exceeds the high-limit value, causing the alarm to become active and the TO_OFFNORMAL bit in the Acked_Transitions property to be cleared. Further assume that there is no associated message text. The device will issue an UnconfirmedEventNotification service request primitive with the following parameters:

Service = UnconfirmedEventNotification
'Process Identifier' = 1
'Initiating Device Identifier' = (Device, Instance 9)
'Event Object Identifier' = (Analog Input, Instance 2)
'Time Stamp' = 16
'Notification Class' = 4
'Priority' = 100
'Event Type' = OUT_OF_RANGE
'Notify Type' = ALARM
'AckRequired' = TRUE
'From State' = NORMAL
'To State' = HIGH_LIMIT
'Event Values' = ((Present_Value, 80.1), (Status_Flags, (TRUE, FALSE, FALSE, FALSE)), (Deadband, 1.0), (High_Limit, 80.0))

This PDU will be sent to a particular recipient or broadcast locally, remotely, or globally depending on the value of the Recipient_List property of the Notification Class object associated with this Analog Input. Because the 'AckRequired' parameter has a value of TRUE, it is expected that a human operator will be notified and will acknowledge the alarm. After this happens, an AcknowledgeAlarm indication primitive will be received. The parameters in this primitive will be:

Service = AcknowledgeAlarm
'Acknowledging Process Identifier' = 1
'Event Object Identifier' = (Analog Input, Instance 2)
'Event State Acknowledged' = HIGH_LIMIT
'Time Stamp' = 16

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'Acknowledgment Source' = "MDL"
'Time Of Acknowledgment' = (21-Jun-1992, 13:03:41.9)

The local BACnet device will locate the appropriate Analog Input object instance, set the TO_OFFNORMAL bit in the Acked_Transitions property, and issue a 'Result(+)' response primitive. Note that the 'Time Stamp' parameter is a sequence number in this example, and it has the value that was sent with the UnconfirmedEventNotification.

E.1.6 Example of the GetAlarmSummary Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT
	(Analog Input, Instance 3)	Zone2_Temp	ANALOG_INPUT

A BACnet device attempting to obtain a list of active alarms from another BACnet device would issue the following service request:

Service = GetAlarmSummary

A typical response to this request would be a 'Result(+)' containing no parameters, indicating that there are no active alarms, or a 'Result(+)', conveying a list of active alarms as shown below.

'List of Alarm Summaries' = (((Analog Input, Instance 2), HIGH_LIMIT, B'011'),
((Analog Input, Instance 3), LOW_LIMIT, B'111'))

Notice that the LOW_LIMIT from "Zone2_Temp" was previously acknowledged, but the HIGH_LIMIT from "Zone1_Temp" was not.

E.1.7 Examples of the GetEnrollmentSummary Service

Example 1: Obtain a summary of all alarms that are not acknowledged.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT
	(Event Enrollment, Instance 6)	CW_Flow_Alarm	EVENT_ENROLLMENT

Service = GetEnrollmentSummary
'Acknowledgment Filter' = NOT-ACKED

A typical response to this request would be a 'Result(+)' containing no parameters, indicating that there are no unacknowledged alarms, or a 'Result(+)', conveying a list of active alarms as shown below.

'List of Enrollment Summaries' = (((Analog Input, Instance 2), OUT_OF_RANGE, HIGH_LIMIT, 100, 4),
((Event Enrollment, Instance 6), CHANGE_OF_STATE, NORMAL, 50, 2))

Example 2: Obtain a summary of all event enrollments with a priority in the range 6-10 for which a particular device subscribes.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT
	(Analog Input, Instance 3)	Zone2_Temp	ANALOG_INPUT
	(Analog Input, Instance 4)	Zone3_Temp	ANALOG_INPUT
	(Event Enrollment, Instance 7)	CW_Temp_Alarm	EVENT_ENROLLMENT
	(Device, Instance 17)	Console 1	DEVICE

Service = GetEnrollmentSummary
'Acknowledgment Filter' = ALL
'Enrollment Filter' = ((Device, Instance 17), 9)
'Priority Filter' = (6,10)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A typical response to this request would be a 'Result(+)' containing no parameters, indicating that there are no event enrollments that meet the specified criteria, or a 'Result(+)', conveying a list of Event Enrollment objects that do meet the criteria as shown below.

'List of Enrollment Summaries' = (((Analog Input, Instance 2), OUT_OF_RANGE, NORMAL, 8, 4),
((Analog Input, Instance 3), OUT_OF_RANGE, NORMAL, 8, 4),
((Analog Input, Instance 4), OUT_OF_RANGE, NORMAL, 8, 4),
((Event Enrollment, Instance 7), FLOATING_LIMIT, NORMAL, 3, 8))

E.1.8 Example of the GetEventInformation Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 2)	Zone1_Temp	ANALOG_INPUT
	(Analog Input, Instance 3)	Zone2_Temp	ANALOG_INPUT

A BACnet device attempting to obtain a list of active events from another BACnet device would issue the following service request:

Service = GetEventInformation

A typical response to this request would be a 'Result(+)' containing no parameters, indicating that there are no active events, or a 'Result(+)', conveying a list of active events as shown below.

'List of Event Summaries' = (((Analog Input, Instance 2), HIGH_LIMIT, B'011', ((7-Jun-99,15:35:00.20),
(*-*-*,*:*:*.*), (*-*-*,*:*:*.*)), ALARM, B'111', (15, 15, 20)),
((Analog Input, Instance 3), NORMAL, B'110', ((7-Jun-99, 15:40:00.00),
(*-*-*,*:*:*:*), (15:45:30.30)), ALARM, B'111', (15, 15, 20)),
FALSE)

Notice that the to NORMAL transition from Zone2_Temp has not been acknowledged.

E.1.9 Example of the LifeSafetyOperation Service

This example illustrates the use of the LifeSafetyOperation service to reset a trouble condition.

Service = LifeSafetyOperation
'Requesting Process Identifier' = 18
'Requesting Source' = "MDL"
'Request' = RESET
'Object Identifier' = (Life Safety Point, 1)

E.1.10 Example of the SubscribeCOV Service

This example illustrates the use of the SubscribeCOV service to subscribe indefinitely to COV notifications from an Analog Input object.

Service = SubscribeCOV
'Subscriber Process Identifier' = 18
'Monitored Object Identifier' = (Analog Input, Instance 10)
'Issue Confirmed Notifications' = TRUE
'Lifetime' = 0

E.1.11 Example of the SubscribeCOVProperty Service

This example illustrates the use of the SubscribeCOVProperty service to subscribe to COV notifications on the present-value of an Analog Input object.

Service = SubscribeCOVProperty
'Subscriber Process Identifier' = 18
'Monitored Object Identifier' = (Analog Input, Instance 10)
'Issue Confirmed Notifications' = TRUE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

'Lifetime' = 60
'Monitored Property' = (Present_Value)
'COV Increment' = 1.0

E.1.12 Example of the SubscribeCOVPropertyMultiple Service

This example illustrates the use of the SubscribeCOVPropertyMultiple service to subscribe to COV notifications on properties of two objects.

Service = SubscribeCOVPropertyMultiple
'Subscriber Process Identifier' = 18
'Issue Confirmed Notifications' = TRUE
'Lifetime' = 60
'Max Notification Delay' = 5
'List of COV Subscription Specifications' =
(
((Analog Input, Instance 10),
((PRESENT_VALUE, 1.0, TRUE),(RELIABILITY, , FALSE))),
((Analog Output, Instance 8), ((PRESENT_VALUE, 0.1, TRUE)))
)

E.1.13 Example of the ConfirmedCOVNotificationMultiple Service

The following example illustrates a notification that two of the properties for which COV notifications were subscribed using the SubscribeCOVPropertyMultiple service have changed. The first property was subscribed with 'Timestamped' set to TRUE, the second with 'Timestamped' set to FALSE.

Service = ConfirmedCOVNotificationMultiple
'Subscriber Process Identifier' = 18
'Initiating Device Identifier' = (Device, Instance 4)
'Time Remaining' = 35
'Timestamp' = ((June 3, 2013 (Day of Week = Monday)), (03:23:53.47))
'List of COV Notifications' = (((Analog Input, Instance 10), ((Present_Value, , 65.0, (03:23:52.00)))),
((Analog Output, Instance 8), ((Present_Value, ,80.1,))))

E.1.14 Example of the UnconfirmedCOVNotificationMultiple Service

The following example illustrates a notification that one of the properties for which COV notifications were subscribed using the SubscribeCOVPropertyMultiple service has changed. Since no timestamped change of value is conveyed, the 'Timestamp' parameter is not present.

Service = UnconfirmedCOVNotificationMultiple
'Subscriber Process Identifier' = 18
'Initiating Device Identifier' = (Device, Instance 4)
'Time Remaining' = 27
'List of COV Notifications' = (((Analog Input, Instance 10), ((Present_Value, , 65.0,))))

E.2 File Access Services

The following examples illustrate typical uses of files. The actual form and content of files is both vendor-specific and application-specific.

E.2.1 Examples of the AtomicReadFile Service

Example 1: Read data from a file.

Assumed objects:	Object Identifier	Object Name	Object Type
	(File, Instance 1)	ChillerData	FILE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Service = AtomicReadFile
'File Identifier' = (File, Instance 1)
'Stream Access':
'File Start Position' = 0
'Requested Octet Count' = 27

The 'Result(+)' parameters would be:

'End Of File' = FALSE
'Stream Access':
'File Start Position' = 0
'File Data ' = "Chiller01 On-Time=4.3 Hours"

Example 2: Read record from a file.

Assumed objects:	Object Identifier	Object Name	Object Type
	(File, Instance 2)	ChillerTrendLog	FILE

Service = AtomicReadFile
'File Identifier' = (File, Instance 2)
'Record Access':
'File Start Record' = 14
'Requested Record Count' = 3

The 'Result(+)' parameters would be:

'End Of File' = TRUE
'Record Access':
'File Start Record' = 14
'Returned Record Count' = 2
'File Record Data' = ((12:00,45.6), (12:15,44.8))

Notice that in this example not all of the requested data are returned.

E.2.2 Examples of the AtomicWriteFile Service

Example 1: Write data to a file.

Assumed objects:	Object Identifier	Object Name	Object Type
	(File, Instance 1)	ChillerData	FILE

Service = AtomicWriteFile
'File Identifier' = (File, Instance 1)
'Stream Access':
'File Start Position' = 30
'File Data' = "Chiller01 On-Time=4.3 Hours"

The 'Result(+)' parameters would be:

'Stream Access':
'File Start Position' = 30

Example 2: Append two records to a file.

Assumed objects:	Object Identifier	Object Name	Object Type
	(File, Instance 2)	ChillerTrendLog	FILE

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Service = AtomicWriteFile
'File Identifier' = (File, Instance 2)
'Record Access':
'File Start Record' = -1
'Record Count' = 2
'File Record Data' = ((12:00,45.6),
(12:15,44.8))

The 'Result(+)' parameters would be:

'Record Access':
'File Start Record' = 14

E.3 Object Access Services

E.3.1 Example of the AddListElement Service

Example 1. Adding members to a group object.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Group, Instance 3)	AHU1_GRAPH	GROUP
	(Analog Input, Instance 9)	AHU1_SA_TEMP	ANALOG_INPUT
	(Analog Input, Instance 10)	AHU1_RA_TEMP	ANALOG_INPUT
	(Analog Input, Instance 11)	AHU1_OAD_POS	ANALOG_INPUT
	(Analog Input, Instance 12)	AHU1_SA_PRESS	ANALOG_INPUT
	(Analog Input, Instance 13)	AHU1_CW_VALVE	ANALOG_INPUT
	(Analog Input, Instance 14)	OA_TEMP	ANALOG_INPUT
	(Analog Input, Instance 15)	OA_HUMID	ANALOG_INPUT

Consider a BACnet device that contains the following group object used for a graphic display:

Property: Object_Identifier = (Group, Instance 3)
Property: Object_Name = "AHU1_GRAPH"
Property: Object_Type = GROUP
Property: Description = "Points for AHU1 graphic"
Property: List_Of_Group_Members = (((Analog Input, Instance 9), (Present_Value, Reliability)),
((Analog Input, Instance 10), (Present_Value, Reliability)),
((Analog Input, Instance 11), (Present_Value, Reliability)),
((Analog Input, Instance 12), (Present_Value, Reliability, Description)),
((Analog Input, Instance 13), (Present_Value, Reliability, Description)),
((Analog Input, Instance 14), (Present_Value)))
Property: Present_Value = (65.2, NO_FAULT_DETECTED, 72.4, NO_FAULT_DETECTED, 99,
NO_FAULT_DETECTED, 0.67, NO_FAULT_DETECTED, "Inches of water", 32,
NO_FAULT_DETECTED, " % open", 68.3)

The system operator has decided to upgrade the control software in AHU1 to use an enthalpy economizer cycle. As a result, the operator wants to add a humidity reading to "AHU1_GRAPH". The AddListElement Service primitive is used with the following parameters:

Service = AddListElement
'Object Identifier' = (Group, Instance 3)
'Property Identifier' = List_Of_Group_Members
'List of Elements' = ((Analog Input, Instance 15),(Present_Value, Reliability))

Assuming the service request succeeds, a 'Result(+)' service primitive will be issued and the object "AHU1_GRAPH" now has the properties:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Property: Object_Identifier =	(Group, Instance 3)
Property: Object_Name =	"AHU1_GRAPH"
Property: Object_Type =	GROUP
Property: Description =	"Points for AHU1 graphic"
Property: List_Of_Group_Members =	(((Analog Input, Instance 9), (Present_Value, Reliability)), ((Analog Input, Instance 10), (Present_Value, Reliability)), ((Analog Input, Instance 11), (Present_Value, Reliability)), ((Analog Input, Instance 12), (Present_Value, Reliability, Description)), ((Analog Input, Instance 13), (Present_Value, Reliability, Description)), ((Analog Input, Instance 14), (Present_Value)), ((Analog Input, Instance 15), (Present_Value, Reliability)))
Property: Present_Value =	(65.2, NO_FAULT_DETECTED, 72.4, NO_FAULT_DETECTED, 99, NO_FAULT_DETECTED, 0.67, NO_FAULT_DETECTED, "Inches of water", 32, NO_FAULT_DETECTED, " % open", 68.3, 42.1, NO_FAULT_DETECTED)

NOTE: In this example the new element was added at the end of the list. This is a logical place because the list must be traversed to determine if the "new" element already exists. This standard does not require adding new list elements at the end.

E.3.2 Example of the RemoveListElement Service

Example 1: Removing a member of a group.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Group, Instance 3)	AHU1_GRAPH	GROUP
	(Analog Input, Instance 9)	AHU1_SA_TEMP	ANALOG_INPUT
	(Analog Input, Instance 10)	AHU1_RA_TEMP	ANALOG_INPUT
	(Analog Input, Instance 11)	AHU1_OAD_POS	ANALOG_INPUT
	(Analog Input, Instance 12)	AHU1_SA_PRESS	ANALOG_INPUT
	(Analog Input, Instance 13)	AHU1_CW_VALVE	ANALOG_INPUT
	(Analog Input, Instance 14)	OA_TEMP	ANALOG_INPUT

This is an example of using the RemoveListElement Service to change an existing group object. Assume that a group object "AHU1_GRAPH" is defined as:

Property: Object_Identifier =	(Group, Instance 3)
Property: Object_Name =	"AHU1_GRAPH"
Property: Object_Type =	GROUP
Property: List_Of_Group_Members =	(((Analog Input, Instance 9), (Present_Value, Reliability)), ((Analog Input, Instance 10), (Present_Value, Reliability)), ((Analog Input, Instance 11), (Present_Value, Reliability)), ((Analog Input, Instance 12), (Present_Value, Reliability, Description)), ((Analog Input, Instance 13), (Present_Value, Reliability, Description)), ((Analog Input, Instance 14),(Present_Value)))
Property: Present_Value =	(65.2, NO_FAULT_DETECTED, 72.4, NO_FAULT_DETECTED, 99.0, NO_FAULT_DETECTED, 0.67, NO_FAULT_DETECTED, "Inches of water", 32.0, NO_FAULT_DETECTED, % open", 68.3)

A system operator is updating graphic displays and decides that the Description properties in this group are not really used and wishes to remove them. Even though Description is an element of a property list, it cannot be removed by this service because it is nested inside the List_Of_Group_Members. A two step process is required as shown below.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The following service request is issued:

Service = RemoveListElement
'Object Identifier' = (Group, Instance 3)
'Property Identifier' = "List_Of_Elements"
'List of Elements' = (((Analog Input, Instance 12), (Present_Value, Reliability, Description)),
((Analog Input, Instance 13), (Present_Value, Reliability, Description)))

This service request is successful and the status of the object "AHU1_GRAPH" at this point is:

Property: Object_Identifier = (Group, Instance 3)
Property: Object_Name = "AHU1_GRAPH"
Property: Object_Type = GROUP
Property: List_Of_Group_Members = (((Analog Input, Instance 9), (Present_Value, Reliability)),
((Analog Input, Instance 10), (Present_Value, Reliability)),
((Analog Input, Instance 11), (Present_Value, Reliability)),
((Analog Input, Instance 14), (Present_Value)))
Property: Present_Value = (65.2, NO_FAULT_DETECTED, 72.4, NO_FAULT_DETECTED, 99.0,
NO_FAULT_DETECTED, 68.3)

The AddListElement service is now used to replace the group members that were removed but are still needed for the graphic display.

The following service request is issued:

Service = AddListElement
'Object Identifier' = (Group, Instance 3)
'Property Identifier' = "List_Of_Group_Members"
'List of Elements' = (((Analog Input, Instance 12), (Present_Value, Reliability)),
((Analog Input, Instance 13), (Present_Value, Reliability)))

This service request is successful and the "AHU1_GRAPH" is now in the desired form:

Property: Object_Identifier = (Group, Instance 3)
Property: Object_Name = "AHU1_GRAPH"
Property: Object_Type = GROUP
Property: List_Of_Group_Members = (((Analog Input, Instance 9), (Present_Value, Reliability)),
((Analog Input, Instance 10), (Present_Value, Reliability)),
((Analog Input, Instance 11), (Present_Value, Reliability)),
((Analog Input, Instance 14), (Present_Value))
((Analog Input, Instance 12), (Present_Value, Reliability)),
((Analog Input, Instance 13), (Present_Value, Reliability)))
Property: Present_Value = (65.2, NO_FAULT_DETECTED, 72.4, NO_FAULT_DETECTED, 99.0,
NO_FAULT_DETECTED, 68.3, 0.67, NO_FAULT_DETECTED, 32.0,
NO_FAULT_DETECTED)

E.3.3 Example of the CreateObject Service

This is an example of using the CreateObject service to create a file object to be used for a trend log.

Service = CreateObject
'Object Type' = FILE
'List of Initial Values' = ((Object_Name, "Trend 1"), (File_Access_Method, RECORD_ACCESS))

Assuming that the CreateObject service request was correctly received and the device has the ability to create a file object, a 'Result(+)' will be returned conveying the Object_Identifier of the newly created object.

'Object_Identifier' = (File, Instance 13)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Note that in this example only the File_Type and File_Access_Method properties of the new object are initialized as a side effect of the object creation. The other properties of the object will contain default values provided by the vendor. If these default values are not acceptable, then initial values for these properties may be included in the 'List of Initial Values' for the CreateObject service. Alternatively, any writable properties of the object may be initialized or written to via a WriteProperty or WritePropertyMultiple service request after the object is created.

E.3.4 Example of the DeleteObject Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Group, Instance 6)	ZONE1_TEMPS	GROUP
	(Group, Instance 7)	NotDeletable	GROUP

Consider a BACnet device that contains two group objects, "ZONE1_TEMPS" and "NotDeletable". The object "NotDeleteable" was created and protected at configuration time and may not be deleted by this protocol service.

Example 1: The successful deletion of an object.

The following request service primitive is issued:

Service = DeleteObject
'Object Identifier' = (Group, Instance 6)

The object is deleted and the server issues a 'Result(+)' response primitive.

Example 2: An unsuccessful attempt to delete a group object.

The following request service primitive is issued:

Service = DeleteObject
'Object Identifier' = (Group, Instance 7)

This object is protected and cannot be deleted by this protocol service. The server issues the following response primitive.

'Result(-)'
'Error Type' = (SERVICES, OBJECT_NOT_DELETABLE)

E.3.5 Example of the ReadProperty Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 5)	SPACE_TEMP	ANALOG_INPUT

We wish to read the present value for an analog input called "SPACE_TEMP".

Service = ReadProperty
'ObjectIdentifier' = (Analog Input, Instance 5)
'PropertyIdentifier' = Present_Value

Assuming the target machine can locate the object with this ID and the requested properties, the result would be:

'ObjectIdentifier' = (Analog Input, Instance 5)
'PropertyIdentifier' = Present_Value
'Value' = 72.3

The result indicates that the present value of "SPACE_TEMP" is 72.3.

E.3.6 Deleted Clause

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

E.3.7 Examples of the ReadPropertyMultiple Service

Example 1: Parameters for reading multiple properties of a single object.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 16)	SPACE_TEMP	ANALOG_INPUT

We wish to read the Present_Value and Reliability of an analog point called "SPACE_TEMP".

Service = ReadPropertyMultiple
'List of Read Access Specifications' = ((Analog Input, Instance 16), (Present_Value, Reliability))

Assuming the target machine can locate the object with this identifier and the requested properties, the result would be:

'List of Read Access Results' = ((Analog Input, Instance 16), ((Present_Value, 72.3),
(Reliability, NO_FAULT_DETECTED)))

The result indicates that the present value of "SPACE_TEMP" is 72.3 and that it is reliable so far as the device can determine.

Example 2: Parameters for reading the properties of several objects.

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Input, Instance 33)	CW_STEMP	ANALOG_INPUT
	(Analog Input, Instance 34)	CW_RTEMP	ANALOG_INPUT
	(Analog Input, Instance 35)	CW_FLOW	ANALOG_INPUT

Suppose we wish to access the present values of three analog inputs in a chilled water system whose properties are grouped together in objects called "CW_STEMP", "CW_RTEMP", and "CW_FLOW".

Service = ReadPropertyMultiple
'List of Read Access Specifications' = (((Analog Input, Instance 33), (Present_Value)),
((Analog Input, Instance 50), (Present_Value)),
((Analog Input, Instance 35), (Present_Value)))

Assuming successful access of two out of the three requested values:

'List of Read Access Results' = (((Analog Input, Instance 33), (Present_Value, 42.3)),
((Analog Input, Instance 50), (Present_Value, 'Property Access Error' =
(OBJECT, UNKNOWN_OBJECT))),
((Analog Input, Instance 35), (Present_Value, 435.7)))

This result indicates that the supply temperature was successfully accessed and is currently 42.3, and the flow is 435.7. But access to the present value of the return temperature failed. The error type related to this part of the request is 'Error Class' OBJECT and the specific 'Error Code' is UNKNOWN_OBJECT. This occurred because the object identifier was incorrect, possibly due to an operator error.

E.3.8 Example of the ReadRange Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Trend Log, Instance 1)	ROOM3TEMP	TREND_LOG

We wish to look at the next 4 records within a Trend Log's Log Buffer starting at 23-MAR-1998, 19:52:34.00.
The Trend Log's Log Buffer currently only holds 2 Entries which are newer than 23-MAR-1998, 19:52:34.00.

Service = ReadRange
'ObjectIdentifier' = (Trend Log, Instance 1)
'PropertyIdentifier' = Log_Buffer

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

'Range'
'By Time'
'Reference Time' = (23-MAR-1998, 19:52:34.00)
'Count' = 4

A typical result might be:

'Result Flags' = (TRUE, TRUE, FALSE)
'Item Count' = 2
'Item Data' = (((23-MAR-1998, 19:54:27.0), 18.0, (FALSE, FALSE, FALSE, FALSE)),
((23-MAR-1998, 19:56:27.0), 18.1, (FALSE, FALSE, FALSE, FALSE)))
'First Sequence Number' = 79201

E.3.9 Example of the WriteProperty Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Value, Instance 1)	HW_Setpoint	ANALOG_VALUE

We wish to modify the Present_Value of an Analog Value object with an Object_Name of "HW_Setpoint".

Service = WriteProperty
'ObjectIdentifier' = (Analog Value, Instance 1)
'PropertyIdentifier' = Present_Value
'PropertyValue' = 180.0

Assuming the target machine can locate the object with the specified Object_Identifier and modify the Present_Value property, the result would be the issuance of a 'Result(+)' primitive with no additional parameters. This acknowledgment would be conveyed to the service requester in a BACnet-SimpleACK-PDU without parameters.

E.3.10 Example of the WritePropertyMultiple Service

Assumed objects:	Object Identifier	Object Name	Object Type
	(Analog Value, Instance 5)	Room 1 Temp Setpoint	ANALOG_VALUE
	(Analog Value, Instance 6)	Room 2 Temp Setpoint	ANALOG_VALUE
	(Analog Value, Instance 7)	Room 3 Temp Setpoint	ANALOG_VALUE

We wish to modify the Present_Value of three analog variable objects used in the control of three space temperature loops.

Service = WritePropertyMultiple
'List of Write Access Specifications' = (((Analog Value, Instance 5), (Present_Value, 67.0)),
((Analog Value, Instance 6), (Present_Value, 67.0)),
((Analog Value, Instance 7)', (Present_Value, 72.0)))

Assuming the responding BACnet-user can locate each of the three objects, the result would be the replacement of the former values of each of these properties with the provided values and the issuance of a 'Result(+)' primitive with no additional parameters. This acknowledgment would be conveyed to the service requester in a BACnet-SimpleACK-PDU without parameters.

E.3.11 Examples of WriteGroup Service

Example #1: We wish to set control group 23 channel 268=1111, channel 269=2222, priority for writing is 8.

Service = WriteGroup
'Group Number' = 23
'Write Priority' = 8
'Change List' = ((268,,1111),(269,,2222))

Example #2: We wish to set control group 23 channel 12=67.0, channel 13=72.0, priority for writing is 8, inhibit execution delays.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Service = WriteGroup
'Group Number' = 23
'Write Priority' = 8
'Change List' = ((12,,67.0),(13,,72.0))
'Inhibit Delay' = TRUE

Example #3: We wish to set control group 23 channel 12=1111 at priority 8, channel 13="ABC" at priority 10.

Service = WriteGroup
'Group Number' = 23
'Write Priority' = 8
'Change List' = ((12,,1111),(13,10,"ABC"))

E.3.12 Example of the ReadPropertyMultiple Service OPTIONAL response if there are no optional properties

Parameters for reading OPTIONAL properties of a single object:

Assumed objects:	Object Identifier	Object Type
	(Analog Input, Instance 19)	ANALOG_INPUT

Service = ReadPropertyMultiple
'List of Read Access Specifications' = ((Analog Input, Instance 19), (OPTIONAL))

Assuming that (Analog Input, 19) exists and contains no optional properties, the result would be:

'List of Read Access Results' = ( )

E.4 Remote Device Management Services

E.4.1 An Example of the DeviceCommunicationControl Service

While troubleshooting a problem on a BACnet network, it becomes necessary to stop communication exchanges from a particular device for a period of five minutes. This is accomplished by means of the DeviceCommunicationControl Service as follows.

Service = DeviceCommunicationControl
'Time Duration' = 5
'Enable/Disable' = DISABLE_INITIATION
'Password' = "#egbdfl"

E.4.2 An Example of the ConfirmedPrivateTransfer Service

This is an example of a typical proprietary service request using a BACnet confirmed service.

Service = ConfirmedPrivateTransfer
'VendorID' = 25
'Service Number' = 8
'ServiceParameters' = (72.4, X'1649')

Assuming that the service is successfully executed but no results need to be returned to the requesting BACnet-user, a 'Result(+)' primitive will be returned conveying the following information.

'VendorID' = 25
'Service Number' = 8
'Result Block' = NIL

E.4.3 An Example of the UnconfirmedPrivateTransfer Service

This is an example of a typical proprietary service request using a BACnet unconfirmed service.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Service = UnconfirmedPrivateTransfer
'VendorID' = 18
'Service Number' = 12
'ServiceParameters' = (72.4, X'1649')

Since this is an unconfirmed service, no response is expected.

E.4.4 Example of the ReinitializeDevice Service

This example illustrates the use of the ReinitializeDevice service for requesting a WARMSTART for a device that requires a password with this service.

Service = ReinitializeDevice
'Reinitialized State of Device' = WARMSTART
'Password' = "AbCdEfGh"

If the password is successfully validated, a 'Result (+)' primitive will be returned and then the device will reinitialize. Otherwise, a 'Result (-)' primitive will be returned.

E.4.5 Examples of the ConfirmedTextMessageService

This example illustrates the use of the ConfirmedTextMessage service.

Service = ConfirmedTextMessage
'Text Message Source Device' = (Device, Instance 5)
'Message Priority' = NORMAL
'Message' = "P.M. required for PUMP347"

If the ConfirmedTextMessage-Request was successfully received, a 'Result(+)' will be returned. Otherwise, a 'Result(-)' will be returned.

E.4.6 Examples of the UnconfirmedTextMessage Service

This example illustrates the use of the UnconfirmedTextMessage service.

Service = UnconfirmedTextMessage
'Text Message Source Device' = (Device, Instance 5)
'Message Priority' = NORMAL
'Message' = "P.M. required for PUMP347"

E.4.7 Example of the TimeSynchronization Service

An example of parameter usage for the TimeSynchronization service follows.

Service = TimeSynchronization
'Time'
'Date' = 17-Nov-92
'Time' = 22:45:30.7

This request will notify all remote devices that the current time is 30.7 seconds past 10:45 P.M. on November 17, 1992. The remote devices may use this information to update their internal clocks so that all devices will be synchronized.

E.4.8 Examples of the Who-Has and I-Have Services

Examples of the parameter usage for the Who-Has and I-Have services follow.

Example 1: Locating the device that contains an object for which the Object_Name is known.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Consider an attempt to locate an object with an object name of "OATemp":

Service = Who-Has
'Object Name' = "OATemp"

Assuming that there is exactly one device that has such an object, the following I-Have service indication would be received:

Service = I-Have
'Device Identifier' = (Device, Instance 8)
'Object Identifier' = (Analog Input, Instance 3)
'Object Name' = "OATemp"

Example 2: Locating the device that contains an object for which the Object_Identifier is known.

Consider an attempt to locate an object with an Object_Identifier of (Analog Input, Instance 3).

Service = Who-Has
'Object Identifier' = (Analog Input, Instance 3)

Assuming that there is exactly one device that has such an object, the following I-Have service indication would be received:

Service = I-Have
'Device Identifier' = (Device, Instance 8)
'Object Identifier' = (Analog Input, Instance 3)
'Object Name' = "OATemp"

E.4.9 Examples of the Who-Is and I-Am Services

Examples of parameter usage for the Who-Is and I-Am services follow.

Example 1: Establishing the network address of a device.

We wish to determine the network address of another BACnet device, but only its Device Identifier is known.

Service = Who-Is
'Who-Is Device Identifier' = (Device, Instance 3)

Assuming that there is such a device on the network, it responds sometime later using the I-Am service:

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 3)
'Max APDU Length Accepted' = 1024
'Segmentation Supported' = NO_SEGMENTATION
'Vendor ID' = 99

The MAC layer source address combined with the network layer SNET and SADR provide all of the addressing information needed to communicate with the device (Device, Instance 3).

Example 2: Finding out about all network devices.

Suppose we wish to determine which devices are currently on the local network.

Service = Who-Is Service

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Each of the devices on the network answers, resulting in several I-Am service requests:

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 1)
'Max APDU Length Accepted' = 480
'Segmentation Supported' = SEGMENTED_TRANSMIT
'Vendor ID' = 99

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 2)
'Max APDU Length Accepted' = 206
'Segmentation Supported' = SEGMENTED_RECEIVE
'Vendor ID' = 33

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 3)
'Max APDU Length Accepted' = 1024
'Segmentation Supported' = NO_SEGMENTATION
'Vendor ID' = 99

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 4)
'Max APDU Length Accepted' = 128
'Segmentation Supported' = SEGMENTED_BOTH
'Vendor ID' = 66

This procedure can be expanded to find out about all devices in a BACnet internet by including the global network address (X'FFFF') in the network layer header.

E.4.10 Examples of the Who-Am-I and You-Are Services

Examples of parameter usage for the Who-Am-I and You-Are services follow.
Example 1: Assigning the network address and Device Identifier of a device.

We wish to assign the network address and Device Identifier of another BACnet Device, and only its Vendor Identifier, Model Name, and Serial Number are known.

Service = You-Are
'Vendor ID' = 555
'Model Name' = "LMCP24"
'Serial Number' = "12345"
'Device Identifier' = (Device, Instance 3)
'Device MAC Address' = X'2A'

Assuming that there is such a device on the network, it responds sometime later using the I-Am service:

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 3)
'Max APDU Length Accepted' = 480
'Segmentation Supported' = NO_SEGMENTATION
'Vendor Identifier' = 555

Example 2: A device needs its Device Identifier configured, and only its Vendor Identifier, Model Name, and Serial Number are known.

Service = Who-Am-I Service
'Vendor ID' = 555
'Model Name' = "LMCP24"
'Serial Number' = "12345"

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Another device has a list, for a number of devices, of a Device Identifier, MAC address, Vendor Identifier, Model Name, and Serial Number for each device in the list. This device has the responsibility for handling the assignment of Device Identifier of the requesting device answers, resulting in a You-Are service request:

Service = You-Are
'Vendor ID' = 555
'Model Name' = "LMCP24"
'Serial Number' = "12345"
'Device Identifier' = (Device, Instance 3)

Assuming that there is such a device on the network, it responds sometime later using the I-Am service:

Service = I-Am
'I-Am Device Identifier' = (Device, Instance 3)
'Max APDU Length Accepted' = 480
'Segmentation Supported' = NO_SEGMENTATION
'Vendor Identifier' = 555

E.5 Virtual Terminal Services

Examples of parameter usage for the VT-Open, VT-Data, and VT-Close services follow. For readability purposes, some of the examples include imbedded control characters, such as carriage return. These are shown within text strings enclosed in curly braces, e.g., "{cr}{lf}text".

Establishing a Session with a Default Terminal:

We wish to create a VT-session with the Default-terminal style of operator interface in another BACnet device.

Service = VT-Open
'VT-class' = ANSI_X3.64
'Local VT Session Identifier' = 5

Assuming that the target machine can create a new VT-session, the 'Result (+)' response might be:

'Remote VT Session Identifier' = 29

The result indicates that target device has accepted the request and created a new session.

Now that a session is established, the operator interface program in the remote device prompts us to sign-on by issuing a VT-Data request:

Service = VT-Data
'VT-session Identifier' = 5
'VT-new Data' = "{cr}{lf}Enter User Name:"
'VT-data Flag' = 0

Our operator interface display queue is empty so it can accept all of the incoming characters. Our VT-User therefore issues a 'Result (+)':

'All New Data Accepted' = TRUE

Eventually, our human operator enters his or her name in response to the prompt, causing a VT-Data request to be issued:

Service = VT-Data
'VT-session Identifier' = 29
'VT-new Data' = "FRED{cr}"
'VT-data Flag' = 0

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The remote operator interface accepts the incoming characters as well and responds with 'Result (+)':

'All New Data Accepted' = TRUE

Then the remote operator interface echoes the entered characters and another prompt:

Service = VT-Data
'VT-session Identifier' = 5
'VT-new Data' = "FRED{cr}{lf}Enter Password:"
'VT-data Flag' = 1

Our operator interface display queue is empty so it can accept all of the incoming characters. Our VT-User therefore issues 'Result (+)':

'All New Data Accepted' = TRUE

For some reason, FRED decides to cancel this virtual terminal session and signals the operator interface program to do so. The operator interface program issues a VT-Close request:

Service = VT-Close
'List of Remote VT Session Identifiers' = (29)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX F - EXAMPLES OF APDU ENCODING (INFORMATIVE)

(This annex is not part of the standard but is included for informative purposes only.)

This annex illustrates the use of BACnet encoding rules by showing the encoded APDUs for the examples in Annex E.

F.1 Example Encodings for Alarm and Event Services

F.1.1 Encoding for Example E.1.1 - AcknowledgeAlarm Service

The encoding for example E.1.1 is included in Clauses F.1.4 and F.1.5.

F.1.2 Encoding for Example E.1.2 - ConfirmedCOVNotification Service

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                       Maximum APDU Size Accepted=206 octets
X'0F'                       Invoke ID=15
X'01'                       Service Choice=1 (ConfirmedCOVNotification-Request)

X'09'                       SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'                       Subscriber Process Identifier=18
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000004'                 Device, Instance 4
X'2C'                       SD Context Tag 2 (Monitored Object Identifier, L=4)
X'0000000A'                 Analog Input, Instance Number=10
X'39'                       SD Context Tag 3 (Time Remaining, L=1)
X'00'                       Time Remaining=0
X'4E'                       PD Opening Tag 4 (List of Values)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)
        X'2E'                       PD Opening Tag 2 (Value)
                X'44'                       Application Tag 4 (Real, L=4)
                X'42820000'                 65.0
        X'2F'                       PD Closing Tag 2 (Value)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'6F'                       111 (STATUS_FLAGS)
        X'2E'                       PD Opening Tag 2 (Value)
                X'82'                       Application Tag 8 (Bit String, L=2)
                X'0400'                     0,0,0,0 (FALSE, FALSE, FALSE, FALSE)
        X'2F'                       PD Closing Tag 2 (Value)
X'4F'                       PD Closing Tag 4 (List Of Values)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                       PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'                       Invoke ID=15
X'01'                       Service ACK Choice=1 (ConfirmedCOVNotification)
```

F.1.3 Encoding for Example E.1.3 - UnconfirmedCOVNotification Service

```
X'10'                       PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'02'                       Service Choice=2 (UnconfirmedCOVNotification-Request)

X'09'                       SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'                       18
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000004'                 Device, Instance Number=4
X'2C'                       SD Context Tag 2 (Monitored Object Identifier, L=4)
X'0000000A'                 Analog Input, Instance Number=10
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'39'                       SD Context Tag 3 (Time Remaining, L=1)
X'00'                       0
X'4E'                       PD Opening Tag 4 (List of Values)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)
        X'2E'                       PD Opening Tag 2 (Value)
                X'44'                       Application Tag 4 (Real, L=4)
                X'42820000'                 65.0
        X'2F'                       PD Closing Tag 2 (Value)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'6F'                       111 (STATUS_FLAGS)
        X'2E'                       PD Opening Tag 2 (Value)
                X'82'                       Application Tag 8 (Bit String, L=2)
                X'0400'                     0,0,0,0 (FALSE, FALSE, FALSE, FALSE)
        X'2F'                       PD Closing Tag 2 (Value)
X'4F'                       PD Closing Tag 4 (List Of Values)
```

F.1.4 Encoding for Example E.1.4 - ConfirmedEventNotification Service

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                       Maximum APDU Size Accepted=206 octets
X'10'                       Invoke ID=16
X'02'                       Service Choice=2 (ConfirmedEventNotification-Request)
X'09'                       SD Context Tag 0 (Process Identifier, L=1)
X'01'                       Process Identifier=1
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000004'                 Device, Instance 4
X'2C'                       SD Context Tag 2 (Event Object Identifier, L=4)
X'00000002'                 Analog Input, Instance Number=2
X'3E'                       PD Opening Tag 3 (Time Stamp)
        X'19'                       SD Context Tag 1 (SequenceNumber, L=1)
        X'10'                       16
X'3F'                       PD Closing Tag 3 (Time Stamp)
X'49'                       SD Context Tag 4 (Notification Class, L=1)
X'04'                       4
X'59'                       SD Context Tag 5 (Priority, L=1)
X'64'                       100
X'69'                       SD Context Tag 6 (Event Type, L=1)
X'05'                       5 (OUT_OF_RANGE)
X'89'                       SD Context Tag 8 (Notify Type, L=1)
X'00'                       0 (ALARM)
X'99'                       SD Context Tag 9 (AckRequired, L=1)
X'01'                       TRUE
X'A9'                       SD Context Tag 10 (From State, L=1)
X'00'                       0 (NORMAL)
X'B9'                       SD Context Tag 11 (To State, L=1)
X'03'                       3 (HIGH_LIMIT)
X'CE'                       PD Opening Tag 12 (Event Values)
        X'5E'                       PD Opening Tag 5 (BACnetNotificationParameters, Choice
5=OUT_OF_RANGE)
                X'0C'                       SD Context Tag 0 (Exceeding Value, L=4)
                X'42A03333'                 80.1
                X'1A'                       SD Context Tag 1 (Status Flags, L=2)
                X'0480'                     1,0,0,0 (TRUE, FALSE, FALSE, FALSE)
                X'2C'                       SD Context Tag 2 (Deadband, L=4)
                X'3F800000'                 1.0
                X'3C'                       SD Context Tag 3 (Exceeded Limit, L=4)
                X'42A00000'                 80.0
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        X'5F'                       PD Closing Tag 5 (BACnetNotificationParameters)
X'CF'                       PD Closing Tag 12 (Event Values)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                       PDU Type=2 (BACnet-SimpleACK-PDU)
X'10'                       Invoke ID=16
X'02'                       Service ACK Choice=2 (ConfirmedEventNotification)
```

At some future time, an AcknowledgeAlarm-Request is generated:

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                       Maximum APDU Size Accepted=206 octets
X'07'                       Invoke ID=7
X'00'                       Service Choice=0 (AcknowledgeAlarm-Request)

X'09'                       SD Context Tag 0 (Acknowledging Process Identifier, L=1)
X'01'                       Acknowledging Process Identifier=1
X'1C'                       SD Context Tag 1 (Event Object Identifier, L=4)
X'00000002'                 Analog Input, Instance Number=2
X'29'                       SD Context Tag 2 (Event State Acknowledged, L=1)
X'03'                       3 (HIGH_LIMIT)
X'3E'                       PD Opening Tag 3 (Time Stamp)
        X'19'                       SD Context Tag 1 (Sequence Number, L=1)
        X'10'                       16
X'3F'                       PD Closing Tag 3 (Time Stamp)
X'4C'                       SD Context Tag 4 (Acknowledgment Source, L=4)
X'00'                       ISO 10646 (UTF-8) Encoding
X'4D444C'                   "MDL"
X'5E'                       PD Opening Tag 5 (Time Of Acknowledgment)
        X'2E'                       PD Opening Tag 2 (Date Time)
                X'A4'                       Application Tag 10 (Date, L=4)
                X'5C0615FF'                 June 21, 1992
                X'B4'                       Application Tag 11 (Time, L=4)
                X'0D032909'                 13:03:41.9
        X'2F'                       PD Closing Tag 2 (Date Time)
X'5F'                       PD Closing Tag 5 (Time Of Acknowledgment)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                       PDU Type=2 (BACnet-SimpleACK-PDU)
X'07'                       Invoke ID=7
X'00'                       Service ACK Choice=0 (AcknowledgeAlarm)
```

F.1.5 Encoding for Example E.1.5 - UnconfirmedEventNotification Service

```
X'10'                       PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'03'                       Service Choice=3 (UnconfirmedEventNotification-Request)

X'09'                       SD Context Tag 0 (Process Identifier, L=1)
X'01'                       1
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000009'                 Device, Instance Number=9
X'2C'                       SD Context Tag 2 (Event Object Identifier, L=4)
X'00000002'                 Analog Input, Instance Number=2
X'3E'                       PD Opening Tag 3 (Time Stamp)
        X'19'                       SD Context Tag 1 (SequenceNumber, L=1)
        X'10'                       16
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'3F'                          PD Closing Tag 3 (Time Stamp)
X'49'                          SD Context Tag 4 (Notification Class, L=1)
X'04'                          4
X'59'                          SD Context Tag 5 (Priority, L=1)
X'64'                          100
X'69'                          SD Context Tag 6 (Event Type, L=1)
X'05'                          5 (OUT_OF_RANGE)
X'89'                          SD Context Tag 8 (Notify Type, L=1)
X'00'                          0 (ALARM)
X'99'                          SD Context Tag 9 (AckRequired, L=1)
X'01'                          1 (TRUE)
X'A9'                          SD Context Tag 10 (From State, L=1)
X'00'                          0 (NORMAL)
X'B9'                          SD Context Tag 11 (To State, L=1)
X'03'                          3 (HIGH_LIMIT)
X'CE'                          PD Opening Tag 12 (Event Values)
        X'5E'                          PD Opening Tag 5 (BACnetNotificationParameters, Choice
5=OUT_OF_RANGE)
                X'0C'                          SD Context Tag 0 (Exceeding Value, L=4)
                X'42A03333'                    80.1
                X'1A'                          SD Context Tag 1 (Status Flags, L=2)
                X'0480'                        1,0,0,0 (TRUE, FALSE, FALSE, FALSE)
                X'2C'                          SD Context Tag 2 (Deadband, L=4)
                X'3F800000'                    1.0
                X'3C'                          SD Context Tag 3 (Exceeded Limit, L=4)
                X'42A00000'                    80.0
        X'5F'                          PD Closing Tag 5 (BACnetNotificationParameters)
X'CF'                          PD Closing Tag 12 (Event Values)
```

At some future time, an AcknowledgeAlarm-Request is generated:

```
X'00'                          PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                          Maximum APDU Size Accepted=206 octets
X'07'                          Invoke ID=7
X'00'                          Service Choice=0 (AcknowledgeAlarm-Request)

X'09'                          SD Context Tag 0 (Acknowledging Process Identifier, L=1)
X'01'                          Acknowledging Process Identifier=1
X'1C'                          SD Context Tag 1 (Event Object Identifier, L=4)
X'00000002'                    Analog Input, Instance Number=2
X'29'                          SD Context Tag 2 (Event State Acknowledged, L=1)
X'03'                          3 (HIGH_LIMIT)
X'3E'                          PD Opening Tag 3 (Time Stamp)
        X'19'                          SD Context Tag 1 (Sequence Number, L=1)
        X'10'                          16
X'3F'                          PD Closing Tag 3 (Time Stamp)
X'4C'                          SD Context Tag 4 (Acknowledgment Source, L=4)
X'00'                          ISO 10646 (UTF-8) Encoding
X'4D444C'              "MDL"
X'5E'                          PD Opening Tag 5 (Time Of Acknowledgment)
        X'2E'                          PD Opening Tag 2 (Date Time)
                X'A4'                          Application Tag 10 (Date, L=4)
                X'5C0615FF'                    June 21, 1992
                X'B4'                          Application Tag 11 (Time, L=4)
                X'0D032909'                    13:03:41.9
        X'2F'                          PD Closing Tag 2 (Date Time)
X'5F'                          PD Closing Tag 5 (Time Of Acknowledgment)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'07'	Invoke ID=7
X'00'	Service ACK Choice=0 (AcknowledgeAlarm)

F.1.6 Encoding for Example E.1.6 - GetAlarmSummary Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets
X'01'	Invoke ID=1
X'03'	Service Choice=3 (GetAlarmSummary-Request)

Assuming the service procedure executes correctly, a complex acknowledgment is returned:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01'	Invoke ID=1
X'03'	Service ACK Choice=3 (GetAlarmSummary-ACK)
X'C4'	Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000002'	Analog Input, Instance Number=2
X'91'	Application Tag 9 (Enumerated, L=1) (Alarm State)
X'03'	3 (HIGH_LIMIT)
X'82'	Application Tag 8 (Bit String, L=2) (Acknowledged Transitions)
X'0560'	0,1,1 (FALSE, TRUE, TRUE)
X'C4'	Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000003'	Analog Input, Instance Number=3
X'91'	Application Tag 9 (Enumerated, L=1) (Alarm State)
X'04'	4 (LOW_LIMIT)
X'82'	Application Tag 8 (Bit String, L=2) (Acknowledged Transitions)
X'05E0'	1,1,1 (TRUE, TRUE, TRUE)

F.1.7 Encoding for Example E.1.7 - GetEnrollmentSummary Service

Example 1: Request encoding.

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets
X'01'	Invoke ID=1
X'04'	Service Choice=4 (GetEnrollmentSummary-Request)
X'09'	SD Context Tag 0 (Acknowledgment Filter, L=1)
X'02'	2 (NOT_ACKED)

Assuming the service procedure executes correctly, a complex acknowledgment is returned containing the requested values:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01'	Invoke ID=1
X'04'	Service ACK Choice=4 (GetEnrollmentSummary-ACK)
X'C4'	Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000002'	Analog Input, Instance Number=2
X'91'	Application Tag 9 (Enumerated, L=1) (Event Type)
X'05'	5 (OUT_OF_RANGE)
X'91'	Application Tag 9 (Enumerated, L=1) (Event State)
X'03'	3 (HIGH_LIMIT)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Priority)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'64'                       100
X'21'                       Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'04'                       4
X'C4'                       Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'02400006'                 Event Enrollment, Instance Number=6
X'91'                       Application Tag 9 (Enumerated, L=1) (Event Type)
X'01'                       1 (CHANGE_OF_STATE)
X'91'                       Application Tag 9 (Enumerated, L=1) (Event State)
X'00'                       0 (NORMAL)
X'21'                       Application Tag 2 (Unsigned Integer, L=1) (Priority)
X'32'                       50
X'21'                       Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'02'                       2
```

Example 2: Request encoding.

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                       Maximum APDU Size Accepted=206 octets
X'02'                       Invoke ID=2
X'04'                       Service Choice=4 (GetEnrollmentSummary-Request)

X'09'                       SD Context Tag 0 (Acknowledgment Filter, L=1)
X'00'                       0 (ALL)
X'1E'                       PD Opening Tag 1 (Enrollment Filter)
    X'0E'                       PD Opening Tag 0 (Recipient)
        X'0C'                       SD Context Tag 0 (Device, L=4)
        X'02000011'                 Device, Instance Number=17
    X'0F'                       PD Closing Tag 0 (Recipient)
    X'19'                       SD Context Tag 1 (Process Identifier)
    X'09'                       9
X'1F'                       PD Closing Tag 1 (Enrollment Filter)
X'4E'                       PD Opening Tag 4 (Priority Filter)
    X'09'                       SD Context Tag 0 (MinPriority, L=1)
    X'06'                       6
    X'19'                       SD Context Tag 1 (MaxPriority, L=1)
    X'0A'                       10
X'4F'                       PD Closing Tag 4 (Priority Filter)
```

Assuming the service procedure executes correctly, a complex acknowledgment is returned containing the requested values:

```
X'30'                       PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'02'                       Invoke ID=2
X'04'                       Service ACK Choice=4 (GetEnrollmentSummary-ACK)

X'C4'                       Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000002'                 Analog Input, Instance Number=2
X'91'                       Application Tag 9 (Enumerated, L=1) (Event Type)
X'05'                       5 (OUT_OF_RANGE)
X'91'                       Application Tag 9 (Enumerated, L=1) (Event State)
X'00'                       0 (NORMAL)
X'21'                       Application Tag 2 (Unsigned Integer, L=1) (Priority)
X'08'                       8
X'21'                       Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'04'                       4

X'C4'                       Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000003'                 Analog Input, Instance Number=3
X'91'                       Application Tag 9 (Enumerated, L=1) (Event Type)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'05' 5 (OUT_OF_RANGE)
X'91' Application Tag 9 (Enumerated, L=1) (Event State)
X'00' 0 (NORMAL)
X'21' Application Tag 2 (Unsigned Integer, L=1) (Priority)
X'08' 8
X'21' Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'04' 4

X'C4' Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000004' Analog Input, Instance Number=4
X'91' Application Tag 9 (Enumerated, L=1) (Event Type)
X'05' 5 (OUT_OF_RANGE)
X'91' Application Tag 9 (Enumerated, L=1) (Event State)
X'00' 0 (NORMAL)
X'21' Application Tag 2 (Unsigned Integer, L=1) (Priority)
X'08' 8
X'21' Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'04' 4

X'C4' Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'02400007' Event Enrollment, Instance Number=7
X'91' Application Tag 9 (Enumerated, L=1) (Event Type)
X'04' 4 (FLOATING_LIMIT)
X'91' Application Tag 9 (Enumerated, L=1) (Event State)
X'00' 0 (NORMAL)
X'21' Application Tag 2 (Unsigned Integer, L=1) (Priority)
X'03' 3
X'21' Application Tag 2 (Unsigned Integer, L=1) (Notification Class)
X'08' 8

F.1.8 Encoding for Example E.1.8 - GetEventInformation Service

X'02' PDU Type = 0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=1)
X'02' Maximum APDU Size Accepted = 206 octets
X'01' Invoke ID = 1
X'1D' Service Choice = 29 (GetEventInformation)

Assuming the service procedure executes correctly, a complex acknowledgment is returned:

X'30' PDU Type = 3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01' Invoke ID=01
X'1D' Service ACK Choice = 29, (GetEventInformation-ACK)

X'0E' PD opening Tag 0
X'0C' SD context Tag 0 (ObjectIdentifier, L=4)
X'00000002' Analog Input, Instance Number = 2
X'19' SD context Tag 1 (Enumerated, L=1)
X'03' 3 (HIGH_LIMIT)
X'2A' SD context Tag 2 (Bit String, L=2)
X'0560' 0,1,1 (FALSE, TRUE, TRUE)
X'3E' PD opening Tag 3
X'0C' SD context Tag 0 (Time L=4)
X'0F230014' Time 15:35:00.20
X'0C' SD context Tag 0 (Time L=4)
X'FFFFFFFF' Time unspecified
X'0C' SD context Tag 0 (Time L=4)
X'FFFFFFFF' Time unspecified
X'3F' PD closing Tag 3
X'49' SD context Tag 4 (Enumerated, L=1)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'00'                   0 (ALARM)
X'5A'                   SD context Tag 5 (Bit String, L=2)
X'05E0'                 1,1,1 (TRUE, TRUE, TRUE)
X'6E'                   PD opening Tag 6
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'0F'               15 (Priority)
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'0F'               15 (Priority)
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'14'               20 (Priority)
X'6F'                   PD closing Tag 6
X'0C'                   SD context Tag 0 (ObjectIdentifier, L=4)
X'00000003'             Analog Input, Instance Number = 3
X'19'                   SD context Tag 1 (Enumerated, L=1)
X'00'                   0 (NORMAL)
X'2A'                   SD context Tag 2 (Bit String, L=2)
X'05C0'                 1,1,0 (TRUE, TRUE, FALSE)
X'3E'                   PD opening Tag 3
    X'0C'               SD context Tag 0 (Time L=4)
    X'0F280000'         Time 15:40:00.00
    X'0C'               SD context Tag 0 (Time L=4)
    X'FFFFFFFF'         Time unspecified
    X'0C'               SD context Tag 0 (Time L=4)
    X'0F2D1E1E'         15:45:30.30
X'3F'                   PD closing Tag 3
X'49'                   SD context Tag 4 (Enumerated, L=1)
X'00'                   0 (ALARM)
X'5A'                   SD context Tag 5 (Bit String, L=2)
X'05E0'                 1,1,1 (TRUE, TRUE, TRUE)
X'6E'                   PD opening Tag 6
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'0F'               15 (Priority)
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'0F'               15 (Priority)
    X'21'               Application Tag 2 (Unsigned Integer, L=1)
    X'14'               20 (Priority)
X'6F'                   PD closing Tag 6

X'0F'               PD closing Tag 0
X'19'               SD context Tag 1 (Boolean, L=1)
X'00'               FALSE (More Events)
```

F.1.9 Encoding for Example E.1.9 - LifeSafetyOperation

```
X'00'           PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'           Maximum APDU Size Accepted=206 octets
X'0F'           Invoke ID=15
X'1B'           Service Choice=27 (LifeSafetyOperation-Request)

X'09'           SD Context Tag 0 (Requesting Process Identifier, L=1)
X'12'           18
X'1C'           SD Context Tag 1 (Requesting Source, L=4)
X'00'                   ISO 10646 (UTF-8) Encoding
X'4D444C'       "MDL"
X'29'           SD Context Tag 2 (Request, L=1)
X'04'           4 (RESET)
X'3C'           SD Context Tag 3 (Object Identifier, L=4)
X'05400001'     Life Safety Point, Instance Number = 1
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'	Invoke ID=15
X'1B'	Service ACK Choice=27 (LifeSafetyOperation)

F.1.10 Encoding for Example E.1.10 - SubscribeCOV

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets
X'0F'	Invoke ID=15
X'05'	Service Choice=5 (SubscribeCOV-Request)
X'09'	SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'	18
X'1C'	SD Context Tag 1 (Monitored Object Identifier, L=4)
X'0000000A'	Analog Input, Instance Number=10
X'29'	SD Context Tag 2 (Issue Confirmed Notifications, L=1)
X'01'	1 (TRUE)
X'39'	SD Context Tag 3 (Lifetime, L=1)
X'00'	0

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'	Invoke ID=15
X'05'	Service ACK Choice=5 (SubscribeCOV)

F.1.11 Encoding for Example E.1.11 - SubscribeCOVProperty

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets
X'0F'	Invoke ID=15
X'1C'	Service Choice=28 (SubscribeCOVProperty-Request)
X'09'	SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'	18
X'1C'	SD Context Tag 1 (Monitored Object Identifier, L=4)
X'0000000A'	Analog Input, Instance Number=10
X'29'	SD Context Tag 2 (Issue Confirmed Notifications, L=1)
X'01'	1 (TRUE)
X'39'	SD Context Tag 3 (Lifetime, L=1)
X'3C'	60
X'4E'	PD Opening Tag 4 (Monitored Property, L=1)
X'09'	SD Context Tag 0 (Property Identifier, L=1)
X'55'	85 (PRESENT_VALUE)
X'4F'	PD Closing Tag 4 (Property Identifier)
X'5C'	SD Context Tag 5 (COV Increment, L=4)
X'3F800000'	1.0

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'	Invoke ID=15
X'1C'	Service ACK Choice=28 (SubscribeCOVProperty)

F.1.12 Encoding for Example E.1.12 - SubscribeCOVPropertyMultiple Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'	Maximum APDU Size Accepted=206 octets

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'0F'                   Invoke ID=15
X'1E'                   Service Choice=30 (SubscribeCOVPropertyMultiple-Request)

X'09'                   SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'                   18
X'19'                   SD Context Tag 1 (Issue Confirmed Notifications, L=1)
X'01'                   1 (TRUE)
X'29'                   SD Context Tag 2 (Lifetime, L=1)
X'3C'                   60
X'39'                   SD Context Tag 3 (Max Notification Delay, L=1)
X'05'                   5
X'4E'                   PD Opening Tag 4 (List of COV Subscription Specifications)
        X'0C'                   SD Context Tag 0 (Monitored Object, L=4)
        X'0000000A'                     Analog Input, Instance Number=10
        X'1E'                           PD Opening Tag 1 (List of COV References)
                X'0E'                           PD Opening Tag 0 (Monitored Property)
                        X'09'                           SD Context Tag 0 (Property Identifier, L=1)
                        X'55'                           85 (PRESENT_VALUE)
                X'0F'                           PD Closing Tag 0 (Monitored Property)
                X'1C'                           SD Context Tag 1 (COV Increment, L=4)
                X'3F800000'                     1.0
                X'29'                           SD Context Tag 2 (Timestamped, L=1)
                X'01'                           TRUE
                X'0E'                           PD Opening Tag 0 (Monitored Property)
                        X'09'                           SD Context Tag 0 (Property Identifier, L=1)
                        X'67'                           103 (RELIABILITY)
                X'0F'                           PD Closing Tag 0 (Monitored Property)
                X'29'                           SD Context Tag 2 (Timestamped, L=1)
                X'00'                           FALSE
        X'1F'                           PD Closing Tag 1 (List of COV References)
        X'0C'                           SD Context Tag 0 (Monitored Object, L=4)
        X'00400008'                     Analog Output, Instance Number=8
        X'1E'                           PD Opening Tag 1 (List of COV References)
                X'0E'                           PD Opening Tag 0 (Monitored Property)
                        X'09'                           SD Context Tag 0 (Property Identifier, L=1)
                        X'55'                           85 (PRESENT_VALUE)
                X'0F'                           PD Closing Tag 0 (Monitored Property)
                X'1C'                           SD Context Tag 1 (COV Increment, L=4)
                X'3F800000'                     1.0
                X'29'                           SD Context Tag 2 (Timestamped, L=1)
                X'01'                           TRUE
        X'1F'                           PD Closing Tag 1 (List of COV References)
X'4F'                   PD Closing Tag 4 (List of COV Subscription Specifications)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                   PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'                   Invoke ID=15
X'1E'                   Service ACK Choice=30 (SubscribeCOVPropertyMultiple)
```

F.1.13 Encoding for Example E.1.13 - ConfirmedCOVNotificationMultiple Service

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                   Maximum APDU Size Accepted=206 octets
X'0F'                   Invoke ID=15
X'1F'                   Service Choice=31 (ConfirmedCOVNotificationMultiple-Request)

X'09'                   SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'                   Subscriber Process Identifier=18
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000004'                 Device, Instance 4
X'29'                       SD Context Tag 2 (Time Remaining, L=1)
X'23'                       35
X'3E'                       PD Opening Tag 3 (Timestamp)
        X'A4'                       Application Tag 10 (Date, L=4)
        X'71060301'                 June 3, 2013 (Day of Week = Monday)
        X'B4'                       Application Tag 11 (Time, L=4)
        X'0317352F'                 03:23:53.47
X'3F'                       PD Closing Tag 3 (Timestamp)
X'4E'                       PD Opening Tag 4 (List of COV Notifications)
        X'0C'                       SD Context Tag 0 (Monitored Object, L=4)
        X'0000000A'                 Analog Input, Instance Number=10
        X'1E'                       PD Opening Tag 1 (List of Values)
                X'09'                       SD Context Tag 0 (Property Identifier, L=1)
                X'55'                       85 (PRESENT_VALUE)
                X'2E'                       PD Opening Tag 2 (Value)
                        X'44'                       Application Tag 4 (Real, L=4)
                        X'42820000'                 65.0
                X'2F'                       PD Closing Tag 2 (Value)
                X'3C'                       SD Context Tag 3 (Time of Change)
                X'03173400'                 03:23:52.00
        X'1F'                       PD Closing Tag 1 (List of Values)
        X'0C'                       SD Context Tag 0 (Monitored Object, L=4)
        X'00400005'                 Analog Output, Instance Number=5
        X'1E'                       PD Opening Tag 1 (List of Values)
                X'09'                       SD Context Tag 0 (Property Identifier, L=1)
                X'55'                       85 (PRESENT_VALUE)
                X'2E'                       PD Opening Tag 2 (Value)
                        X'44'                       Application Tag 4 (Real, L=4)
                        X'42A03333'                 80.1
                X'2F'                       PD Closing Tag 2 (Value)
        X'1F'                       PD Closing Tag 1 (List of Values)
X'4F'                   PD Closing Tag 4 (List of COV Notifications)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                   PDU Type=2 (BACnet-SimpleACK-PDU)
X'0F'                   Invoke ID=15
X'1F'                   Service ACK Choice=31 (ConfirmedCOVNotificationMultiple)
```

F.1.14 Encoding for Example E.1.14 - UnconfirmedCOVNotificationMultiple Service

```
X'10'                       PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'0B'                       Service Choice=11 (UnconfirmedCOVNotificationMultiple-Request)

X'09'                       SD Context Tag 0 (Subscriber Process Identifier, L=1)
X'12'                       Subscriber Process Identifier=18
X'1C'                       SD Context Tag 1 (Initiating Device Identifier, L=4)
X'02000004'                 Device, Instance 4
X'29'                       SD Context Tag 2 (Time Remaining, L=1)
X'1B'                       27
X'4E'                   PD Opening Tag 4 (List of COV Notifications)
        X'0C'                       SD Context Tag 0 (Monitored Object, L=4)
        X'0000000A'                 Analog Input, Instance Number=10
        X'1E'                       PD Opening Tag 1 (List of Values)
                X'09'                       SD Context Tag 0 (Property Identifier, L=1)
                X'55'                       85 (PRESENT_VALUE)
                X'2E'                       PD Opening Tag 2 (Value)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                    X'44'                   Application Tag 4 (Real, L=4)
                    X'42820000'             65.0
                X'2F'                   PD Closing Tag 2 (Value)
            X'1F'                   PD Closing Tag 1 (List of Values)
X'4F'               PD Closing Tag 4 (List of COV Notifications)
```

F.2 Example Encodings for File Access Services

F.2.1 Encoding for Example E.2.1 - AtomicReadFile Service

Example 1: Read data from a file.

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                   Maximum APDU Size Accepted=206 octets
X'00'                   Invoke ID=0
X'06'                   Service Choice=6 (AtomicReadFile-Request)

X'C4'                   Application Tag 12 (Object Identifier, L=4) (File Identifier)
X'02800001'             File, Instance Number=1
X'0E'                   PD Opening Tag 0 (Stream Access)
        X'31'                   Application Tag 3 (Signed Integer, L=1) (File Start Position)
        X'00'                   0
        X'21'                   Application Tag 2 (Unsigned, L=1) (Requested Octet Count)
        X'1B'                   27
X'0F'                   PD Closing Tag 0 (Stream Access)
```

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                   PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'00'                   Invoke ID=0
X'06'                   Service ACK Choice=6 (AtomicReadFile-ACK)

X'10'                   Application Tag 1 (Boolean, 0 (FALSE)) (End Of File)
X'0E'                   PD Opening Tag 0 (Stream Access)
        X'31'                   Application Tag 3 (Signed Integer, L=1) (File Start Position)
        X'00'                   0
        X'65'                   Application Tag 6 (Octet String, L>4)
        X'1B'                   Extended Length=27
        X'4368696C6C657230312044F6E2D54696D653D342E3320486F757273'
                                "Chiller01 On-Time=4.3 Hours"
X'0F'                   PD Closing Tag 0 (Stream Access)
```

Example 2: Read record from a file.

```
X'00'                   PDU type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                   Maximum APDU Size Accepted=206 octets
X'12'                   Invoke ID=18
X'06'                   Service Request=6 (AtomicReadFile-Request)

X'C4'                   Application Tag 12 (Object Identifier, L=4) (FileIdentifier)
X'02800002'             File, Instance Number=2
X'1E'                   PD Opening Tag 1 (Record Access)
        X'31'                   Application Tag 3 (Signed Integer, L=1) (File Start Record)
        X'0E'                   14
        X'21'                   Application Tag 2 (Unsigned, L=1) (Requested Record Count)
        X'03'                   3
X'1F'                   PD Closing Tag 1 (Record Access)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                          PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'12'                          Invoke ID=18
X'06'                          Service ACK Choice=6 (AtomicReadFile-ACK)

X'11'                          Application Tag 1 (Boolean, 1 (TRUE)) (End Of File)
X'1E'                          PD Opening Tag 1(RecordAccess)
      X'31'                      Application Tag 3 (Signed Integer, L=1) (File Start Record)
      X'0E'                      14
      X'21'                      Application Tag 2 (Unsigned, L=1) (Returned Record Count)
      X'02'                      2
      X'65'                      Application Tag 6 (Octet String, L>4) (File Record Data)
      X'0A'                      Extended Length=10
      X'31323A30302C34352E36'    "12:00,45.6"
      X'65'                      Application Tag 6 (Octet String, L>4) (File Record Data)
      X'0A'                      Extended Length=10
      X'31323A31352C34342E38'    "12:15,44.8"
X'1F'                          PD Closing Tag 1 (Record Access)
```

F.2.2 Encoding for Example E.2.2 - AtomicWriteFile Service

Example 1: Write data to a file.

```
X'00'                          PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                          Maximum APDU Size Accepted=206 octets
X'55'                          Invoke ID=85
X'07'                          Service Choice=7 (AtomicWriteFile-Request)

X'C4'                          Application Tag 12 (Object Identifier, L=4) (File Identifier)
X'02800001'                    File, Instance Number=1
X'0E'                          PD Opening Tag 0 (Stream Access)
        X'31'                      Application Tag 3 (Signed Integer, L=1) (File Start Position)
        X'1E'                      30
        X'65'                      Application Tag 6 (Octet String, L>4) (File Data)
        X'1B'                      Extended Length=27
        X'4368696C6C65723031204F6E2D54696D653D342E3320486F757273'
                                   "Chiller01 On-Time=4.3 Hours"
X'0F'                          PD Closing Tag 0 (Stream Access)
```

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                          PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'55'                          Invoke ID=85
X'07'                          Service ACK Choice=7 (AtomicWriteFile-ACK)
X'09'                          SD Context Tag 0 (File Start Position, L=1)
X'1E'                          30
```

Example 2: Append two records to a file.

```
X'00'                          PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                          Maximum APDU Size Accepted=206 octets
X'55'                          Invoke ID=85
X'07'                          Service Choice=7 (AtomicWriteFile-Request)

X'C4'                          Application Tag 12 (Object Identifier, L=4) (File Identifier)
X'02800002'                    File, Instance Number=2
X'1E'                          PD Opening Tag 1 (Record Access)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        X'31'                   Application Tag 3 (Signed Integer, L=1) (File Start Record)
        X'FF'                   -1 (Append to End of File)
        X'21'                   Application Tag 2 (Unsigned Integer, L=1) (Record Count)
        X'02'                   2
        X'65'                   Application Tag 6 (Octet String, L>4) (File Record Data)
        X'0A'                   Extended Length=10
        X'31323A30302C34352E36' "12:00,45.6"
        X'65'                   Application Tag 6 (Octet String, L>4) (File Record Data)
        X'0A'                   Extended Length=10
        X'31323A31352C34342E38' "12:15,44.8"
X'1F'                   PD Closing Tag 1 (Record Access)
```

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                   PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'55'                   Invoke ID=85
X'07'                   Service ACK Choice=7 (AtomicWriteFile-ACK)
X'19'                   SD Context Tag 1 (File Start Record, L=1)
X'0E'                   14
```

F.3 Example Encodings for Object Access Services

F.3.1 Encoding for Example E.3.1 - AddListElement Service

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                   Maximum APDU Size Accepted=206 octets
X'01'                   Invoke ID=1
X'08'                   Service Choice=8 (AddListElement-Request)

X'0C'                   SD Context Tag 0 (Object Identifier, L=4)
X'02C00003'             Group, Instance Number=3
X'19'                   SD Context Tag 1 (Property Identifier, L=1)
X'35'                   53 (LIST_OF_GROUP_MEMBERS)
X'3E'                   PD Opening Tag 3 (List Of Elements)
        X'0C'                   SD Context Tag 0 (Object Identifier, L=4)
        X'0000000F'             Analog Input, Instance Number=15
        X'1E'                   PD Opening Tag 1 (List Of Property References)
                X'09'                   SD Context Tag 0 (Property Identifier, L=1)
                X'55'                   85 (PRESENT_VALUE)
                X'09'                   SD Context Tag 0 (Property Identifier, L=1)
                X'67'                   103 (RELIABILITY)
        X'1F'                   PD Closing Tag 1 (List Of Property References)
X'3F'                   PD Closing Tag 3 (List Of Elements)
```

Assuming this service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                   PDU Type=2 (BACnet-SimpleACK-PDU)
X'01'                   Invoke ID=1
X'08'                   Service ACK Choice=8 (AddListElement)
```

F.3.2 Encoding for Example E.3.2 - RemoveListElement Service

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                   Maximum APDU Size Accepted=206 octets
X'34'                   Invoke ID=52
X'09'                   Service Choice=9 (RemoveListElement-Request)

X'0C'                   SD Context Tag 0 (Object Identifier, L=4)
X'02C00003'             Group, Instance Number=3
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'19'                                   SD Context Tag 1 (Property Identifier, L=1)
X'35'                                   53 (LIST_OF_GROUP_MEMBERS)
X'3E'                                   PD Opening Tag 3 (List Of Elements)
        X'0C'                                   SD Context Tag 0 (Object Identifier, L=4)
        X'0000000C'                             Analog Input, Instance Number=12
        X'1E'                                   PD Opening Tag 1 (List Of Property References)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'55'                                   85 (PRESENT_VALUE)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'67'                                   103 (RELIABILITY)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'1C'                                   28 (DESCRIPTION)
        X'1F'                                   PD Closing Tag 1 (List Of Property References)

        X'0C'                                   SD Context Tag 0 (Object Identifier, L=4)
        X'0000000D'                             Analog Input, Instance Number=13
        X'1E'                                   PD Opening Tag 1 (List Of Property References)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'55'                                   85 (PRESENT_VALUE)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'67'                                   103 (RELIABILITY)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'1C'                                   28 (DESCRIPTION)
        X'1F'                                   PD Closing Tag 1 (List Of Property References)
X'3F'                                   PD Closing Tag 3 (List Of Elements)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                                   PDU Type=2 (BACnet-SimpleACK-PDU)
X'34'                                   Invoke ID=52
X'09'                                   Service ACK Choice=9 (RemoveListElement)
```

This second part of the example re-inserts two of the three elements removed above:

```
X'00'                                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'02'                                   Maximum APDU Size Accepted=206 octets
X'35'                                   Invoke ID=53
X'08'                                   Service Choice=8 (AddListElement-Request)

X'0C'                                   SD Context Tag 0 (Object Identifier, L=4)
X'02C00003'                             Group, Instance Number=3
X'19'                                   SD Context Tag 1 (Property Identifier, L=1)
X'35'                                   53 (LIST_OF_GROUP_MEMBERS)
X'3E'                                   PD Opening Tag 3 (List Of Elements)
        X'0C'                                   SD Context Tag 0 (Object Identifier, L=4)
        X'0000000C'                             Analog Input, Instance Number=12
        X'1E'                                   PD Opening Tag 1 (List Of Property References)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'55'                                   85 (PRESENT_VALUE)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'67'                                   103 (RELIABILITY)
        X'1F'                                   PD Closing Tag 1 (List Of Property References)

        X'0C'                                   SD Context Tag 0 (Object Identifier, L=4)
        X'0000000D'                             Analog Input, Instance Number=13
        X'1E'                                   PD Opening Tag 1 (List Of Property References)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
                X'55'                                   85 (PRESENT_VALUE)
                X'09'                                   SD Context Tag 0 (Property Identifier, L=1)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
                X'67'                   103 (RELIABILITY)
        X'1F'                   PD Closing Tag 1 (List Of Property References)
X'3F'                   PD Closing Tag 3 (List Of Elements)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                   PDU Type=2 (BACnet-SimpleACK-PDU)
X'35'                   Invoke ID=53
X'08'                   Service ACK Choice=8 (AddListElement)
```

F.3.3 Encoding for Example E.3.3 - CreateObject Service

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'                   Maximum APDU Size Accepted=1024 octets
X'56'                   Invoke ID=86
X'0A'                   Service Choice=10 (CreateObject-Request)

X'0E'                   PD Opening Tag 0 (Object Specifier)
        X'09'                   SD Context Tag 0 (Object Type, L=1)
        X'0A'                   10 (File Object)
X'0F'                   PD Closing Tag 0 (Object Specifier)
X'1E'                   PD Opening Tag 1 (List Of Initial Values)
        X'09'                   SD Context Tag 0 (Property Identifier, L=1)
        X'4D'                   77 (OBJECT_NAME)
        X'2E'                   PD Opening Tag 2 (Value)
                X'75'                   Application Tag 7 (Character String, L>4)
                X'08'                   Extended Length=8
                X'00'                   ISO 10646 (UTF-8) Encoding
                X'5472656E642031'   "Trend 1"
        X'2F'                   PD Closing Tag 2 (Value)

        X'09'                   SD Context Tag 0 (Property Identifier, L=1)
        X'29'                   41 (FILE_ACCESS_METHOD)
        X'2E'                   PD Opening Tag 2 (Value)
                X'91'                   Application Tag 9 (Enumerated, L=1)
                X'00'                   0 (RECORD_ACCESS)
        X'2F'                   PD Closing Tag 2 (Value)
X'1F'                   PD Closing Tag 1 (List Of Initial Values)
```

Assuming the service procedure executes correctly, an acknowledgment is returned conveying the new object identifier:

```
X'30'                   PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'56'                   Invoke ID=86
X'0A'                   Service ACK Choice=10 (CreateObject-ACK)
X'C4'                   Application Tag 12 (Object Identifier, L=4)
X'0280000D'             File, Instance Number=13
```

F.3.4 Encoding for Example E.3.4 - DeleteObject Service

Example 1: A successful attempt to delete an object.

```
X'00'                   PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'                   Maximum APDU Size Accepted=1024 octets
X'57'                   Invoke ID=87
X'0B'                   Service Choice=11 (DeleteObject-Request)
X'C4'                   Application Tag 12 (Object Identifier, L=4)
X'02C00006'             Group, Instance Number=6
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'57'	Invoke ID=87
X'0B'	Service ACK Choice=11 (DeleteObject)

Example 2: An unsuccessful attempt to delete an object.

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'	Maximum APDU Size Accepted=1024 octets
X'58'	Invoke ID=88
X'0B'	Service Choice=11 (DeleteObject-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4)
X'02C00007'	Group, Instance Number=7

In this example, the object is assumed to be protected and cannot be deleted by this protocol service. The server issues the following response:

X'50'	PDU Type=5 (BACnet-Error-PDU)
X'58'	Original Invoke ID=88
X'0B'	Error Choice=11 (DeleteObject)
X'91'	Application Tag 9 (Enumerated, L=1) (Error Class)
X'01'	1 (OBJECT)
X'91'	Application Tag 9 (Enumerated, L=1) (Error Code)
X'17'	23 (OBJECT_DELETION_NOT_PERMITTED)

F.3.5 Encoding for Example E.3.5 - ReadProperty Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'00'	Maximum APDU Size Accepted=50 octets
X'01'	Invoke ID=1
X'0C'	Service Choice=12 (ReadProperty-Request)
X'0C'	SD Context Tag 0 (Object Identifier, L=4)
X'00000005'	Analog Input, Instance Number=5
X'19'	SD Context Tag 1 (Property Identifier, L=1)
X'55'	85 (PRESENT_VALUE)

This request produces the following result:

X'30'		PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01'		Invoke ID=1
X'0C'		Service ACK Choice=12 (ReadProperty-ACK)
X'0C'		SD Context Tag 0 (Object Identifier, L=4)
X'00000005'		Analog Input, Instance Number=5
X'19'		SD Context Tag 1 (Property Identifier, L=1)
X'55'		85 (PRESENT_VALUE)
X'3E'		PD Opening Tag 3 (Property Value)
	X'44'	Application Tag 4 (Real, L=4)
	X'4290999A'	72.3
X'3F'		PD Closing Tag 3 (Property Value)

F.3.6 Deleted Clause

This clause has been removed.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

F.3.7 Encoding for Example E.3.7 - ReadPropertyMultiple Service

Example 1: Reading multiple properties of a single object.

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'                       Maximum APDU Size Accepted=1024 octets
X'F1'                       Invoke ID=241
X'0E'                       Service Choice=14 (ReadPropertyMultiple-Request)

X'0C'                       SD Context Tag 0 (Object Identifier, L=4)
X'00000010'                 Analog Input, Instance Number=16
X'1E'                       PD Opening Tag 1 (List Of Property References)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)

        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'67'                       103 (RELIABILITY)
X'1F'                       PD Closing Tag 1 (List Of Property References)
```

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                       PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'F1'                       Invoke ID=241
X'0E'                       Service ACK Choice=14 (ReadPropertyMultiple-ACK)

X'0C'                       SD Context Tag 0 (Object Identifier, L=4)
X'00000010'                 Analog Input, Instance Number=16
X'1E'                       PD Opening Tag 1 (List Of Results)
        X'29'                       SD Context Tag 2 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)
        X'4E'                       PD Opening Tag 4 (Property Value)
                X'44'                       Application Tag 4 (Real, L=4)
                X'4290999A'                 72.3
        X'4F'                       PD Closing Tag 4 (Property Value)

        X'29'                       SD Context Tag 2 (Property Identifier, L=1)
        X'67'                       103 (RELIABILITY)
        X'4E'                       PD Opening Tag 4 (Property Value)
                X'91'                       Application Tag 9 (Enumerated, L=1)
                X'00'                       0 (NO_FAULT_DETECTED)
        X'4F'                       PD Closing Tag 4 (Property Value)
X'1F'                       PD Closing Tag 1 (List Of Results)
```

Example 2: Reading properties of several objects.

```
X'00'                       PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'                       Maximum APDU Size Accepted=1024 octets
X'02'                       Invoke ID=2
X'0E'                       Service Choice=14 (ReadPropertyMultiple-Request)

X'0C'                       SD Context Tag 0 (Object Identifier, L=4)
X'00000021'                 Analog Input, Instance Number=33
X'1E'                       PD Opening Tag 1 (List Of Property References)
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)
X'1F'                       PD Closing Tag 1 (List Of Property References)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
X'0C'                           SD Context Tag 0 (Object Identifier, L=4)
X'00000032'                     Analog Input, Instance Number=50
X'1E'                           PD Opening Tag 1 (List Of Property References)
        X'09'                           SD Context Tag 0 (Property Identifier, L=1)
        X'55'                           85 (PRESENT_VALUE)
X'1F'                           PD Closing Tag 1 (List Of Property References)

X'0C'                           SD Context Tag 0 (Object Identifier, L=4)
X'00000023'                     Analog Input, Instance Number=35
X'1E'                           PD Opening Tag 1 (List Of Property References)
        X'09'                           SD Context Tag 0 (Property Identifier, L=1)
        X'55'                           85 (PRESENT_VALUE)
X'1F'                           PD Closing Tag 1 (List Of Property References)
```

Assuming this service procedure executes correctly, a complex acknowledgment is returned:

```
X'30'                           PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'02'                           Invoke ID=2
X'0E'                           Service ACK Choice=14 (ReadPropertyMultiple-ACK)

X'0C'                           SD Context Tag 0 (Object Identifier, L=4)
X'00000021'                     Analog Input, Instance Number=33
X'1E'                           PD Opening Tag 1 (List Of Results)
        X'29'                           SD Context Tag 2 (Property Identifier, L=1)
        X'55'                           85 (PRESENT_VALUE)
        X'4E'                           PD Opening Tag 4 (Property Value)
                X'44'                           Application Tag 4 (Real, L=4)
                X'42293333'                     42.3
        X'4F'                           PD Closing Tag 4 (Property Value)
X'1F'                           PD Closing Tag 1 (List Of Results)

X'0C'                           SD Context Tag 0 (Object Identifier, L=4)
X'00000032'                     Analog Input, Instance Number=50
X'1E'                           PD Opening Tag 1 (List Of Results)
        X'29'                           SD Context Tag 2 (Property Identifier, L=1)
        X'55'                           85 (PRESENT_VALUE)
        X'5E'                           PD Opening Tag 5 (Property Access Error)
                X'91'                           Application Tag 9 (Enumerated, L=1) (Error Class)
                X'01'                           1 (OBJECT)
                X'91'                           Application Tag 9 (Enumerated, L=1) (Error Code)
                X'1F'                           31 (UNKNOWN_OBJECT)
        X'5F'                           PD Closing Tag 5 (Property Access Error)
X'1F'                           PD Closing Tag 1 (List Of Results)

X'0C'                           SD Context Tag 0 (Object Identifier, L=4)
X'00000023'                     Analog Input, Instance Number=35
X'1E'                           PD Opening Tag 1 (List Of Results)
        X'29'                           SD Context Tag 2 (Property Identifier, L=1)
        X'55'                           85 (PRESENT_VALUE)
        X'4E'                           PD Opening Tag 4 (Property Value)
                X'44'                           Application Tag 4 (Real, L=4)
                X'43D9D99A'                     435.7
        X'4F'                           PD Closing Tag 4 (Property Value)
X'1F'                           PD Closing Tag 1 (List Of Results)
```

F.3.8 Encoding for Example E.3.8 - ReadRange Service

Example 1: Reading records from a Trend Log object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

```
X'02'                       PDU Type = 0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=1)
X'02'                       Maximum APDU Size Accepted = 206 octets
X'01'                       Invoke ID = 1
X'1A'                       Service Choice = (26), (ReadRange-Request)

X'0C'                       SD Context Tag 0 (Object Identifier, L=4)
X'05000001'                 Trend Log, Instance Number = 1
X'19'                       SD Context Tag 1 (Property Identifier, L=1)
X'83'                       131 (LOG_BUFFER)
X'7E'                       PD Opening Tag 7 (By Time)
        X'A4'                       Application Tag 10 (Date, L=4)
        X'62031701'                 March 23, 1998 (Day Of Week Monday)
        X'B4'                       Application Tag 11, (Time, L=4)
        X'13342200'                 19:52:34.0
        X'31'                       Application Tag 1 (Signed Integer, L=1)
        X'04'                       4 (Count)
X'7F'                       PD Closing Tag 7 (By Time)
```

Assuming the service procedure executes correctly, a complex acknowledgment is returned containing the requested data:

```
X'30'                       PDU Type = 3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'01'                       Invoke ID=1
X'1A'                       Service ACK Choice = (26), (ReadRange-ACK)

X'0C'                       SD Context Tag 0 (Object Identifier, L=4)
X'05000001'                 Trend Log, Instance Number = 1
X'19'                       SD Context Tag 1 (Property Identifier, L=1)
X'83'                       131 (LOG_BUFFER)
X'3A'                       SD Context Tag 3 (Result Flags, L=2)
X'05C0'                     1,1,0 (TRUE, TRUE, FALSE)
X'49'                       SD Context Tag 4 (Item Count, L=1)
X'02'                       2
X'5E'                       PD Opening Tag 5 (Item Data)
        X'0E'                       PD Opening Tag 0 (Timestamp)
                X'A4'                       Application Tag 10 (Date, L=4)
                X'62031701'                 Monday, March 23, 1998
                X'B4'                       Application Tag 11, (Time, L=4)
                X'13361B00'                 19:54:27.0
        X'0F'                       PD Closing Tag 0 (Timestamp)
        X'1E'                       PD Opening Tag 1 (Log Datum)
                X'2C'                       SD Context Tag 2 (REAL, L=4)
                X'41900000'                 18.0
        X'1F'                       PD Closing Tag 1 (Log Datum)
        X'2A'                       SD Context Tag 2 (Status Flags, L=2)
        X'0400'                     0,0,0,0 (FALSE, FALSE, FALSE, FALSE)
        X'0E'                       PD Opening Tag 0 (Timestamp)
                X'A4'                       Application Tag 10 (Date, L=4)
                X'62031701'                 Monday, March 23, 1998
                X'B4'                       Application Tag 11, (Time, L=4)
                X'13381B00'                 19:56:27.0
        X'0F'                       PD Closing Tag 0 (Timestamp)
        X'1E'                       PD Opening Tag 1 (Log Datum)
                X'2C'                       SD Context Tag 2 (REAL, L=4)
                X'4190CCCD'                 18.1
        X'1F'                       PD Closing Tag 1 (Log Datum)
        X'2A'                       SD Context Tag 2 (Status Flags, L=2)
        X'0400'                     0,0,0,0 (FALSE, FALSE, FALSE, FALSE)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'5F' PD Closing Tag 5 (Item Data)
X'6B' SD Context Tag 6 (First Sequence Number, L=3)
X'013561' 79201

F.3.9 Encoding for Example E.3.9 - WriteProperty Service

X'00' PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04' Maximum APDU Size Accepted=1024 octets
X'59' Invoke ID=89
X'0F' Service Choice=15 (WriteProperty-Request)

X'0C' SD Context Tag 0 (Object Identifier, L=4)
X'00800001' Analog Value, Instance Number=1
X'19' SD Context Tag 1 (Property Identifier, L=1)
X'55' 85 (PRESENT_VALUE)
X'3E' PD Opening Tag 3 (Property Value)
X'44' Application Tag 4 (Real, L=4)
X'43340000' Property Value=180.0
X'3F' PD Closing Tag 3 (Property Value)

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20' PDU Type=2 (BACnet-SimpleACK-PDU)
X'59' Invoke ID=89
X'0F' Service ACK Choice=15 (WriteProperty)

F.3.10 Encoding for Example E.3.10 - WritePropertyMultiple Service

X'00' PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04' Maximum APDU Size Accepted=1024 octets
X'01' Invoke ID=1
X'10' Service Choice=16 (WritePropertyMultiple-Request)

X'0C' SD Context Tag 0 (Object Identifier, L=4)
X'00800005' Analog Value, Instance Number=5
X'1E' PD Opening Tag 1 (List Of Properties)
X'09' SD Context Tag 0 (Property Identifier, L=1)
X'55' 85 (PRESENT_VALUE)
X'2E' PD Opening Tag 2 (Property Value)
X'44' Application Tag 4 (Real, L=4)
X'42860000' 67.0
X'2F' PD Closing Tag 2 (Property Value)
X'1F' PD Closing Tag 1 (List Of Properties)

X'0C' SD Context Tag 0 (Object Identifier, L=4)
X'00800006' Analog Value, Instance Number=6
X'1E' PD Opening Tag 1 (List Of Properties)
X'09' SD Context Tag 0 (Property Identifier, L=1)
X'55' 85 (PRESENT_VALUE)
X'2E' PD Opening Tag 2 (Property Value)
X'44' Application Tag 4 (Real, L=4)
X'42860000' 67.0
X'2F' PD Closing Tag 2 (Property Value)
X'1F' PD Closing Tag 1 (List Of Properties)

X'0C' SD Context Tag 0 (Object Identifier, L=4)
X'00800007' Analog Value, Instance Number=7
X'1E' PD Opening Tag 1 (List Of Properties)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
        X'09'                       SD Context Tag 0 (Property Identifier, L=1)
        X'55'                       85 (PRESENT_VALUE)
        X'2E'                       PD Opening Tag 2 (Property Value)
              X'44'                       Application Tag 4 (Real, L=4)
              X'42900000'                 72.0
        X'2F'                       PD Closing Tag 2 (Property Value)
X'1F'                       PD Closing Tag 1 (List Of Properties)
```

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

```
X'20'                       PDU Type=2 (BACnet-SimpleACK-PDU)
X'01'                       Invoke ID=1
X'10'                       Service ACK Choice=16 (WritePropertyMultiple)
```

F.3.11 Encoding for Examples E.3.11 - WriteGroup Service

Example #1: Setting control group 23 channel 268=1111, channel 269=2222, priority for writing is 8.

```
X'10'                       PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'0A'                       Service Choice=10 (WriteGroup-Request)

X'09'                       SD Context Tag 0 (Group Number, L=1)
X'17'                       23
X'19'                       SD Context Tag 1 (Write Priority, L=1)
X'08'                       8
X'2E'                       PD Opening Tag 2 (Change List)
        X'0A'                       SD Context Tag 0 (Channel, L=2)
        X'010C'                     268
        X'22'                       Application Tag 2 (Unsigned, L=2) (value)
        X'0457'                     1111
        X'0A'                       SD Context Tag 0 (Channel, L=2)
        X'010D'                     269
        X'22'                       Application Tag 2 (Unsigned, L=2) (value)
        X'08AE'                     2222
X'2F'                       PD Closing Tag 2
```

Note that no response is required for this message since it is of type unconfirmed.

Example #2: Setting control group 23 channel 12=67.0, channel 13=72.0, priority for writing is 8, inhibit execution delays.

```
X'10'                       PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'0A'                       Service Choice=10 (WriteGroup-Request)

X'09'                       SD Context Tag 0 (Group Number, L=1)
X'17'                       23
X'19'                       SD Context Tag 1 (Write Priority, L=1)
X'08'                       8
X'2E'                       PD Opening Tag 2 (Change List)
  X'09'                       SD Context Tag 0 (Channel, L=1)
  X'0C'                       12
  X'44'                       Application Tag 4 (Real, L=4) (value)
  X'42860000'                 67.0
  X'09'                       SD Context Tag 0 (Channel, L=1)
  X'0D'                       13
  X'44'                       Application Tag 4 (Real, L=4) (value)
  X'42900000'                 72.0
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'2F'	PD Closing Tag 2
X'39'	SD Context Tag 3 (Inhibit Delay, L=1)
X'01'	1 (TRUE)

Note that no response is required for this message since it is of type unconfirmed.

Example #3: Setting control group 23 channel 12=1111 at priority 8, channel 13="ABC" at priority 10.

X'10'	PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'0A'	Service Choice=10 (WriteGroup-Request)
X'09'	SD Context Tag 0 (Group Number, L=1)
X'17'	23
X'19'	SD Context Tag 1 (Write Priority, L=1)
X'08'	8
X'2E'	PD Opening Tag 2 (Change List)
X'09'	SD Context Tag 0 (Channel, L=1)
X'0C'	12
X'22'	Application Tag Unsigned L=2 (value)
X'0457'	1111
X'09'	SD Context Tag 0 (Channel, L=1)
X'0D'	13
X'19'	SD Context Tag 1 (overridingPriority, L=1)
X'0A'	10
X'74'	Application Tag Character String L=4 (value)
X'00'	0 (Charset UTF-8)
X'414243'	"ABC"
X'2F'	PD Closing Tag 2

Note that no response is required for this message since it is of type unconfirmed.

F.3.12 Encoding for Example E.3.12 - ReadPropertyMultiple OPTIONAL response if there are no optional properties

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'	Maximum APDU Size Accepted=1024 octets
X'02'	Invoke ID=2
X'0E'	Service Choice=14 (ReadPropertyMultiple-Request)
X'0C'	SD Context Tag 0 (Object Identifier, L=4)
X'00000013'	Analog Input, Instance Number=19
X'1E'	PD Opening Tag 1 (List Of Property References)
X'09'	SD Context Tag 0 (Property Identifier, L=1)
X'55'	80 (OPTIONAL)
X'1F'	PD Closing Tag 1 (List Of Property References)

Assuming that (Analog Input, 19) exists and contains no optional properties, the result would be:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'02'	Invoke ID=2
X'0E'	Service ACK Choice=14 (ReadPropertyMultiple-ACK)
X'0C'	SD Context Tag 0 (Object Identifier, L=4)
X'00000013'	Analog Input, Instance Number=19
X'1E'	PD Opening Tag 1 (List of Results)
X'1F'	PD Closing Tag 1 (List of Results)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

F.4 Example Encodings for Remote Device Management Services

F.4.1 Encoding for Example E.4.1 - DeviceCommunicationControl Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'04'	Maximum APDU Size Accepted=1024 octets
X'05'	Invoke ID=5
X'11'	Service Choice=17 (DeviceCommunicationControl-Request)
X'09'	SD Context Tag 0 (Time Duration, L=1)
X'05'	5
X'19'	SD Context Tag 1 (Enable-Disable, L=1)
X'02'	2 (DISABLE_INITIATION)
X'2D'	SD Context Tag 2 (Password, L>4)
X'08'	Extended Length=8
X'00'	ISO 10646 (UTF-8) Encoding
X'23656762646621'	"#egbdf!"

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'05'	Invoke ID=5
X'11'	Service ACK Choice=17 (DeviceCommunicationControl)

F.4.2 Encoding for Example E.4.2 - ConfirmedPrivateTransfer Service

X'00'		PDU Type=0 (BACnet Confirmed-Request-PDU) SEG=0, MOR=0, SA=0)
X'04'		Maximum APDU Size Accepted=1024 octets
X'55'		Invoke ID=85
X'12'		Service Choice=18 (ConfirmedPrivateTransfer)
X'09'		SD Context Tag 0 (Vendor ID, L=1)
X'19'		25 (XYZ Controls Company Limited)
X'19'		SD Context Tag 1 (Service Number, L=1)
X'08'		8 (XYZ Proprietary Service #8)
X'2E'		PD Opening Tag 2 (Service Parameters)
	X'44'	Application Tag 4 (Real, L=4)
	X'4290CCCD'	72.4
	X'62'	Application Tag 6 (Octet String, L=2)
	X'1649'	X'1649'
X'2F'		PD Closing Tag 2 (Service Parameters)

Assuming that the service is successfully executed but no results need to be returned to the requesting BACnet-user, a 'Result(+)' primitive will be returned using a BACnet-ComplexACK-PDU:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'55'	Invoke ID=85
X'12'	Service Choice=18 (ConfirmedPrivateTransfer-ACK)
X'09'	SD Context Tag 0 (Vendor ID, L=1)
X'19'	25
X'19'	SD Context Tag 1 (Service Number, L=1)
X'08'	8

F.4.3 Encoding for Example E.4.3 - UnconfirmedPrivateTransfer Service

X'10'	PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'04'	Service Choice=4 (UnconfirmedPrivateTransfer-Request)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'09'		SD Context Tag 0 (Vendor ID, L=1)
X'19'		25
X'19'		SD Context Tag 1 (Service Number, L=1)
X'08'		8
X'2E'		PD Opening Tag 2 (Service Parameters)
	X'44'	Application Tag 4 (Real, L=4)
	X'4290CCCD'	72.4
	X'62'	Application Tag 6 (Octet String, L=2)
	X'1649'	X'1649'
X'2F'		PD Closing Tag 2 (Service Parameters)

F.4.4 Encoding for Example E.4.4 - ReinitializeDevice Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'02'	Invoke ID=2
X'14'	Service Choice=20 (ReinitializeDevice-Request)
X'09'	SD Context Tag 0 (Reinitialized State Of Device, L=1)
X'01'	1 (WARMSTART)
X'1D'	SD Context Tag 1 (Password, L>4)
X'09'	Extended Length=9
X'00'	ISO 10646 (UTF-8) Encoding
X'4162436445664768'	"AbCdEfGh"

Assuming this service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'02'	Invoke ID=2
X'14'	Service ACK Choice=20 (ReinitializeDevice)

F.4.5 Encoding for Example E.4.5 - ConfirmedTextMessage Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'03'	Invoke ID=3
X'13'	Service Choice=19 (ConfirmedTextMessage-Request)
X'0C'	SD Context Tag 0 (Text Message Source Device, L=4)
X'02000005'	Device, Instance Number=5
X'29'	SD Context Tag 2 (Message Priority, L=1)
X'00'	0 (NORMAL)
X'3D'	SD Context Tag 3 (Message, L>4)
X'18'	Extended Length=24
X'00'	ISO 10646 (UTF-8) Encoding
X'504D207265717569726564206 66F722050554D50333437'	"PM required for PUMP347"

Assuming the service procedure executes correctly, a simple acknowledgment is returned:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'03'	Invoke ID=3
X'13'	Service ACK Choice=19 (ConfirmedTextMessage)

F.4.6 Encoding for Example E.4.6 - UnconfirmedTextMessage Service

X'10'	PDU Type=1 (BACnet-Unconfirmed-Request-PDU)
X'05'	Service Choice=5 (UnconfirmedTextMessage-Request)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'0C' SD Context Tag 0 (Text Message Source Device, L=4)
X'02000005' Device, Instance Number=5
X'29' SD Context Tag 2 (Message Priority, L=1)
X'00' 0 (NORMAL)
X'3D' SD Context Tag 3 (Message, L>4)
X'18' Extended Length=24
X'00' ISO 10646 (UTF-8) Encoding
X'504D20726571756972656420666F722050554D50333437' "PM required for PUMP347"

Note that no response is required for this message since it is of type unconfirmed.

F.4.7 Encoding for Example E.4.7 - TimeSynchronization Service

X'10' PDU Type=1 (BACnet-Unconfirmed-Service-Request-PDU)
X'06' Service Choice=6 (TimeSynchronization-Request)

X'A4' Application Tag 10 (Date, L=4)
X'5C0B1102' November 17, 1992 (Day of Week = Tuesday)
X'B4' Application Tag 11 (Time, L=4)
X'162D1E46' 22:45:30.70

F.4.8 Encoding for Example E.4.8 - Who-Has and I-Have Services

Example 1: Locating the device that contains an object for which the Object_Name is known.

X'10' PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'07' Service Choice=7 (Who-Has-Request)

X'3D' SD Context Tag 3 (Object Name, L>4)
X'07' Extended Length=7
X'00' ISO 10646 (UTF-8) Encoding
X'4F4154656D70' "OATemp"

Assuming that exactly one other device has such an object, the following I-Have service indication would be received.

X'10' PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'01' Service Choice=1 (I-Have-Request)

X'C4' Application Tag 12 (Object Identifier, L=4) (Device Identifier)
X'02000008' Device, Instance Number=8
X'C4' Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000003' Analog Input, Instance Number=3
X'75' Application Tag 7 (Character String, L>4) (Object Name)
X'07' Extended Length=7
X'00' ISO 10646 (UTF-8) Encoding
X'4F4154656D70' "OATemp"

Example 2: Locating the device that contains an object for which the Object_Identifier is known.

X'10' PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'07' Service Choice=7 (Who-Has-Request)

X'2C' SD Context Tag 2 (Object Identifier, L=4)
X'00000003' Analog Input, Instance Number=3

Assuming that exactly one other device has such an object, the following I-Have service indication would be received.

X'10' PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'01' Service Choice=1 (I-Have-Request)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'C4'	Application Tag 12 (Object Identifier, L=4) (Device Identifier)
X'02000008'	Device, Instance Number=8
X'C4'	Application Tag 12 (Object Identifier, L=4) (Object Identifier)
X'00000003'	Analog Input, Instance Number=3
X'75'	Application Tag 7 (Character String, L>4) (Object Name)
X'07'	Extended Length=7
X'00'	ISO 10646 (UTF-8) Encoding
X'4F4154656D70'	"OATemp"

F.4.9 Encoding for Example E.4.9 - Who-Is and I-Am Services

Example 1: Establishing the network address of a device with a known Device object identifier, i.e. instance number.

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'08'	Service Choice=8 (Who-Is-Request)
X'09'	SD Context Tag 0 (Device Instance Range Low Limit, L=1)
X'03'	3
X'19'	SD Context Tag 1 (Device Instance Range High Limit, L=1)
X'03'	3

Assuming that there is such a device on the network, it responds some time later using the I-Am service:

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000003'	Device, Instance Number=3
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Max APDU Length Accepted)
X'0400'	1024
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'03'	3 (NO_SEGMENTATION)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Vendor ID)
X'63'	99

Example 2: Finding out about all network devices.

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'08'	Service Choice=8 (Who-Is-Request)

Each device on the network responds using the I-Am service:

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000001'	Device, Instance Number=1
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Max APDU Length Accepted)
X'01E0'	480
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'01'	1 (SEGMENTED_TRANSMIT)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Vendor ID)
X'63'	99
X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000002'	Device, Instance Number=2
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Max APDU Length Accepted)
X'CE'	206
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'02'	2 (SEGMENTED_RECEIVE)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Vendor ID)
X'21'	33
X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000003'	Device, Instance Number=3
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Max APDU Length Accepted)
X'0400'	1024
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'03'	3 (NO_SEGMENTATION)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Vendor ID)
X'63'	99
X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000004'	Device, Instance Number=4
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Max APDU Length Accepted)
X'80'	128
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'00'	0 (SEGMENTED_BOTH)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Vendor ID)
X'42'	66

F.4.10 Encoding for Example E.4.10 - Who-Am-I and You-Are Services

Example 1: Assigning the network address and Device Identifier of a device.

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'0E'	Service Choice=14 (You-Are-Request)
X'22'	Application Tag 2 (Unsigned Integer, L=2)(Vendor ID)
X'022B'	555
X'75'	Application Tag 7 (Character String, L>4)
X'06'	Extended Length=6
X'00'	ISO 10646 (UTF-8) Encoding
X'4C4D43503234'	"LMCP24"
X'75'	Application Tag 7 (Character String, L>4)
X'1C'	Extended Length=5
X'00'	ISO 10646 (UTF-8) Encoding
X'3132333435'	"12345"

Assuming that there is such a device on the network, it responds sometime later using the I-Am service:

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000003'	Device, Instance Number=3

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'22'	Application Tag 2 (Unsigned Integer, L=2) (Max APDU Length Accepted)
X'01E0'	480
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'03'	3 (NO_SEGMENTATION)
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Vendor ID)
X'022B'	555

Example 2: A device (Vendor ID=555, Model Name "LMCP24", Serial Number="12345") needs its Device Identifier and MAC address configured, and only its Vendor Identifier, Model Name, and Serial Number are known.

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'0D'	Service Choice=13 (Who-Am-I-Request)
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Vendor ID)
X'022B'	555
X'75'	Application Tag 7 (Character String, L>4)
X'06'	Extended Length=6
X'00'	ISO 10646 (UTF-8) Encoding
X'4C4D43503234'	"LMCP24"
X'75'	Application Tag 7 (Character String, L>4)
X'1C'	Extended Length=5
X'00'	ISO 10646 (UTF-8) Encoding
X'3132333435'	"12345"

Some other device has a list, for a number of devices, of a Device Identifier, MAC address, Vendor Identifier, Model Name, and a Serial Number for each device in the list. This device has the responsibility for handling the assignment of Device Identifier and MAC Address of the requesting device (Vendor ID=555, Model Name "LMCP24", Serial Number="12345"), resulting in a You-Are service request:

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'0E'	Service Choice=14 (You-Are-Request)
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Vendor ID)
X'022B'	555
X'75'	Application Tag 7 (Character String, L>4)
X'06'	Extended Length=6
X'00'	ISO 10646 (UTF-8) Encoding
X'4C4D43503234'	"LMCP24"
X'75'	Application Tag 7 (Character String, L>4)
X'1C'	Extended Length=5
X'00'	ISO 10646 (UTF-8) Encoding
X'3132333435'	"12345"
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000003'	Device, Instance Number=3
X'61'	Application Tag 6 (Octet String, L=1)
X'2A'	

Assuming that a device on the network matches the criteria in the You-Are service request, it responds sometime later using the I-Am service:

X'10'	PDU Type=1 (Unconfirmed-Service-Request-PDU)
X'00'	Service Choice=0 (I-Am-Request)
X'C4'	Application Tag 12 (Object Identifier, L=4) (I-Am Device Identifier)
X'02000003'	Device, Instance Number=3
X'22'	Application Tag 2 (Unsigned Integer, L=2) (Max APDU Length Accepted)
X'01E0'	480
X'91'	Application Tag 9 (Enumerated, L=1) (Segmentation Supported)
X'03'	3 (NO_SEGMENTATION)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'22'	Application Tag 2 (Unsigned Integer, L=2) (Vendor ID)
X'022B'	555

F.5 Example Encodings for Virtual Terminal Services

Establishing a VT session:

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'50'	Invoke ID=80
X'15'	Service Choice=21 (VT-Open-Request)
X'91'	Application Tag 9 (Enumerated, L=1) (VT Class)
X'01'	1 (ANSI_X3.64)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Local VT Session Identifier)
X'05'	5

Assuming that the target device can create a new VT-session, a complex acknowledgment is returned:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'50'	Invoke ID=80
X'15'	Service Choice=21 (VT-Open-ACK)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (Remote VT Session Identifier)
X'1D'	29

Terminal sign-on. The target device sends a prompt:

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'51'	Invoke ID=81
X'17'	Service Choice=23 (VT-Data-Request)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Session Identifier)
X'05'	5
X'65'	Application Tag 6 (Octet String, L>4) (VT New Data)
X'12'	Extended Length=18
X'0D0A456E746572205573657204E616D653A'	"{cr}{lf}Enter User Name:"
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Data Flag)
X'00'	0

Assuming that the operator interface device receives the data correctly, a complex acknowledgment is returned:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'51'	Invoke ID=81
X'17'	Service Choice=23 (VT-Data-ACK)
X'09'	SD Context Tag 0 (All New Data Accepted, L=1)
X'01'	1 (TRUE)

Entering user name:

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'52'	Invoke ID=82
X'17'	Service Choice=23 (VT-Data-Request)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Session Identifier)
X'1D'	29
X'65'	Application Tag 6 (Octet String, L>4) (VT New Data)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

X'05'	Extended Length=5
X'465245440D'	"FRED{cr}"
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Data Flag)
X'00'	0

To which the target device would respond:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'52'	Invoke ID=82
X'17'	Service Choice=23 (VT-Data-ACK)
X'09'	SD Context Tag 0 (All New Data Accepted, L=1)
X'01'	1 (TRUE)

Entering password:

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'53'	Invoke ID=83
X'17'	Service Choice=23 (VT-Data-Request)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Session Identifier)
X'05'	5
X'65'	Application Tag 6 (Octet String, L>4) (VT New Data)
X'15'	Extended Length=21
X'465245440D0A456E74657220506173373776F72643A'"FRED{cr}{lf}Enter Password:"	
X'21'	Application Tag 2 (Unsigned Integer, L=1) (VT Data Flag)
X'01'	1

To which the target device would respond:

X'30'	PDU Type=3 (BACnet-ComplexACK-PDU, SEG=0, MOR=0)
X'53'	Invoke ID=83
X'17'	Service Choice=23 (VT-Data-ACK)
X'09'	SD Context Tag 0 (All New Data Accepted, L=1)
X'01'	1 (TRUE)

Terminal sign-off:

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)
X'01'	Maximum APDU Size Accepted=128 octets
X'54'	Invoke ID=84
X'16'	Service Choice=22 (VT-Close-Request)
X'21'	Application Tag 2 (Unsigned Integer, L=1) (List Of Remote VT Session Identifiers)
X'1D'	29

Response:

X'20'	PDU Type=2 (BACnet-SimpleACK-PDU)
X'54'	Invoke ID=84
X'16'	Service ACK Choice=22 (VT-Close)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX G - CALCULATION OF CRC (INFORMATIVE)

(This annex is not part of this standard but is included for informative purposes only.)

Historically, CRC generators have been implemented as shift registers with exclusive OR feedback. This provides an inexpensive way to process CRC information on a serial bit stream in hardware. Since commercial UARTs do not provide hardware calculation of CRC, UART-based protocols such as those described in Clauses 9 and 10 must perform this calculation with software. While this can be done one bit at a time, simulating a shift register with feedback, a much more efficient algorithm is possible that computes the CRC on an entire octet at once. This annex shows how the CRC may be computed in this manner. This algorithm is presented as an example and is not intended to restrict the vendor's implementation of the CRC calculation.

G.1 Calculation of the Header CRC

We begin with the diagram of a hardware CRC generator as shown in Figure G-1. The polynomial used is

$$X^8 + X^7 + 1$$

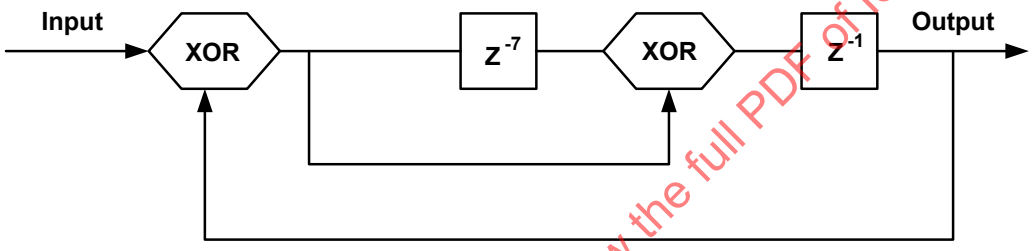


Figure G-1. Hardware header-CRC generator.

The hardware implementation operates on a serial bit stream, whereas our calculation must operate on entire octets. To this end, we follow the operation of the circuit through eight bits of data. The CRC shift register is initialized to X0 (input end) to X7 (output end), the input data is D0 to D7 (least significant bit first, as transmitted and received by a UART). Within each block below, the terms are exclusive OR'ed vertically.

```
input    register contents
   |-------------------------------|
D0| X0   X1   X2   X3   X4   X5   X6   X7  |
   |-------------------------------|
D1|      X0   X1   X2   X3   X4   X5   X6  |
   | D0                             D0  |
   | X7                             X7  |
   |-------------------------------|
D2|           X0   X1   X2   X3   X4   X5  |
   | D1   D0                        D1  |
   | X6   X7                        X6  |
   | D0                             D0  |
   | X7                             X7  |
   |-------------------------------|
D3|                X0   X1   X2   X3   X4  |
   | D2   D1   D0                   D2  |
   | X5   X6   X7                   X5  |
   | D1   D0                        D1  |
   | X6   X7                        X6  |
   | D0                             D0  |
   | X7                             X7  |
   |-------------------------------|
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
   |---------------------------------|
D4|                   X0  X1  X2  X3 |
   | D3  D2  D1  D0                D3 |
   | X4  X5  X6  X7                X4 |
   | D2  D1  D0                    D2 |
   | X5  X6  X7                    X5 |
   | D1  D0                        D1 |
   | X6  X7                        X6 |
   | D0                            D0 |
   | X7                            X7 |
   |---------------------------------|
D5|                       X0  X1  X2 |
   | D4  D3  D2  D1  D0            D4 |
   | X3  X4  X5  X6  X7            X3 |
   | D3  D2  D1  D0                D3 |
   | X4  X5  X6  X7                X4 |
   | D2  D1  D0                    D2 |
   | X5  X6  X7                    X5 |
   | D1  D0                        D1 |
   | X6  X7                        X6 |
   | D0                            D0 |
   | X7                            X7 |
   |---------------------------------|
D6|                           X0  X1 |
   | D5  D4  D3  D2  D1  D0        D5 |
   | X2  X3  X4  X5  X6  X7        X2 |
   | D4  D3  D2  D1  D0            D4 |
   | X3  X4  X5  X6  X7            X3 |
   | D3  D2  D1  D0                D3 |
   | X4  X5  X6  X7                X4 |
   | D2  D1  D0                    D2 |
   | X5  X6  X7                    X5 |
   | D1  D0                        D1 |
   | X6  X7                        X6 |
   | D0                            D0 |
   | X7                            X7 |
   |---------------------------------|
D7|                               X0 |
   | D6  D5  D4  D3  D2  D1  D0    D6 |
   | X1  X2  X3  X4  X5  X6  X7    X1 |
   | D5  D4  D3  D2  D1  D0        D5 |
   | X2  X3  X4  X5  X6  X7        X2 |
   | D4  D3  D2  D1  D0            D4 |
   | X3  X4  X5  X6  X7            X3 |
   | D3  D2  D1  D0                D3 |
   | X4  X5  X6  X7                X4 |
   | D2  D1  D0                    D2 |
   | X5  X6  X7                    X5 |
   | D1  D0                        D1 |
   | X6  X7                        X6 |
   | D0                            D0 |
   | X7                            X7 |
   |---------------------------------|
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

							D0
							X7
D7	D6	D5	D4	D3	D2	D1	D7
X0	X1	X2	X3	X4	X5	X6	X0
D6	D5	D4	D3	D2	D1	D0	D6
X1	X2	X3	X4	X5	X6	X7	X1
D5	D4	D3	D2	D1	D0		D5
X2	X3	X4	X5	X6	X7		X2
D4	D3	D2	D1	D0			D4
X3	X4	X5	X6	X7			X3
D3	D2	D1	D0				D3
X4	X5	X6	X7				X4
D2	D1	D0					D2
X5	X6	X7					X5
D1	D0						D1
X6	X7						X6
D0							D0
X7							X7

The final block above is the net result of shifting an octet of data through the CRC generator. By performing the indicated exclusive OR operations, an octet can be processed into the CRC. In order to simplify the required shifting operations, we define X0 to be the most significant bit of the parallel representation and X7 to be the least significant. Since most microprocessors number bits in the opposite order, we rename the bits to correspond to standard microprocessor bit nomenclature. Let C7 = X0, C6 = X1, etc. The final block in the previous table thus becomes:

X0	X1	X2	X3	X4	X5	X6	X7
C7	C6	C5	C4	C3	C2	C1	C0
							D0
							C0
D7	D6	D5	D4	D3	D2	D1	D7
C7	C6	C5	C4	C3	C2	C1	C7
D6	D5	D4	D3	D2	D1	D0	D6
C6	C5	C4	C3	C2	C1	C0	C6
D5	D4	D3	D2	D1	D0		D5
C5	C4	C3	C2	C1	C0		C5
D4	D3	D2	D1	D0			D4
C4	C3	C2	C1	C0			C4
D3	D2	D1	D0				D3
C3	C2	C1	C0				C3
D2	D1	D0					D2
C2	C1	C0					C2
D1	D0						D1
C1	C0						C1
D0							D0
C0							C0

or, rearranging vertically (since exclusive OR is commutative) to minimize computation, and recognizing that any term exclusive OR'ed with itself may be eliminated:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
|--------------------------------|
| D7  D6  D5  D4  D3  D2  D1     |
| C7  C6  C5  C4  C3  C2  C1     |
| D6  D5  D4  D3  D2  D1  D0 D7  |
| C6  C5  C4  C3  C2  C1  C0 C7  |
| D5  D4  D3  D2  D1  D0     D6  |
| C5  C4  C3  C2  C1  C0     C6  |
| D4  D3  D2  D1  D0         D5  |
| C4  C3  C2  C1  C0         C5  |
| D3  D2  D1  D0             D4  |
| C3  C2  C1  C0             C4  |
| D2  D1  D0                 D3  |
| C2  C1  C0                 C3  |
| D1  D0                     D2  |
| C1  C0                     C2  |
| D0                         D1  |
| C0                         C1  |
|--------------------------------|
```

In operation, the CRC register C7-C0 is initialized to all ones. During transmission, the Frame Type, Destination Address, Source Address, and Length are passed though the calculation before transmission. The ones complement of the final CRC register value is then transmitted. At the receiving end, the Frame Type, Destination Address, Source Address, Length octets, and the received CRC octet are passed though the calculation. If all octets are received correctly, then the final value of the receiver's CRC register will be the constant X'55'.

As an example of usage, consider a Token frame from node X'05' to node X'10'. The frame appears as:

Description	Value	CRC Register After Octet is Processed
preamble 1, not included in CRC	X'55'	
preamble 2, not included in CRC	X'FF'	
		X'FF' (initial value)
frame type = TOKEN	X'00'	X'55'
destination address	X'10'	X'C2'
source address	X'05'	X'BC'
data length MSB = 0	X'00'	X'95'
data length LSB = 0	X'00'	X'73' ones complement is X'8C'
Header CRC	X'8C'	X'55' final result at receiver

Thus, the transmitter would calculate the CRC on the five octets X'00', X'10', X'05', X'00', and X'00' to be X'73'. The ones complement of this is X'8C', which is appended to the frame.

The receiver would calculate the CRC on the six octets X'00', X'10', X'05', X'00', and X'8C' to be X'55', which is the expected result for a correctly received frame.

G.1.1 Sample Implementation of the Header CRC Algorithm in C

As an example, a C language implementation of the header CRC algorithm is presented. Inputs are the octet to be processed and the accumulating CRC value. The function return value is the updated CRC value.

In order to minimize shifting, we make use of the bits shifted left out of the least significant octet. In particular, bit 8 will be exclusive OR'ed with the least significant octet.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Desired result:

```
                              | C7  C6  C5  C4  C3  C2  C1  C0 |
                              | C6  C5  C4  C3  C2  C1  C0  C7 |
                              | C5  C4  C3  C2  C1  C0      C6 |
                              | C4  C3  C2  C1  C0          C5 |
                              | C3  C2  C1  C0              C4 |
                              | C2  C1  C0                  C3 |
                              | C1  C0                      C2 |
                              | C0                          C1 |
```

Shifted 16 bit word:

```
                              | C7  C6  C5  C4  C3  C2  C1      | v
                          C7  | C6  C5  C4  C3  C2  C1  C0      | v<<1
                      C7  C6  | C5  C4  C3  C2  C1  C0          | v<<2
                  C7  C6  C5  | C4  C3  C2  C1  C0              | v<<3
              C7  C6  C5  C4  | C3  C2  C1  C0                  | v<<4
          C7  C6  C5  C4  C3  | C2  C1  C0                      | v<<5
      C7  C6  C5  C4  C3  C2  | C1  C0                          | v<<6
C7  C6  C5  C4  C3  C2  C1    | C0                              | v<<7
```

```
/* Accumulate "dataValue" into the CRC in crcValue.
/  Return value is updated CRC
/
/  Assumes that "unsigned char" is equivalent to one octet.
/  Assumes that "unsigned int" is 16 bits.
/  The ^ operator means exclusive OR.
*/
unsigned char CalcHeaderCRC(unsigned char dataValue, unsigned char crcValue)
{
   unsigned int crc;

   crc = crcValue ^ dataValue;    /* XOR C7..C0 with D7..D0 */

/* Exclusive OR the terms in the table (top down) */
 crc = crc ^ (crc << 1) ^ (crc << 2) ^ (crc << 3)
      ^ (crc << 4) ^ (crc << 5) ^ (crc << 6) ^ (crc << 7);

 /* Combine bits shifted out left hand end */
 return (crc & 0xfe) ^ ((crc >> 8) & 1);
}
```

G.1.2 Sample Implementation of the Header CRC Algorithm in Assembly Language

As an example, an assembly language implementation of the Header CRC algorithm for the 68HC11 (an eight bit processor with a simple instruction set) is presented. The routine accumulates the CRC into the variable CRCLO. T1 is a temporary storage variable. Most instructions act on either accumulator A or B.

```
HEADERCRC:                      ;ACCUMULATE THE OCTET (0,X) INTO THE DATA CRC
        LDAB    0,X               ;FETCH DATA OCTET (INDEXED OPERATION)
        EORB    CRCLO             ;D7-D0 EXCLUSIVE OR C7-C0
        STAB    CRCLO             ;SAVE RESULT
        CLRA                      ;CLEAR REGISTER A
        ASLD                      ;SHIFT (A,B) LEFT AS A 16 BIT VALUE
        ; A=(- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
        EORB    CRCLO
        ASLD
        ; A=(- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
        ;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
EORB    CRCLO
ASLD
; A=(- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
EORB    CRCLO

ASLD
; A=(- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
;   (- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
EORB    CRCLO
ASLD
; A=(- - - 7 6 5 4 3) B=(2 1 0 - - - - -)
;   (- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
;   (- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
EORB    CRCLO
ASLD
; A=(- - 7 6 5 4 3 2) B=(1 0 - - - - - -)
;   (- - - 7 6 5 4 3) B=(2 1 0 - - - - -)
;   (- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
;   (- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
EORB    CRCLO

ASLD
; A=(- 7 6 5 4 3 2 1) B=(0 - - - - - - -)
;   (- - 7 6 5 4 3 2) B=(1 0 - - - - - -)
;   (- - - 7 6 5 4 3) B=(2 1 0 - - - - -)
;   (- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
;   (- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - 7 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
EORB    CRCLO
ANDB    #0FEH               ;CLEAR LSB OF BOTTOM HALF
ANDA    #01H                ;CLEAR ALL BUT LSB OF HIGH HALF
STAA    T1
; A=(- - - - - - - 1) B=(0 - - - - - - -)
;   (- - - - - - - 2) B=(1 0 - - - - - -)
;   (- - - - - - - 3) B=(2 1 0 - - - - -)
;   (- - - - - - - 4) B=(3 2 1 0 - - - -)
;   (- - - - - - - 5) B=(4 3 2 1 0 - - -)
;   (- - - - - - - 6) B=(5 4 3 2 1 0 - -)
;   (- - - - - - - 7) B=(6 5 4 3 2 1 0 -)
;   (- - - - - - - -) B=(7 6 5 4 3 2 1 -)
EORB    T1                  ;COMBINE LSB OF HIGH HALF
STAB    CRCLO
RTS
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

G.1.3 Other Implementations of the Header CRC Algorithm

Other implementations of the Header CRC algorithm are possible. It may be seen that the new CRC value is a function of the prior CRC value exclusive OR'ed with the data value. Thus, a lookup table with 256 elements may be used to quickly determine the new CRC value, using the prior CRC exclusive OR'ed with the data value as an index. The contents of the table may be computed using the implementations shown in Clause G.1.1 or G.1.2.

© ISO 2022 – All rights reserved

G.2 Calculation of the Data CRC

We begin with the diagram of a hardware CRC generator as shown in Figure G-2. The polynomial used is the CRC-CCITT:

$$X^{16} + X^{12} + X^5 + 1$$

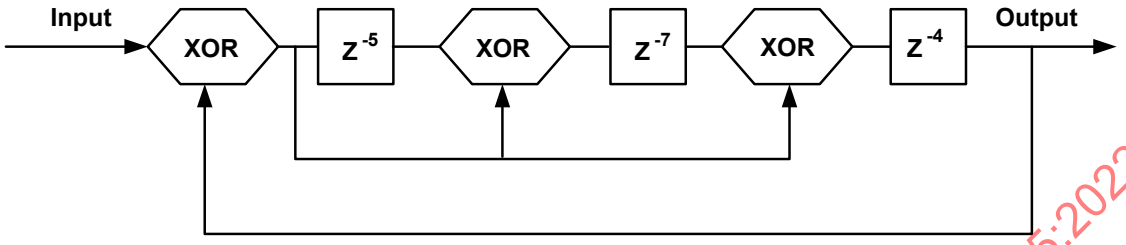


Figure G-2. Hardware data-CRC generator.

The hardware implementation operates on a serial bit stream, whereas our calculation must operate on entire octets. To this end, we follow the operation of the circuit through eight bits of data. The CRC shift register is initialized to X0 (input end) to X15 (output end), the input data is D0 to D7 (least significant bit first, as transmitted and received by a UART). Within each block below, the terms are exclusive OR'ed vertically.

```
input           register contents
  |-------------------------------------------------------------|
D0| X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 X12 X13 X14 X15 |
  |-------------------------------------------------------------|
D1|     X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 X12 X13 X14 |
  | D0                  D0                      D0              |
  | X15                 X15                     X15             |
  |-------------------------------------------------------------|
D2|         X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 X12 X13 |
  | D1  D0              D1  D0                      D1  D0          |
  | X14 X15             X14 X15                     X14 X15         |
  |-------------------------------------------------------------|
D3|             X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 X12 |
  | D2  D1  D0          D2  D1  D0                  D2  D1  D0      |
  | X13 X14 X15         X13 X14 X15                 X13 X14 X15     |
  |-------------------------------------------------------------|
D4|                 X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 X11 |
  | D3  D2  D1  D0      D3  D2  D1  D0              D3  D2  D1  D0  |
  | X12 X13 X14 X15     X12 X13 X14 X15            X12 X13 X14 X15 |
  |-------------------------------------------------------------|
D5|                     X0  X1  X2  X3  X4  X5  X6  X7  X8  X9  X10 |
  |     D3  D2  D1  D0  D3  D2  D1  D0                  D3  D2  D1  |
  |     X12 X13 X14 X15     X12 X13 X14 X15             X12 X13 X14 |
  | D4                  D4                  D4                      |
  | X11                 X11                 X11                     |
  | D0                  D0                  D0                      |
  | X15                 X15                 X15                     |
  |-------------------------------------------------------------|
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

D6						X0	X1	X2	X3	X4	X5	X6	X7	X8	X9
		D3	D2	D1	D0		D3	D2	D1	D0				D3	D2
		X12	X13	X14	X15		X12	X13	X14	X15				X12	X13
	D5	D4			D5	D4						D5	D4		
	X10	X11			X10	X11						X10	X11		
	D1	D0			D1	D0						D1	D0		
	X14	X15			X14	X15						X14	X15		
D7							X0	X1	X2	X3	X4	X5	X6	X7	X8
			D3	D2	D1	D0		D3	D2	D1	D0				D3
			X12	X13	X14	X15		X12	X13	X14	X15				X12
	D6	D5	D4		D6	D5	D4					D6	D5	D4	
	X9	X10	X11		X9	X10	X11					X9	X10	X11	
	D2	D1	D0		D2	D1	D0					D2	D1	D0	
	X13	X14	X15		X13	X14	X15					X13	X14	X15	
							X0	X1	X2	X3	X4	X5	X6	X7	
			D3	D2	D1	D0		D3	D2	D1	D0				
			X12	X13	X14	X15		X12	X13	X14	X15				
	D7	D6	D5	D4	D7	D6	D5	D4				D7	D6	D5	D4
	X8	X9	X10	X11	X8	X9	X10	X11				X8	X9	X10	X11
	D3	D2	D1	D0	D3	D2	D1	D0				D3	D2	D1	D0
	X12	X13	X14	X15	X12	X13	X14	X15				X12	X13	X14	X15

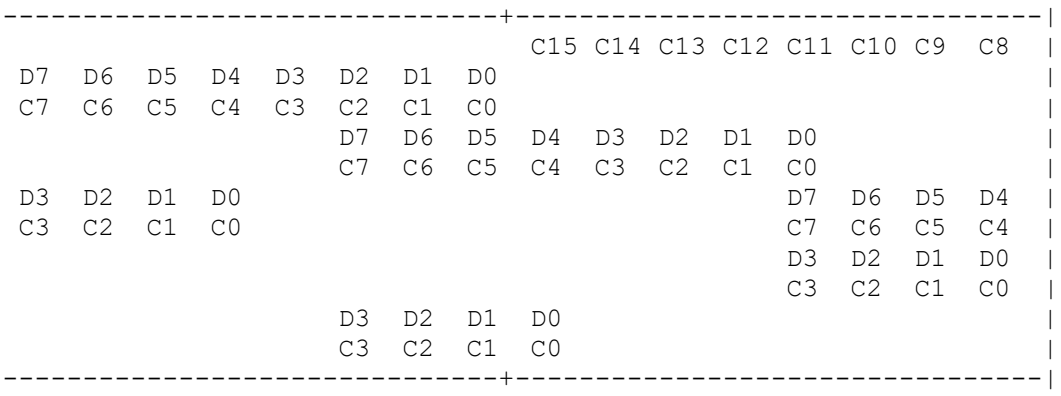
The final block above is the net result of shifting an octet of data through the CRC generator. By performing the indicated exclusive OR operations, an octet can be processed into the CRC. In order to simplify the required shifting operations, we define X0 to be the most significant bit of the parallel representation and X15 to be the least significant. Since most microprocessors number bits in the opposite order, we rename the bits to correspond to standard microprocessor bit nomenclature. Let C15 = X0, C14 = X1, etc. The final block in the previous table thus becomes:

X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15
C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
								C15	C14	C13	C12	C11	C10	C9	C8
				D3	D2	D1	D0		D3	D2	D1	D0			
				C3	C2	C1	C0		C3	C2	C1	C0			
D7	D6	D5	D4		D7	D6	D5	D4				D7	D6	D5	D4
C7	C6	C5	C4		C7	C6	C5	C4				C7	C6	C5	C4
D3	D2	D1	D0		D3	D2	D1	D0				D3	D2	D1	D0
C3	C2	C1	C0		C3	C2	C1	C0				C3	C2	C1	C0

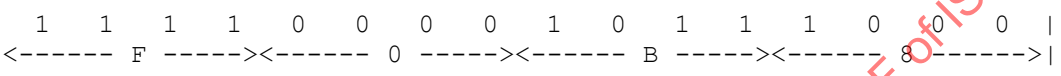
or, rearranging vertically (since exclusive OR is commutative) to minimize computation:

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved



In operation, the CRC register C15-C0 is initialized to all ones. During transmission, the Data are passed though the calculation before transmission. The ones complement of the final CRC register value is then transmitted with the least significant octet (C7-C0) first. At the receiving end, all data octets, including the received CRC octets, are passed though the calculation. If all octets are received correctly, then the final value of the receiver's CRC register will be the constant X'F0B8':



As an example of usage, consider a data sequence X'01', X'22', X'30':

Description	Value	CRC Accumulator After Octet is Processed
		X'FFFF' (initial value)
first data octet	X'01'	X'1E0E'
second data octet	X'22'	X'EB70'
third data octet	X'30'	X'42EF' ones complement is X'BD10'
CRC1 (least significant octet)	X'10'	X'0F3A'
CRC2 (most significant octet)	X'BD'	X'F0B8' final result at receiver

Thus, the transmitter would calculate the CRC on the three octets X'01', X'22', and X'30' to be X'42EF'. The ones complement of this is X'BD10', which is appended to the frame least significant octet first as X'10', X'BD'.

The receiver would calculate the CRC on the five octets X'01', X'22', X'30', X'10', and X'BD' to be X'F0B8', which is the expected result for a correctly received frame.

G.2.1 Sample Implementation of the Data CRC Algorithm in C

As an example, a C language implementation of the Data CRC algorithm is presented. Inputs are the octet to be processed and the accumulating CRC value. The function return value is the updated CRC value.

```
/* Accumulate "dataValue" into the CRC in crcValue.
/  Return value is updated CRC
/
/  Assumes that "unsigned char" is equivalent to one octet.
/  Assumes that "unsigned int" is 16 bits.
/  The ^ operator means exclusive OR.
*/

unsigned int CalcDataCRC(unsigned char dataValue, unsigned int crcValue)
{
 unsigned int crcLow;

 crcLow = (crcValue & 0xff) ^ dataValue;  /* XOR C7..C0 with D7..D0 */
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
 /* Exclusive OR the terms in the table (top down) */
 return (crcValue >>8) ^ (crcLow << 8)  ^ (crcLow <<3)
                                ^ (crcLow <<12)  ^ (crcLow >> 4)
                                ^ (crcLow & 0x0f) ^ ((crcLow & 0x0f) << 7);
}
```

G.2.2 Sample Implementation of the Data CRC Algorithm in Assembly Language

As an example, an assembly language implementation of the Data CRC calculation for the 68HC11 (an eight-bit processor with a simple instruction set) is presented. The routine accumulates the CRC into the variables CRCLO and CRCHI. T1 and T2 are temporary storage variables. Most instructions act on either accumulator A or B.

```
DATACRC:                        ;ACCUMULATE THE OCTET (0,X) INTO THE DATA CRC
        LDAA    0,X             ;FETCH DATA OCTET (INDEXED OPERATION)
        EORA    CRCLO           ;D7-D0 EXCLUSIVE OR C7-C0
        STAA    CRCLO           ;SAVE RESULT
        CLRB                    ;CLEAR REGISTER B
        LSRA                    ;SHIFT A RIGHT, 0 INTO MSB, LSB INTO CARRY
        RORB                    ;ROTATE B RIGHT, CARRY INTO MSB
        LSRA
        RORB
        LSRA
        RORB
        LSRA
        RORB
        ; A=(- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
        ;
        EORA    CRCLO
        ANDA    #0FH            ;MASK OFF ALL BUT LS 4 BITS
        ; A=(- - - - 7 6 5 4) B=(3 2 1 0 - - - -)
        ;    (- - - - 3 2 1 0)
        ;
        STAA    T1              ;SAVE TEMP RESULT (NOT USED EXCEPT AS TEMP)
        STAB    T2
        LSRA                    ;SHIFT RIGHT 1 BIT
        RORB
        ; A=(- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
        ;    (- - - - - 3 2 1)   (0 - - - - - - -)
        ;
        EORA    CRCLO           ;COMBINE PARTIAL TERMS
        EORA    T2
        EORB    CRCHI           ;C8-C15
        EORB    T1
        ; A=(- - - - - 7 6 5) B=(4 3 2 1 0 - - -)
        ;    (- - - - - 3 2 1)   (0 - - - - - - -)
        ;    (7 6 5 4 3 2 1 0)   (15 14 13 12 11 10 9 8)
        ;    (3 2 1 0 - - - -)   (- - - - 7 6 5 4)
        ;                        (- - - - 3 2 1 0)
        ;
        STAA    CRCHI           ;SAVE RESULT
        STAB    CRCLO
        RTS
```

G.2.3 Other Implementations of the Data CRC Algorithm

Other implementations of the Data CRC algorithm are possible. It can be seen that the new CRC value may be factored into the exclusive OR of two terms. One term is the most significant octet of the prior CRC value. The other term is a function of the least significant octet of the prior CRC value exclusive OR'ed with the data value. A lookup table with 256 elements may be used to quickly determine the value of the second term, using the least significant octet of the prior CRC exclusive OR'ed with the data value as an index. This term may then be exclusive OR'ed with the most significant

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

octet of the prior CRC value to form the new CRC value. The contents of the table may be computed using the implementation shown in Clause G.2.1 or G.2.2.

G.3 Calculation of the Encoded CRC-32K

The equivalence of hardware and software methods for calculating CRCs is demonstrated by the examples shown previously in Clauses G.1 and G.2. There are numerous open source examples of CRC algorithms available, including "A Painless Guide to CRC Error Detection Algorithms." It describes a library that can implement arbitrary CRC systems based on input parameters such as CRC length, polynomial, initial value, shift direction (based on the transmission order of the underlying data link), etc.

G.3.1 Sample Implementation of the CRC-32K in C

An example C language implementation for the CRC-32K (Koopman) polynomial is shown below. Inputs are the octet to be processed and the accumulated CRC value (i.e. the value of the CRC32K variable defined in Clause 9.5.2). The function return value is the updated CRC (to be copied back into the CRC32K variable). An example of use is shown in Clause T.1. Note that Koopman expresses CRC polynomial representations in X^{32} ... X^{1} order (X^{0} is implied because it is always set to X'01'). The polynomial specified in Clause 9.6 is represented in his notation as X'BA0DC66B'. When this is reversed and represented in X^{0} ... X^{31} order (X^{32} is implied because it is always set to X'01'), the equivalent polynomial representation becomes X'EB31D82E'.

```
#include <stdint.h>

/* Accumulate "dataValue" into the CRC in "crc32kValue".
 * Return value is updated CRC.
 *
 * Assumes that "uint8_t" is equivalent to one octet.
 * Assumes that "uint32_t" is four octets.
 * The ^ operator means exclusive OR.
 */
uint32_t
CalcCRC32K(uint8_t dataValue, uint32_t crc32kValue)
{
  uint8_t data, b;
  uint32_t crc;

  data = dataValue;
  crc  = crc32kValue;

  for (b = 0; b < 8; b++) {
    if ((data & 1) ^ (crc & 1)) {
      crc >>= 1;
      crc ^= 0xEB31D82E; /* CRC-32K polynomial, 1 + x**1 + ... + x**30 (+ x**32) */
    } else {
      crc >>= 1;
    }
    data >>= 1;
  }
  return crc;  /* Return updated crc value */
}
```

In operation, the value of the CRC32K variable is initialized to all ones. During transmission, the Encoded Data octets are passed through the calculation after encoding but before transmission. The ones' complement of the final CRC32K variable value is then ordered least-significant octet first (i.e. in right-to-left order) and COBS-encoded according to Clause 9.10.2. At the receiving end, the Encoded Data octets are passed through the calculation before decoding. Then, the received Encoded CRC-32K field is decoded according to Clause 9.10.3 and the resulting octets (i.e. the ones' complement of the sender's CRC32K, least-significant octet first) are passed through the calculation. If all the octets are received correctly, then the final value of the receiver's CRC32K variable (termed the "residue") will be the constant X'0843323B'. NOTE: the correct residue for a given polynomial may be determined by passing a string of zeros equal in length to the size of the CRC register through the calculation.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
| 0000 | 1000 | 0100 | 0011 | 0011 | 0010 | 0011 | 1011  |
|<- 0 ->|<- 8 ->|<- 4 ->|<- 3 ->|<- 3 ->|<- 2 ->|<- 3 ->|<- B ->|
```

As a simplified example of usage, consider a non-encoded data sequence X'01', X'22', X'30':

Description	Value	CRC-32K after octet is processed
		X'FFFFFFFF' (initial value)
first data octet	X'01'	X'56381747'
second data octet	X'22'	X'12557D20'
third data octet	X'30'	X'83DD5A41'
		(ones' complement is X'7C22A5BE')
CRC1 (least significant octet)	X'BE'	X'C0D1F128'
CRC2	X'A5'	X'1C708944'
CRC3	X'22'	X'7FC2C8BD'
CRC4 (most significant octet)	X'7C'	X'0843323B'

Thus, the sender would calculate the CRC-32K on the three octets X'01', X'22', X'30' to be X'83DD5A41'. The ones' complement of this is X'7C22A5BE', which is appended to the frame least significant octet first as X'BE', X'A5', X'22', X'7C'.

The receiver would calculate the CRC-32K on the seven octets X'01', X'22', X'30', X'BE', X'A5', X'22', and X'7C' to be X'0843323B', which is the expected result for a correctly received frame.

G.3.2 Sample Implementation of the CRC-32K in Assembly Language

One way to generate an efficient assembly language implementation of the CRC-32K function is to cross-compile the C example above for a given target processor, examine the assembly language listing, and hand optimize register usage, etc. An assembly language implementation of the CRC-32K for the 8-bit AVR[1] instruction set is presented below. The avr-gcc calling conventions are observed and only scratch registers available for use by the compiler are modified.

```
#include <stdint.h>

; uint32_t
; CalcCRC32K(uint8_t dataValue, uint32_t crcValue)
;
; call:
;
;   r24     dataValue
;   r23-r20 crcValue (LSB-MSB)
;
; return:
;
;   r25-r22 updated crcValue (LSB-MSB)
;
CalcCRC32K:
    ldi     r18, 0x01    ; r18 = constant ONE
    ldi     r19, 0x00    ; for (b = 0; ...

_for:
; Calculate and save result of ((data ^ crc) & 1)

    mov     r25, r24     ; data lsb
    eor     r25, r20     ; ^crc msb
    and     r25, r18
```

[1] AVR is a registered trademark of Atmel Corporation.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

```
; crc >>= 1 in either case

    lsr     r23
    ror     r22
    ror     r21
    ror     r20

; If result of previous calculation was '1', accumulate feedback

    cp      r25, r18
    brne    _else

; crc ^= 0xEB31D82E; CRC-32K polynomial, 1 + x**1 + ... + x**30 (+ x**32)

    ldi     r25, 0xEB
    eor     r23, r25
    ldi     r25, 0x31
    eor     r22, r25
    ldi     r25, 0xD8
    eor     r21, r25
    ldi     r25, 0x2E
    eor     r20, r25

_else:
; data >>= 1;

    lsr     r24

    add     r19, r18    ; b++;
    cpi     r19, 0x08
    brlt    _for        ; b < 8;

    movw    r24,r22     ; copy 32-bit return value to r25-r22
    movw    r22,r20
    ret
```

G.3.3 Parallel Implementation of the CRC-32K

Other implementations of the CRC-32K algorithm are possible. It can be seen that the new CRC-32K value may be factored into the exclusive OR of two terms. One term is the most significant octet of the prior CRC-32K value. The other term is a function of the least significant octet of the prior CRC-32K exclusive OR'ed with the data value. A lookup table with 256 elements may be used to quickly determine the value of the second term, using the least significant octet of the prior CRC-32K exclusive OR'ed with the data value as an index. This term may then be exclusive OR'ed with the most significant octet of the prior CRC-32K value to form the new CRC-32K value. The contents of the table may be computed using an implementation similar to the one shown in Clause G.3.1. A sample implementation appears below.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

```
#include <stdint.h>

void
CreateCRC32Table()
{
  uint16_t data;
  uint32_t crc;
  uint16_t b;

  printf( "static const uint32_t CRC32Table[256] = {" );
  for (data = 0; ; ) {
    if (data % 8 == 0)
      printf("\n");

    crc = data & 0xFF;
    for (b = 0; b < 8; b++) {
      if (crc & 1) {
          crc >>= 1;
          crc ^= 0xEB31D82E;
        } else {
          crc >>= 1;
        }
    }
    printf( "0x%08lX", crc );

    if (++data == 256)
      break;
    printf( ", " );
  }
  printf( "\n};\n" );
}


/* Update running "crcValue" with "dataValue"
 * The crcValue shall be initialized to all ones.
 *
 * For transmission, the returned value shall be complemented and then
 * sent low-order octet first (i.e. right to left).
 *
 * On reception, if Data ends with a correct CRC, the returned value
 * will be 0x08433C3B (0000 1000 0100 0011 0011 0010 0011 1011).
 */
uint32_t
LookupCRC32K(uint8_t dataValue, uint32_t crcValue)
{
  crcValue = CRC32Table[(crcValue ^ dataValue) & 0xFF] ^ (crcValue >> 8);
  return (crcValue);
}
```

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX H - COMBINING BACnet NETWORKS WITH NON-BACnet NETWORKS (NORMATIVE)

(This annex is part of this standard and is required for its use.)

H.1 BACnet Gateways

There are three basic approaches to BACnet gateways: mapping non-BACnet networks onto BACnet routers, modeling a single non-BACnet device as multiple BACnet devices and mapping multiple non-BACnet devices into a single device.

H1.1 Modeling non-BACnet Devices as BACnet Devices

H.1.1.1 Mapping Non-BACnet Networks onto BACnet Routers

In addition to providing the means to interconnect multiple BACnet networks, BACnet routers may also be used to provide a gateway function to non-BACnet networks. Non-BACnet networks are characterized by the use of message structures, procedures, and medium access control techniques other than those contained in this standard. The mapping from BACnet to non-BACnet networks is performed by extending the routing table concept to allow non-BACnet devices to be addressable using BACnet NPCI. Thus, each non-BACnet network is assigned a unique two-octet network number, and each device on the non-BACnet network is represented by a "MAC address" that may, or may not, correspond to the actual octets used to address the device using the medium access control in use on the foreign network. Since communication with devices on non-BACnet networks is, by definition, not standardized here, the specific procedures for interpreting, translating, or relaying messages received by such a router-gateway from either the BACnet or non-BACnet ports are a local matter.

The non-BACnet network that is modeled in this manner by a BACnet gateway is referred to as a virtual BACnet network and the devices modeled on that network are referred to as virtual BACnet devices.

H.1.1.2 Multiple "Virtual" BACnet Devices in a Single Physical Device

A BACnet device is one that possesses a Device object and communicates using the procedures specified in this standard. In some instances, however, it may be desirable to model the activities of a physical building automation and control device through the use of more than one BACnet device. Each such device will be referred to as a virtual BACnet device. This can be accomplished by configuring the physical device to act as a router to one or more virtual BACnet networks. The idea is that each virtual BACnet device is associated with a unique DNET and DADR pair, i.e. a unique BACnet address. The physical device performs exactly as if it were a router between physical BACnet networks.

H.1.1.3 Modeling non-BACnet Data as Objects in a Single BACnet Device

In many situations, the amount of data provided by a non-BACnet device is small, and there is no benefit in exposing the non-BACnet devices as individual BACnet devices. In these cases, the gateway provides access to data in the non-BACnet devices through objects within the gateway device. The methods used for interaction between the gateway and the non-BACnet devices, and the methods used for mapping data into BACnet objects is a local matter.

H.2 Requirements and Best Practices for BACnet Gateway Implementations

The design and implementation of BACnet gateways is a local matter. However, there are certain design choices that result in better interoperability between the BACnet devices, the BACnet gateway, and the non-BACnet devices.

Informative Note: This clause lists some of the design choices that have been shown to provide a positive user experience. In this clause, therefore, the word "should" is used to indicate a recommendation that conforms to those design choices.

H.2.1 General Best Practices

H.2.1.1 Caching Writes to Non-BACnet Devices

To achieve reasonable performance on the BACnet network, it is not always feasible for a gateway to wait for data to be written through to non-BACnet devices. As such, it is acceptable for gateways to return a Result(+) to a WriteProperty or WritePropertyMultiple request even though the values have not been written through to the non-BACnet device. It is also acceptable for the gateway to return the previously written value to a subsequent ReadProperty or

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ReadPropertyMultiple even if it has not been written through to the non-BACnet device. This may result in a slight difference in behavior between native BACnet data and data from non-BACnet devices.

H.2.1.2 Input, Output, and Value Objects

The choice of modeling non-BACnet inputs, outputs, and values as BACnet Input, Output, or Value objects is a local matter; however, the following guidance is provided.

For non-BACnet inputs:

(a) BACnet Input objects are recommended for non-BACnet inputs. When the Present_Value is made writable by taking the object out of service, values written to the Present_Value should not be assumed to be written through to the non-BACnet input.

For non-BACnet outputs:

(a) BACnet Output objects are recommended for use when the non-BACnet output is controlled only through BACnet and therefore the BACnet Present_Value can normally be assumed to be equal to the non-BACnet output value.

(b) Writable non-commandable BACnet Value objects are recommended for use when control of the non-BACnet output is shared between BACnet and non-BACnet entities. In this case, is it typical for the gateway to periodically read the non-BACnet output so that the BACnet Present_Value matches the non-BACnet output.

For non-BACnet values:

(a) Commandable BACnet Value objects are recommended when the non-BACnet value is controlled only through BACnet and therefore the BACnet Present_Value can normally be assumed to be equal to the non-BACnet value.

(b) Writable, non-commandable BACnet Value objects are recommended when control of the non-BACnet value is shared between BACnet and non-BACnet entities. In this case, is it typical for the gateway to periodically read the non-BACnet value so that the BACnet Present_Value matches the non-BACnet value.

(c) Non-writable, non-commandable BACnet Value objects are recommended for the case where the non-BACnet value is only being monitored and is not being controlled from BACnet. When the Present_Value is made writable by taking the object out of service, values written to the Present_Value should not be assumed to be written through to the non-BACnet value.

H.2.1.3 Priority_Array Handling

Gateways are required to implement Priority_Array properties correctly with all 16 entries as defined in Clause 19.2.

H.2.1.4 Handling Requests That Take Too Long

Confirmed requests that cannot be fulfilled within the allowed APDU_Timeout shall result in an abort PDU being returned to the client. This condition may be caused by requests for too much data or for too many properties that are not cached in the gateway. One condition when this would occur is when a ReadPropertyMultiple request is received that would require the gateway to communicate with more non-BACnet devices that it can within the APDU_Timeout because the values are not cached in the gateway. Under such conditions, the gateway shall return the abort PDU with an abort reason of APPLICATION_EXCEEDED_REPLY_TIME, and the client is expected to retry the request with fewer properties.

H.2.1.5 Non-BACnet Devices That Sleep

When a gateway is providing access to a non-BACnet device that sleeps (i.e. the device is not always available to answer requests), the gateway shall cache all data and answer all BACnet requests using a data cache. Later, when the sleeping device wakes up, data updates are sent to the device, and reads are performed as required.

If non-BACnet devices are expected to wake up and report their status on a pre-determined schedule, when a device does not report its status, it shall be treated as if it is offline (see the following sections on how a gateway should treat offline devices).

H.2.2 Virtual Network Gateways

H.2.2.1 Offline Devices

When modeling devices as a virtual BACnet network of devices, communication timeout errors between the gateway and the non-BACnet device shall not result in the gateway returning an error, abort, or reject PDU in response to BACnet

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

requests directed to the non-BACnet device. Instead, the gateway shall remain silent. This gives the best indication to the BACnet client that the device is offline.

When a gateway has determined that a non-BACnet device is offline, it shall also not send I-Am or I-Have requests on behalf of that device.

H.2.2.2 Spread Out I-Am, I-Have Requests

The gateway device is responsible for sending the I-Am and I-Have requests for the non-BACnet devices. When the number of non-BACnet devices increases, so too will the number of I-Am and I-Have requests that the gateway will be sending. In order to ease network traffic, the gateway should space out I-Am and I-Have requests.

H.2.2.3 Prepare for Numerous Outstanding Requests

Gateways to virtual BACnet networks should be prepared for a large number of simultaneous requests. Many client devices will restrict the number of outstanding requests to any particular device. Due to the gateway responding on behalf of a number of devices, such client niceties will not have the same benefit for the gateway device. As such, gateway devices should be prepared for significantly more traffic than a single normal BACnet device would receive.

H.2.3 Single Device Gateways

H.2.3.1 Organization of Data

When modeling non-BACnet data as objects within the gateway device, as the number of data points mapped into the gateway increases, the inter-data relationships become less apparent. In order to improve the organization of the mapped data points, gateways should use Structured View objects.

H.2.3.2 Offline Devices

Communication errors between the gateway and the non-BACnet device should be indicated via the Reliability property and Fault flag in the associated BACnet objects. The gateway should support initiation of event notifications to report such faults.

Properties that exist in objects in the gateway device shall return values when read and not errors, even when the non-BACnet device from which those values are normally read is offline. It is not acceptable for this style of gateway device to not respond to confirmed requests just because the non-BACnet device is offline.

H.3 Using BACnet with the DARPA Internet Protocols

This clause describes procedures whereby BACnet messages may be conveyed using the protocols developed by the Defense Advanced Research Projects Agency (DARPA) of the Department of Defense (DoD). Collectively, these protocols are known as the Internet Protocol suite. The methodology described in this appendix involves encapsulating/decapsulating BACnet LSDUs and conveying them across an internet using the capabilities of IP routers, a technique often referred to as "tunneling." Although this appendix will describe the procedures in terms of a BACnet/Internet Protocol Packet-Assembler-Disassembler, the necessary functionality could be built into BACnet nodes directly.

H.3.1 BACnet/Internet Protocol Packet-Assembler-Disassembler (B/IP PAD)

A B/IP PAD is a device that implements the BACnet network layer as described in Clause 6 of this standard as well as the DoD User Datagram Protocol (UDP) and the Internet Protocol (IP).

Upon receipt of a BACnet packet from the local network, the B/IP PAD inspects the NPCI to see if a DNET network number has been supplied. If so, the B/IP PAD then consults an internal table consisting of entries mapping network numbers, IP addresses of all B/IP PADs within the BACnet internetwork, and the IP address of an IP router on the local network that represents the next hop for the IP datagram. If an appropriate entry is found, the B/IP PAD encapsulates the LSDU portion of the BACnet message (see Figure H-1) in a UDP packet where the LSDU represents the data portion of the packet. The UDP source and destinations ports shall be set to X'BAC0'. The B/IP PAD shall then send an IP datagram containing the UDP packet to the local IP router indicating its own IP address as the source address and the IP address of the B/IP PAD corresponding to the DNET contained in the BACnet message as the destination IP address. If the DNET in the BACnet

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

message is the global broadcast address, this procedure shall result in the transmission of the BACnet LSDU to all B/IP PADs contained in the table. Conveyance of packets between B/IP PADs follows standard IP procedures.

Upon receipt of an IP datagram from an IP router, a B/IP PAD shall locate the BACnet process at UDP port X'BAC0', which shall prepare the data portion of the UDP datagram for transmission as a BACnet message on the local network using the procedures defined in Clause 6.3.

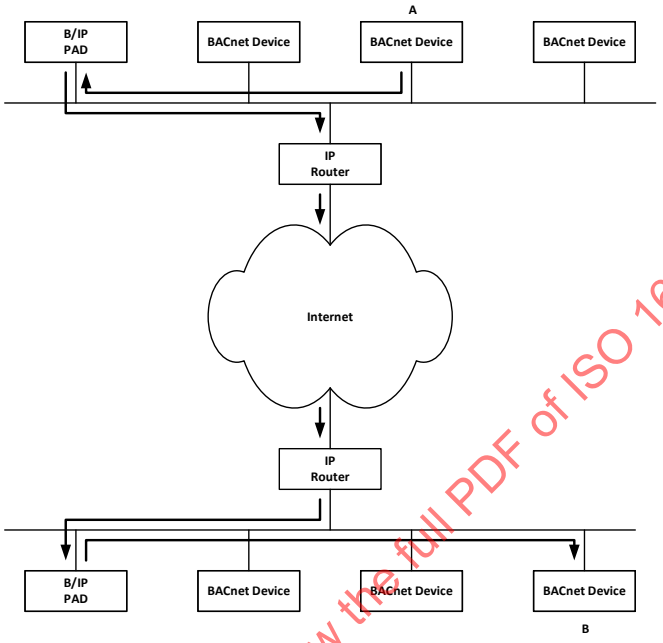


Figure H-1. IP tunneling implemented with a B/IP PAD.

H.3.2 Implementation Notes

An implementation on an 8802-3 LAN might be configured as shown in Figure H-1. BACnet packets are differentiated from IP packets and those of other protocols by the LSAP contained in the LLC header. IP uses an LSAP of X'06', whereas the BACnet network layer is identified by an LSAP of X'82'. In the case shown in the figure, each packet actually appears twice on the network, once as a BACnet message and once as an IP message. B/IP PADs could also be constructed to implement the IP routing procedure for any IP packet in addition to performing the BACnet encapsulation/decapsulation. Such a device would be a B/IP PAD/Router. This is illustrated in Figure H-2.

For further information about the DoD protocols, consult the DDN Protocol Handbook referenced in Clause 25.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

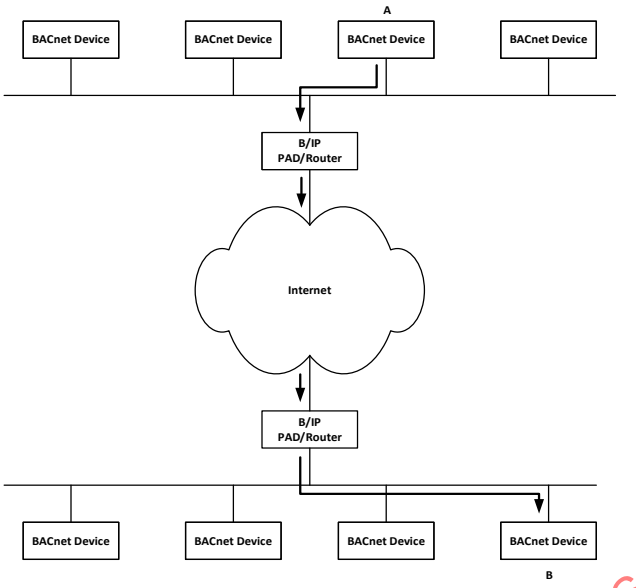


Figure H-2. IP tunneling implemented with a B/IP PAD router.

H.4 Using BACnet with the IPX Protocol

This clause describes procedures whereby BACnet messages may be conveyed using the protocols developed by Novell Corporation, which are known as Internetwork Datagram Protocol or, more popularly, IPX. This protocol layer is based on a subset of the Xerox Network Services (XNS) protocols, in particular using the Packet Exchange Protocol (PEP). The methodology described in this clause involves encapsulating/decapsulating BACnet LSDUs and conveying them across an IPX internetwork using the capabilities of IPX routers, a technique that here will be called IPX Tunneling. Although this appendix will describe the procedures in terms of a BACnet/IPX packet-assembler-disassembler, the necessary functionality could be built into BACnet nodes directly.

H.4.1 BACnet/IPX Packet-Assembler-Disassembler (B/IPX PAD)

A B/IPX PAD is a device that implements both the BACnet network layer as described in Clause 6 of this standard, as well as the IPX network layer.

Upon receipt of a BACnet packet from the local network, the B/IPX PAD inspects the NPCI to see if a DNET network number has been supplied. If so, the B/IPX PAD then consults an internal table consisting of entries mapping BACnet network numbers to (IPX network number, MAC layer address) pairs that represent all remote B/IPX PADs within the BACnet internetwork. If an appropriate entry is found, the B/IPX PAD encapsulates the LSDU portion of the BACnet message (see Figure 6-1) in an IPX Tunnel packet (see Figure H-3) where the destination network and MAC-layer addresses correspond to those of the remote B/IPX PAD and the destination socket number shall be set to the BACnet IPX socket number X'87C1'. The B/IPX PAD shall then send an IPX datagram containing this tunnel packet to the local IPX router, indicating its own IPX network number, MAC-layer address, and the BACnet IPX socket number as the source. If the DNET in the BACnet message is the global broadcast address, this procedure shall result in the transmission of the BACnet LSDU tunnel packet to all B/IPX PADs contained in the table. Conveyance of packets between B/IPX PADs follows standard IPX routing procedures.

Upon receipt of an IPX datagram using the BACnet socket, a B/IPX PAD shall prepare the data portion of the BACnet IPX Tunnel packet for transmission as a BACnet message on the local network using the procedures defined in Clause 6.3.

H.4.2 Implementation Notes

An implementation on an 8802-3 LAN might be configured as shown in Figure H-1. BACnet packets are differentiated from IPX packets and those of other protocols by the LSAP contained in the LLC header. IPX uses an LSAP of X'E0', whereas the BACnet network layer is identified by an LSAP of X'82'. In the case shown in the figure, each packet actually appears twice on the network: once as a BACnet message and once as an IPX message. B/IPX PADs could also be constructed to implement the IPX routing procedure for any IPX packet in addition to performing the BACnet encapsulation/decapsulation. Such a device would be a B/IPX PAD/Router. This is illustrated in Figure H-2.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

References
NetWare® System Interface Technical Overview
Addison-Wesley Publishing Company, Reading Massachusetts
ISBN 0-201-57027-0

```
;IPX Header (30 octets)
        dw      X'FFFF'         ;XNS checksum (not used by IPX)
        dw      ?               ;packet length (set by IPX)
        db      ?               ;routers crossed (set by IPX)
        db      4               ;type 4=Packet Exchange Packet
        dd      ?               ;32 bit dest network number
        db      ?,?,?,?,?,?     ;dest MAC layer address
        dw      X'87C1'         ;dest BACnet socket
        dd      ?               ;32 bit source network number
        db      ?,?,?,?,?,?     ;source MAC layer address
        dw      X'87C1'         ;source BACnet socket
        db 546 dup ?            ;BACnet LSDU
```

Figure H-3. Structure of a BACnet IPX tunnel packet.

H.5 Using BACnet with EIB/KNX

This clause describes how BACnet objects and properties are mapped to corresponding EIB/KNX Datapoints and Functional Blocks. Functional Blocks are part of the Interworking Model defined by the EIB Association / Konnex Association. In the following clauses, references to "EIB" also apply to Konnex.

H.5.1 Object Structures

The following clauses describe the relationship between EIB Functional Blocks and BACnet objects.

H.5.1.1 EIB

EIB Functional Blocks not only describe the semantics of a function in the English language, but also define how to access the services associated with that function. This is done on the basis of "Datapoints." The Datapoints are divided into two principal categories, input and output. A Functional Block consists of a non-empty collection of one or more input and output Datapoints. At least one Datapoint is required.

Functional Blocks are contained within Physical Devices. A Physical Device implements at least one Functional Block. It may be regarded as a container for, or collection of, Functional Blocks.

H.5.1.2 BACnet

BACnet's object types define functions in terms of semantics and the services used to access these functions. To accomplish this task, BACnet object types contain properties. An object type consists of a non-empty collection of properties, some of which are mandatory while others may be optional.

BACnet also defines a Device object and a "BACnet device" contains a collection of instances of object types. Each BACnet device contains one, and only one, Device object. See Clause 12.11. Typically, each physical device corresponds to a single BACnet device and therefore contains a single Device object. An exception is described in Clause H.2.

H.5.1.3 Relationship of EIB to BACnet

EIB Functional Blocks are comparable to BACnet object types while EIB Datapoints correlate to BACnet properties.

H.5.2 Mapping Rules for the Device Object Type

This clause provides rules for the assignment of values to the properties of the Device object type.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

H.5.2.1 Object_Identifier

The Object_Identifier of a Device object is of type BACnetObjectIdentifier and shall be unique internetwork-wide. See Clause 12.11.1. The encoding is defined in Clause 20.2.14. The Object Type field (the upper 10 bits) contains the enumerated value of the BACnetObjectType. In this mapping, the content of the 22-bit Instance Number field depends on whether the Object_Identifier is identifying a Device object or some other type of object. Subject to the uniqueness constraint, a one-to-one mapping of EIB Physical Devices to BACnet devices can be achieved by setting the upper 6 bits of the instance number to a unique EIB subnetwork identifier. This could be a part of an EIB Domain Address if it is available and is unique over all linked projects. The lower 16 bits of the instance number shall be set to the Individual Address of the EIB device. Other mapping algorithms shall also ensure that the Device's BACnetObjectIdentifier remains unique internetwork wide.

H.5.2.2 Object_Name

This CharacterString is unique internetwork-wide. The string is generated from the EIB Project-Installation ID and the EIB Individual Address. The Object_Name is "EIB_Project-Installation_ID::EIB_Individual_Address". Example: For an EIB Project-Installation ID of X'0011' = D'17' and an EIB Individual Address of 1.6.7, the Object_Name string would be "17::1.6.7".

H.5.2.3 Object_Type

The value of this property shall be DEVICE (= D'8').

H.5.2.4 System_Status

Clause 12.10.4 lists the values for this BACnet property. EIB device status is determined by two variables: LoadStateMachine (LSM) and RunStateMachine (RSM). The BACnet OPERATIONAL_READ_ONLY property value is not supported by EIB devices. Table H-1 lists the relationship between EIB RunStateMachine and EIB LoadStateMachine values and BACnet System_Status.

Table H-1. Mapping of the EIB device status to BACnet System_Status

BACnet System_Status	EIB RunStateMachine	EIB LoadStateMachine
OPERATIONAL	running	Loaded
OPERATIONAL_READ_ONLY	-	-
DOWNLOAD_REQUIRED	ready	Unloaded
DOWNLOAD_IN_PROGRESS	ready	Loading
NON_OPERATIONAL	halted	Error

H.5.2.5 Vendor_Name

This CharacterString identifies the manufacturer of the EIB device, with the EIB manufacturer code in parentheses. Example: "XYZ Company (1)"

H.5.2.6 Vendor_Identifier

This is the unique Vendor Identifier code assigned by ASHRAE. If the vendor has no BACnet Vendor Identifier, this property shall be set to the Vendor Identifier for EIBA: D'74'.

H.5.2.7 Model_Name

This is the model name of the EIB device as registered with the EIBA / Konnex Association.

H.5.2.8 Firmware_Revision

The firmware revision shall be the mask version of the EIB device. The content is equal to [EIB::DeviceObject:PID_FIRMWARE_REVISION].

H.5.2.9 Application_Software_Revision

This property is the software revision of the application running on the EIB device. The content is equal to [EIB::DeviceObject:PID_PROGRAM_VERSION].

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

H.5.2.10 Protocol_Version

This is the version of the BACnet protocol supported by this device. See Clause 12.11.12.

H.5.2.11 Protocol_Revision

This is the minor revision level of the BACnet protocol supported by this device. See Clause 12.11.13.

H.5.2.12 Protocol_Services_Supported

This property indicates the BACnet protocol services supported by this device. See Clause 12.11.14.

H.5.2.13 Protocol_Object_Types_Supported

This property indicates the BACnet protocol object types supported by this device. See Clause 12.11.15. The protocol object types supported shall be at least Analog Input, Analog Output, Analog Value, Binary Input, Binary Output, and Binary Value.

H.5.2.14 Object_List

This property is a BACnetARRAY of BACnetObjectIdentifier. See Clause 12.11.16.

H.5.2.15 Max_APDU_Length_Accepted

The value of this property shall be greater than or equal to 50. See Clause 12.11.18.

H.5.2.16 Segmentation_Supported

See Clause 12.11.19.

H.5.2.17 APDU_Timeout

See Clause 12.11.28.

H.5.2.18 Number_Of_APDU_Retries

See Clause 12.11.29.

H.5.2.19 Device_Address_Binding

See Clause 12.11.34.

H.5.2.20 Database_Revision

See Clause 12.11.35. This value shall be updated anytime the EIB device configuration is changed.

H.5.3 Mapping Rules for Other BACnet Object Types

This clause provides rules for the assignment of values to specific properties of these BACnet object types: Analog Input, Analog Output, Analog Value, Binary Input, Binary Output, and Binary Value.

H.5.3.1 Object_Identifier

An Object_Identifier is a 32-bit number that shall be unique within a BACnet device. The encoding is defined in Clause 20.2.14. The Object Type field (the upper 10 bits) contains the enumerated value of the BACnetObjectType. In this mapping, the content of the 22-bit Instance Number field depends on whether the Object_Identifier is identifying a Device object or some other type of object. The Object_Identifiers of other object types shall be unique within the BACnet device that contains them. See Clause 12. Subject to this constraint, the upper 6 bits of the instance number shall uniquely identify each instance of a mapped object contained within a BACnet device while the remaining 16 bits shall contain the EIB Individual Address to simplify diagnostics.

H.5.3.2 Object_Name

This CharacterString is unique within the BACnet device. The string is generated from the EIB Project-Installation ID, the EIB Individual Address, and the EIB Functional Block ID - Instance. The Object_Name is "EIB_Project-

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Installation_ID::EIB_Individual_Address#Functional_Block_ID-Instance". Example: For an EIB Project-Installation ID of X'0011' = D'17', an EIB Individual Address of 1.6.7, a Functional Block ID of D'10' and a Functional Block Instance of D'2' the Object_Name string would be "17::1.6.7#10-2".

H.5.3.3 Object_Type

The value of this property shall be as listed below:

ANALOG-INPUT	D'0'
ANALOG-OUTPUT	D'1'
ANALOG-VALUE	D'2'
BINARY-INPUT	D'3'
BINARY-OUTPUT	D'4'
BINARY-VALUE	D'5'

H.5.3.4 Present_Value

This property contains the present value of the Input / Output / Value. For Binary object types the permissible values are ACTIVE and INACTIVE. For Analog object types the property value is a REAL.

H.5.3.5 Description

This property contains the EIB group address associated with the Present_Value property as a CharacterString with the notation "x/y/z", where "x" is the main group, "y" is the subgroup, and "z" is the function.

H.5.3.6 Status_Flags

The values for the status flags shall be set as:

IN_ALARM	The value of this flag shall be logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if manual override can be detected by the EIB device and is executed, otherwise logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

H.5.3.7 Event_State

The value of this property is always set to NORMAL.

H.5.3.8 Reliability

The optional "Reliability" property shall, if implemented, return at least NO_FAULT_DETECTED in case that the EIB Datapoints' Quality Codes are GOOD, and UNRELIABLE_OTHER in case at least one EIB Datapoint's Quality Code is BAD. An EIB Datapoint's Quality Code is GOOD if the Datapoint can be read from (Input) or written to (Output); otherwise its Quality Code is BAD. If an implementation is able to distinguish different sources of a failure, it may return other reliability codes of type BACnetReliability.

H.5.3.9 Out_Of_Service

The mandatory "Out_Of_Service" property shall be set to TRUE if the corresponding EIB device cannot be reached or the Present_Value cannot be read from (Input) or written to (Output) the device. Otherwise this property is set to FALSE.

H.5.3.10 Polarity

The value of this property shall always be NORMAL.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

H.5.3.11 Units

The BACnet Analog Input/Output/Value Present_Value is of datatype REAL. The EIB datatypes 8-Bit Unsigned Value, 8-Bit Signed Value, 2-Octet Unsigned Value, 2-Octet Signed Value, 2-Octet Float Value, 4-Octet Unsigned Value, 4-Octet Signed Value, and 4-Octet Float Value are mapped into the datatype REAL.

These datatypes encompass a larger number of Datapoint types that are mapped into a REAL with a BACnetEngineeringUnit. See Table H-2.

Table H-2. EIB Datapoint Types

ID	Name	Range	Units	BACnetEngineeringUnits
5.001	DPT_Scaling	0...100	%	(98) percent
5.003	DPT_Angle	0...360	°	(90) degrees-angular
5.010	DPT_Value_1_Ucount	0...255	counter pulses	
6.010	DPT_Value_1_Count		counter value	
7.001	DPT_Value_2_Ucount		counter pulses (16-bit unsigned value)	
8.001	DPT_Value_2_Count		counter pulses	
9.001	DPT_Value_Temp	-273...+670760	C	(62) degrees-Celsius
9.002	DPT_Value_Tempd	-670760...+670760	K	(63) degrees-Kelvin
9.003	DPT_Value_Tempa	-670760...+670760	K/h	(181) degrees-Kelvin-per-hour
9.004	DPT_Value_Lux	0...670760	Lux	(37) luxes
9.005	DPT_Value_Wsp	0...670760	m/s	(74) meters-per-second
9.006	DPT_Value_Pres	0...670760	Pa	(53) pascals
9.010	DPT_Value_Time1	-670760...+670760	s	(73) seconds
9.011	DPT_Value_Time2	-670760...+670760	ms	(159) milliseconds
9.020	DPT_Value_Volt	-670760...+670760	mV	(124) millivolts
9.021	DPT_Value_Curr	-670760...+670760	mA	(2) milliamperes
12.001	DPT_Value_4_Ucount	0...4294967295	counter pulses	
13.001	DPT_Value_4_Count	-2147483648... +2147483647	counter value	
14.000	DPT_Value_Acceleration		$m\ s^{-2}$	(166) meters-per-second-per-second
14.003	DPT_Value_Activity		s^{-1}	(101) per-second
14.005	DPT_Value_Amplitude		(unit as appropriate)	
14.006	DPT_Value_AngleRad		rad angle	(103) radians
14.007	DPT_Value_AngleDeg		° angle	(90) degrees-angular
14.008	DPT_Value_Angular_Momentum		J s	(183) joule-seconds
14.009	DPT_Value_Angular_Velocity		rad s^{-1}	(184) radians-per-second
14.010	DPT_Value_Area		m^2	(0) square-meters
14.011	DPT_Value_Capacitance		F	(170) farads
14.014	DPT_Value_Compressibility		$m^2\ N^{-1}$	(185) square-meters-per-Newton
14.015	DPT_Value_Conductance		$S = \Omega^{-1}$	(173) siemens
14.016	DPT_Value_ Electrical_Conductivity		$S\ m^{-1}$	(174) siemens-per-meter
14.017	DPT_Value_Density		$kg\ m^{-3}$	(186) kilograms-per-cubic-meter
14.019	DPT_Value_Electric_Current		A	(3) amperes

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ID	Name	Range	Units	BACnetEngineeringUnits
14.020	DPT_Value_Electric_CurrentDensity		A m^{-2}	(168) amperes-per-square-meter
14.023	DPT_Value_Electric_FieldStrength		V m^{-1}	(177) volts-per-meter
14.027	DPT_Value_Electric_Potential		V	(5) volts
14.028	DPT_Value_Electric_PotentialDifference		V	(5)volts
14.029	DPT_Value_ElectromagneticMoment		A m^2	(169) ampere-square-meters
14.030	DPT_Value_Electromotive_Force		V	(5) volts
14.031	DPT_Value_Energy		J	(16) joules
14.032	DPT_Value_Force		N	(153) newton
14.033	DPT_Value_Frequency		Hz = s^{-1}	(27) hertz
14.034	DPT_Value_Angular_Frequency		rad s^{-1}	(184) radians-per-second
14.035	DPT_Value_Heat_Capacity		J K^{-1}	(127) joules-per-degree-Kelvin
14.036	DPT_Value_Heat_FlowRate		W	(47) watts
14.037	DPT_Value_Heat_Quantity		J	(16) joules
14.038	DPT_Value_Impedance		W	(47) watts
14.039	DPT_Value_Length		m	(31) meters
14.040	DPT_Value_Light_Quantity		J or lm s	(16) joules
14.041	DPT_Value_Luminance		cd m^{-2}	(180) candelas-per-square-meter
14.042	DPT_Value_Luminous_Flux		lm	(36) lumens
14.043	DPT_Value_Luminous_Intensity		cd	(179) candelas
14.044	DPT_Value_Magnetic_FieldStrength		A m^{-1}	(167) amperes-per-meter
14.045	DPT_Value_Magnetic_Flux		Wb	(178) webers
14.046	DPT_Value_Magnetic_FluxDensity		T	(175) teslas
14.047	DPT_Value_Magnetic_Moment		A m^2	(169) ampere-square-meters
14.048	DPT_Value_Magnetic_Polarization		T	(175) teslas
14.049	DPT_Value_Magnetization		A m^{-1}	(167) amperes-per-meter
14.050	DPT_Value_MagnetomotiveForce		A	(3) amperes
14.051	DPT_Value_Mass		kg	(39) kilograms
14.052	DPT_Value_MassFlux		kg s^{-1}	(42) kilograms-per-second
14.053	DPT_Value_Momentum		N s	(187) newton-seconds
14.054	DPT_Value_Phase_AngleRad		rad	(103) radians
14.055	DPT_Value_Phase_AngleDeg		°	(14) degrees-phase
14.056	DPT_Value_Power		W	(47) watts
14.057	DPT_Value_Power_Factor		cos Φ	(15) power-factor
14.058	DPT_Value_Pressure		Pa = N m^{-2}	(53) pascals
14.059	DPT_Value_Reactance		Ω	(4) ohms
14.060	DPT_Value_Resistance		Ω	(4) ohms
14.061	DPT_Value_Resistivity		Ωm	(172) ohm-meters
14.062	DPT_Value_SelfInductance		H	(171) henrys
14.064	DPT_Value_Sound_Intensity		W m^{-2}	(189) watts-per-square-meter
14.065	DPT_Value_Speed		m s^{-1}	(74) meters-per-second
14.066	DPT_Value_Stress		Pa = N m^{-2}	(53) pascals
14.067	DPT_Value_Surface_Tension		N m^{-1}	(188) newtons-per-meter
14.068	DPT_Value_Common_Temperature		C	(62) degrees-Celsius
14.069	DPT_Value_Absolute_Temperature		K	(63) degrees-Kelvin
14.070	DPT_Value_TemperatureDifference		K	(63) degrees-Kelvin
14.071	DPT_Value_Thermal_Capacity		J K^{-1}	(127) joules-per-degree-Kelvin
14.072	DPT_Value_Thermal_Conductivity		W m^{-1} K^{-1}	(190) watts-per-meter-per-degree-Kelvin
14.073	DPT_Value_ThermoelectricPower		V K^{-1}	(176) volts-per-degree-Kelvin

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ID	Name	Range	Units	BACnetEngineeringUnits
14.074	DPT_Value_Time		s	(73) seconds
14.075	DPT_Value_Torque		N m	(160) newton-meters
14.076	DPT_Value_Volume		m^3	(80) cubic-meters
14.077	DPT_Value_Volume_Flux		$m^3 s^{-1}$	(85) cubic-meters-per-second
14.078	DPT_Value_Weight		N	(153) newton
14.079	DPT_Value_Work		J	(16) joules

H.5.3.12 Priority_Array

This is an internal implementation requirement, which is not required to be mapped. See Clause 19.

H.5.3.13 Relinquish_Default

For prioritized writable properties, it is typically required by the mapping that the Present_Value shall remain unchanged when no active entry (value not equal to NULL) is present. Therefore, the Relinquish_Default shall be set equal to the Present_Value.

H.5.3.14 Profile_Name

The profile name shall be set to "74-EIB_[Profile]", where [Profile] is the name of the EIB Function Block.

H.5.4 Mappings of EIB Functional Blocks

The following Functional Block mappings are specified in this clause: Analog Input, Analog Output, Analog Value, Binary Input, Binary Output, Binary Value and Dimming Actuator.

H.5.4.1 Overview

This clause provides mappings of standardized EIB Functional Blocks to BACnet objects. These are intended to provide both standardized definitions and to serve as illustrative examples of the mapping rules prescribed above. They therefore contain additional explanations and descriptive text.

For some properties, the mapping may not be to an EIB Datapoint but rather to a static value or to a function that transforms internal information to a BACnet datatype or format.

H.5.4.2 Analog Input

The Analog Input Functional Block is mapped to the standard BACnet Analog Input object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

Table H-3. Analog Input Mapping

Property Identifier	Property Datatype	O/R/W1	EIB Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	REAL	R	PID_ANALOG_PRESENT.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Units	BACnetEngineeringUnits	R	As specified in Clause H.5.3.11
Min_Pres_Value	REAL	O	As specified in Clause H.5.3.11, range lower value
Max_Pres_Value	REAL	O	As specified in Clause H.5.3.11, range higher value
COV_Increment	REAL	O	1.0
Profile_Name	CharacterString	R	"74-EIB_AnalogInput"

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

[1] O/R/W = Optional, required Readable or required Writable property (see Clause 12). In the context of the mapping, a property is always required if defined as mandatory for the BACnet object type, and additional Properties which are optional in the BACnet specification or are proprietary may be defined as mandatory for the mapping. If a Property is writable, the Property shall also be readable.

H.5.4.3 Analog Output

The Analog Output Functional Block is mapped to the standard BACnet Analog Output object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

Table H-4. Analog Output Mapping

Property Identifier	Property Datatype	O/R/W	Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	REAL	W	PID_ANALOG_SET.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Units	BACnetEngineeringUnits	R	As specified in Clause H.5.3.11
Priority_Array	BACnetPriorityArray	R	As specified in Clause H.5.3.12
Relinquish_Default	REAL	R	As specified in Clause H.5.3.13
Profile_Name	CharacterString	R	"74-EIB_AnalogOutput"

H.5.4.4 Analog Value

The Analog Value Functional Block is mapped to the standard BACnet Analog Value object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

Table H-5. Analog Value Mapping

Property Identifier	Property Datatype	O/R/W	Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	REAL	R	PID_ANALOG_PRESENT.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Units	BACnetEngineeringUnits	R	As specified in Clause H.5.3.11
COV_Increment	REAL	R	1.0
Profile_Name	CharacterString	R	"74-EIB_AnalogValue"

H.5.4.5 Binary Input

The Binary Input Functional Block is mapped to the standard BACnet Binary Input object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table H-6. Binary Input Mapping

Property Identifier	Property Datatype	O/R/W	EIB Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	BACnetBinaryPV	R	PID_BOOLEAN_PRESENT.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Polarity	BACnetPolarity	R	NORMAL
Profile_Name	CharacterString	O	"74-EIB_BinaryInput"

H.5.4.6 Binary Output

The Binary Output Functional Block is mapped to the standard BACnet Binary Output object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

Table H-7. Binary Output Mapping

Property Identifier	Property Datatype	O/R/W	EIB Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	BACnetBinaryPV	W	PID_BOOLEAN_SET.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Polarity	BACnetPolarity	R	NORMAL
Priority_Array	BACnetPriorityArray	R	As specified in Clause H.5.3.12
Relinquish_Default	BACnetBinaryPV	R	As specified in Clause H.5.3.13
Profile_Name	CharacterString	O	"74-EIB_BinaryOutput"

H.5.4.7 Binary Value

The Binary Value Functional Block is mapped to the standard BACnet Binary Value object type as the semantics of these two data structures are identical and the required properties of the BACnet object type can be mapped.

Table H-8. Binary Value Mapping

Property Identifier	Property Datatype	O/R/W	EIB Mapping
Object_Identifier	BACnetObjectIdentifier	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	R	As specified in Clause H.5.3.3
Present_Value	BACnetBinaryPV	R[1]	PID_BOOLEAN_SET.Value
Description	CharacterString	O	As specified in Clause H.5.3.5
Status_Flags	BACnetStatusFlags	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	R	NORMAL
Reliability	BACnetReliability	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	R	As specified in Clause H.5.3.9
Profile_Name	CharacterString	O	"74-EIB_BinaryValue"

[1] If Present_Value is commandable, then it is required to be writable. This property is required to be writable when Out_Of_Service is TRUE.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

H.5.4.8 Dimming Actuator

The Dimming Actuator Functional Block is mapped to the standard BACnet Analog Value object, which is extended by the addition of 4 EIB-specific properties.

The present value of the EIB Dimming Actuator is always an Unsigned Integer with a range of 0 to 255 (Unsigned8). These 256 values are linearly mapped to a percentage of the maximum output, i.e. 0 = 0%, 127 = 50%, 255 = 100%. The actual physical value, if required, shall be determined by the application.

The Dimming Actuator Functional Block specifies that the requested target state and current physical state be represented by different Datapoints. Although implementations may treat this internally as an identical device state (target requested and current state), this may not be the case when the dimming is the result of ramping or technically required delays. Therefore, for the two current state properties Present_Value and Present_Bin_Value, the corresponding target properties, Target_Value and Target_Bin_Value, have been defined.

Table H-9. Dimming Actuator Mapping

Property Identifier	Property Datatype	S/P[1]	O/R/W	Mapping
Object_Identifier	BACnetObjectIdentifier	S	R	As specified in Clause H.5.3.1
Object_Name	CharacterString	S	R	As specified in Clause H.5.3.2
Object_Type	BACnetObjectType	S	R	As specified in Clause H.5.3.3
Description	CharacterString	S	O	As specified in Clause H.5.3.5
Device_Type	CharacterString	S	O	Device's functional description from manufacturer data from ETS
Present_Value	REAL	S	R	PID_ANALOG_PRESENT.Value
Status_Flags	BACnetStatusFlags	S	R	As specified in Clause H.5.3.6
Event_State	BACnetEventState	S	R	NORMAL
Reliability	BACnetReliability	S	O	As specified in Clause H.5.3.8
Out_Of_Service	BOOLEAN	S	R	As specified in Clause H.5.3.9
Units	BACnetEngineeringUnits	S	R	As specified in Clause H.5.3.11
COV_Increment	REAL	S	R	1.0
Profile_Name	CharacterString	S	R	"74-EIB_DimmingActuator"
Target_Value	REAL	P	W	PID_ANALOG_SET.Value
Present_Bin_Value	BACnetBinaryPV	P	R	PID_BOOLEAN_PRESENT.Value
Target_Bin_Value	BACnetBinaryPV	P	W	PID_BOOLEAN_SET.Value
Dimming_Control[2]	REAL	P	O/W	PID_CONTROL_SET.Value

[1] S/P = Standard/Proprietary property
[2] Because BACnet specifies that a writable property shall also be readable, the value returned when the Dimming_Control property is read shall be 0.0

H.5.4.9 Defining Proprietary Object Types

If it is not possible to map an EIB Functional Block to an existing BACnet object type, a new proprietary object type shall be defined. Such object types shall be assigned an object type enumeration greater than 127 and contain the Profile_Name property to uniquely identify such object types and to provide a reference to an object-specific profile or description of the object type. At a minimum, proprietary object types shall have the Object_Identifier, Object_Name, Object_Type and Profile_Name properties. See Clause 23.

H.5.4.10 Defining Proprietary Properties

If it is possible to map a Functional Block to a standard BACnet object type but Datapoints exist within the EIB standard that cannot be mapped to existing object properties, or if a completely new proprietary object type with proprietary properties is defined, it will be necessary to define proprietary properties for that object type. Proprietary properties shall be assigned property identifiers greater than 511 and the profile pointed to by the Profile_Name property of the object type shall provide the property name, datatype and conformance code for each such profile-specific property. See Clause 23.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

H.5.5 Additional Information

This clause provides information on the EIB Functional Blocks Specification.

H.5.5.1 EIB Functional Blocks (FBs)

Within the Functional Block specifications, the Datapoints of a device are called properties and receive a property ID in terms of a unique name within the description of a functional block. The names of the properties always start with "PID_", as in the example PID_BOOLEAN_PRESENT.

Each such Datapoint property provides the following state information at run time:

Value: The current value of the property; it may be any kind of Datapoint Type.
Timestamp: Optional; it may be absolute or relative, depending on the device's capabilities.
Qualitycode: Either GOOD or BAD.

H.6 Using BACnet with the Former BACnet/WS Web Services Interface Defined by Annex N

Note that the SOAP-based Web Services interface defined by Annex N is deprecated for new designs. This clause is thus provided only for historical purposes.

This clause provides examples of the correspondence between BACnet/WS node attributes to specific properties of BACnet Objects. For some nodes and attributes, mapping might not be to a BACnet property but rather to a static value or to a function that transforms internal information to a BACnet datatype or concept.

H.6.1 Typical Mappings of BACnet/WS Attributes to BACnet Object Properties

The "normalized attributes", as defined by Annex N, are designed to provide an interoperable model of selected data to a Web services client. The following clauses define the correspondence of those attributes with BACnet properties.

H.6.1.1 Display Name

This attribute may correspond to the BACnet property Object_Name, except that DisplayName values do not need to be unique in the Web services data model.

H.6.1.2 Description

This attribute may correspond to the BACnet property Description.

H.6.1.3 Value and Related Attributes

The mappings for attributes related to the Value attribute, and its ValueType, vary according to BACnet object type, and may correspond as shown in the following table.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table H-10. Value and Value Related Attribute Mappings to BACnet Object Properties

BACnet Object Type	Value	ValueType	Units	Maximum	Minimum	Resolution
Accumulator	Present_Value	"Integer"	Units	Max_Pres_Value		
Analog Input	Present_Value	"Real"	Units	Max_Pres_Value	Min_Pres_Value	Resolution
Analog Output	Present_Value	"Real"	Units	Max_Pres_Value	Min_Pres_Value	Resolution
Analog Value	Present_Value	"Real"	Units			
Averaging	(varies)	"Real"				
Binary Input	Present_Value	"Boolean"				
Binary Output	Present_Value	"Boolean"				
Binary Value	Present_Value	"Boolean"				
Calendar	Present_Value	"Boolean"				
Command	Present_Value	"Integer"				
Device	System_Status	"Multistate"				
Event Enrollment	Event_State	"Multistate"				
Life Safety Point	Present_Value	"Multistate"				
Life Safety Zone	Present_Value	"Multistate"				
Loop	Present_Value	"Real"	Output_Units	Maximum_Output	Minimum_Output	
Multistate Input	Present_Value	"Multistate"				
Multistate Output	Present_Value	"Multistate"				
Multistate Value	Present_Value	"Multistate"				
Pulse Converter	Present_Value	"Real"	Units			
Schedule	Present_Value	(varies)				

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

H.6.1.4 Writable

This attribute may correspond to the PICS conformance statement declaration of writable for the BACnet property to which this node maps, but it may also vary depending on which user is making the Web services request or on other configuration or operational criteria.

H.6.1.5 InAlarm

This boolean attribute may correspond to the IN_ALARM flag in the BACnet property Status_Flags. If the IN_ALARM flag of that property is set to true, then InAlarm shall be true.

H.6.1.6 PossibleValues and WritableValues

The mapping for these attributes varies based on BACnet object type, and may be mapped according to the following table. The WritableValues attribute is always a subset of the PossibleValues attribute.

© ISO 2022 – All rights reserved

Table H-11. PossibleValues and WritableValues Attribute Mappings

BACnet Object Type	BACnet Property or Datatype Mapping
Binary Input	Active_Text, Inactive_Text properties
Binary Output	Active_Text, Inactive_Text properties
Binary Value	Active_Text, Inactive_Text properties
Command	Action_Text property
Device	BACnetDeviceStatus enumeration
Event Enrollment	BACnetEventState enumeration
Life Safety Point	BACnetLifeSafetyState enumeration
Life Safety Zone	BACnetLifeSafetyState enumeration
Multistate Input	State_Text property
Multistate Output	State_Text property
Multistate Value	State_Text property
Schedule	(varies)

H.6.1.7 Overridden

This boolean attribute may correspond to the OVERRIDDEN flag in the BACnet property StatusFlags. If the OVERRIDDEN flag of that property is set to true, then Overridden shall be true.

H.7 Virtual MAC Addressing

H.7.1 General

With the exception of LonTalk, a data link layer with a MAC address size greater than 6 octets shall expose a BACnet Virtual MAC (VMAC) address of 6 octets or fewer to the BACnet network layer.

The VMAC address shall function analogously as the MAC address of the technologies of Clauses 7, 8, 9, 11, and Annx J.

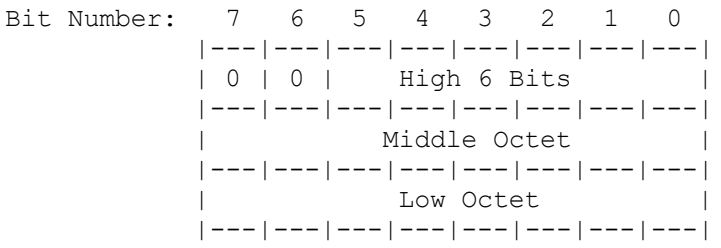
A VMAC table shall exist within the data link layer on all BACnet nodes on a BACnet network that employs VMAC addresses. A VMAC table shall be used to map native MAC addresses of the data link layer to VMAC addresses. The VMAC table contains VMAC entries corresponding to nodes in the BACnet network.

The data link layer uses native MAC addresses when communicating over its data link. The data link translates from VMAC addresses to native MAC addresses when BACnet messages are sent out. The data link translates from native MAC addresses to VMAC addresses when BACnet messages are received. If the address translation fails, the NPDU shall be dropped.

The methods used to maintain a VMAC table are dependent on the specific data link that is using a VMAC table.

H.7.2 Using Device Instance as a VMAC Address

When a particular data link layer specifies that each node's BACnet device instance is to be used as the VMAC address for the node, then the device instance as a VMAC address shall be transmitted as 3 octets, with the high order octet first, and formatted as follows:

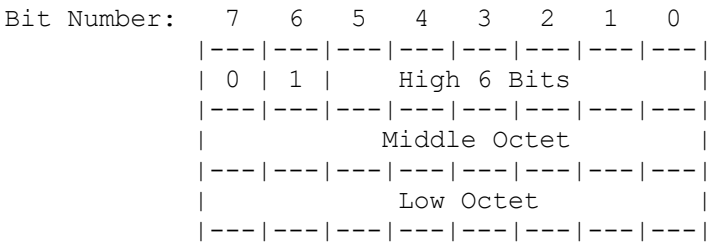


Nodes that do not have a BACnet device instance configured shall generate and use a random instance VMAC address. The generation and use of a random instance VMAC address does not affect the BACnet device instance which remains not configured. To ensure that the random instance VMAC is not used by another node, the node shall attempt to resolve

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

the generated VMAC in the network. If the node detects that another node is already using the random instance VMAC it has generated, it shall generate another random instance VMAC address. Once a node obtains a BACnet device instance, the node shall cease using the random VMAC and shall start using the regular device instance VMAC as described above.

The random portion of a random instance VMAC address is a number in the range 0 to 4194303. The resulting random instance VMAC address is in the range 4194304 to 8388607 (X'100000' to X'7FFFFF '). The generation of a random instance VMAC shall yield any number in the entire range with equal probability. A random instance VMAC is formatted as follows:



H.7.3 EUI-48 and Random-48 VMAC Address

When a particular data link layer specifies that a EUI-48 VMAC is to be used, then the device shall use a 6-octet VMAC in the form of an IEEE EUI-48 identifier. The means of obtaining or generating the EUI-48 identifier is a local matter. For example, if a device has a physical Ethernet adapter, and there is only one BACnet device hosted by that adapter on a particular BACnet network, then the Ethernet hardware address would be an appropriate choice for the initial value of the VMAC.

The Random-48 VMAC is a 6-octet VMAC address in which the least significant 4 bits (Bit 3 to Bit 0) in the first octet shall be B'0010' (X'2'), and all other 44 bits are randomly selected to be 0 or 1. The generation of a Random-48 VMAC shall yield any Random-48 VMAC in the entire range with equal probability.

To ensure that the VMAC is not used by another device, the device shall attempt to resolve its own VMAC in the network. If the device detects that another device is already using this VMAC, the device shall generate a new Random-48 VMAC address and try again.

The values X'000000000000' and X'FFFFFFFFFFFF' are not valid as VMAC addresses for a device and can have other uses defined by the data link.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX I - COMMANDABLE PROPERTIES WITH MINIMUM ON AND OFF TIMES (INFORMATIVE)

(This annex is not part of this standard but is included for informative purposes only)

This annex is an example of the use of a commandable property with minimum on and off times.

Suppose that we have a binary output object with a 600-second (10-minute) Minimum_On_Time, a 1200-second (20-minute) Minimum_Off_Time, and NORMAL polarity. The Priority_Array is all NULL, and the Present_Value is INACTIVE (off) due to the Relinquish_Default and has been INACTIVE for several hours. See Figure I-1(a).

In Figure I-1(b) a write is made to the Present_Value at priority 9 with a value of ACTIVE (on). Logic internal to the request server writes the value to entry 9 in the Priority_Array. Since the Present_Value has been off for more than 20 minutes, the change of state can occur immediately. Thus, the Present_Value will take the value ACTIVE and the Change_Of_State_Time will be set to the time of control.

Because the state of the Present_Value has changed, the minimum on and off time maintenance entity writes the new state of ACTIVE to entry 6 in the Priority_Array in order to enforce the minimum time. Any further writes to the Present_Value at priorities numerically greater than 6 (less important) will be entered into the Priority_Array but will not be acted upon due to the presence of the ACTIVE at priority 6. For example, in Figure I-1(c), a write to Present Value at priority 7 with a value of INACTIVE will be entered into the Priority_Array but will not be acted upon.

Writes to priorities numerically less than 6 will be entered into the Priority_Array and may cause changes of state due to their higher priority. This is desired for emergency and fire control.

In Figure I-1(e) the minimum on and off time maintenance entity in the device issues a relinquish (NULL) at priority 6 when the Minimum_On_Time expires, 10 minutes after the initial write. The remaining entries in the Priority_Array are then examined to determine the new Present_Value. If no change of state results, then no further action occurs. This would be the case in our example if no INACTIVE requests at priorities numerically less than 9 had been made.

In our example we had an INACTIVE at priority 7. Thus, in Figure I-1(f), when the Minimum_On_Time expires and the ACTIVE at priority 6 is relinquished, the value at priority 7 takes control. The Present_Value changes state to INACTIVE. As before, because the state of the Present_Value has changed, the minimum on and off time maintenance entity writes the new state of INACTIVE to entry 6 in the Priority_Array in order to enforce the minimum time.

We note that while write to priorities numerically less than 6 are not subject to minimum on and off times, if such writes cause changes of states the server should still write the new value to priority 6. Thus, if the higher priority request is relinquished within the minimum time, the minimum will be enforced before any lower priority requests can cause changes of state.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

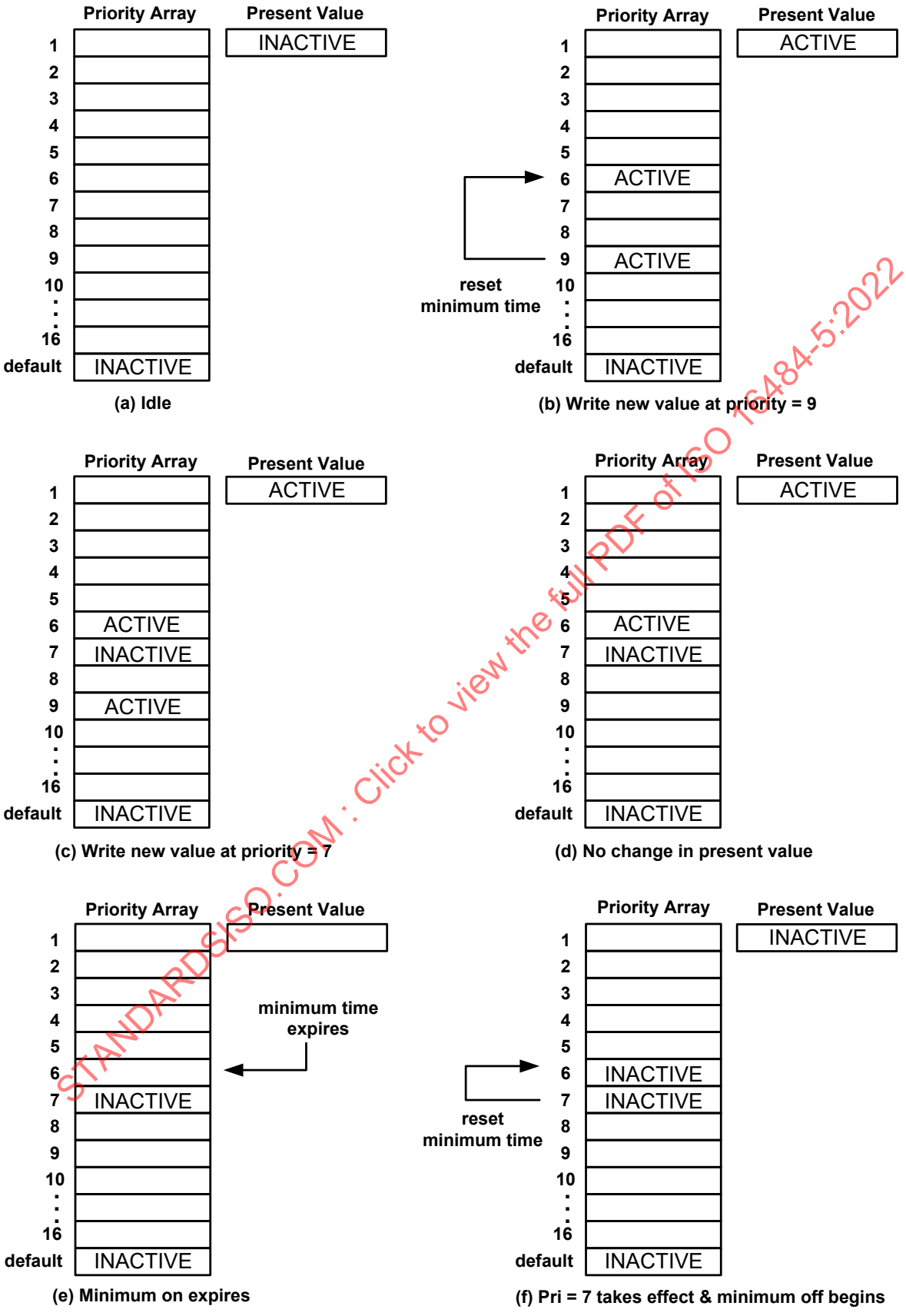


Figure I-1. Example of minimum on and off times.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX J - BACnet/IP (NORMATIVE)

(This annex is part of this standard and is required for its use.)

J.1 General

This normative annex specifies the use of BACnet messaging with the networking protocols originally defined as the result of research sponsored by the U. S. government's Defense Advanced Research Projects Agency and now maintained by the Internet Engineering Task Force. This suite of protocols is generally known as the "Internet Protocols."

J.1.1 BACnet/IP (B/IP) Network Definition

A BACnet/IP network is a collection of one or more IP subnetworks (IP domains) that are assigned a single BACnet network number. A BACnet internetwork consists of two or more BACnet networks. These networks may be BACnet/IP networks or use the technologies specified in Clauses 7, 8, 9, 11, H.2, Annex O, and Annex U. This standard also supports the inclusion of IP multicast groups in a fashion analogous to IP subnets, as described below in Clause J.8.

J.1.2 Addressing within B/IP Networks

In the case of B/IP networks, six octets consisting of the four-octet IP address followed by a two-octet UDP port number (both of which shall be transmitted most significant octet first) shall function analogously to the MAC address of the technologies of Clauses 7, 8, 9, and 11 with respect to communication between individual devices and inclusion in the Clause 6 NPCI, where a DADR or SADR is required. This address shall be referred to as a B/IP address. The default UDP port for both directed messages and broadcasts shall be X'BAC0' and all B/IP devices shall support it. In some cases, e.g., a situation where it is desirable for two groups of BACnet devices to coexist independently on the same IP subnet, the UDP port may be configured locally to a different value without it being considered a violation of this protocol. Where the "B/IP broadcast address" is referred to in this Annex, it means an IP address with the subnet of the broadcasting device in the network portion and all 1's in the host portion of the address and the UDP port of the devices on the B/IP network in question. An IP multicast address in conjunction with an appropriate UDP port may be used in lieu of the B/IP broadcast address under the circumstances defined in Clause J.8.

B/IP devices shall support configurable IP addresses and each shall be able to be set to any valid unicast IP address. B/IP devices shall also support a configurable UDP port number and shall support, at a minimum, values in the ranges 47808 - 47823 and 49152 - 65535. For B/IP devices that support multiple B/IP ports, the UDP port number for each B/IP port shall be settable across the above noted valid range.

J.1.2.1 Domain Names within B/IP Networks

B/IP devices shall be capable of resolving Internet host names to IP addresses via the use of the Domain Name Service (see RFC 1123) in the case where a DNS resolver is available. When a DNS resolver is available, then the IP address obtained via DNS name resolution shall be used in the IP address portion of B/IP addresses. The dotted-decimal form (“#.#.#.#”) may also be used in the DNS name. The DNS name format shall be checked for dotted-decimal format prior to attempting name resolution.

B/IP devices shall be capable of using IP addresses in place of DNS host names when a DNS resolver is not available.

For example, a B/IP device might specify bbmd.example.ashrae.org as the BBMD with which to register as a foreign device. In this case, the B/IP device will attempt to resolve bbmd.example.ashrae.org with a DNS resolver.

The frequency that a B/IP device resolves DNS host names is a local matter.

J.1.3 B/IP Concept

A BACnet/IP network shall function in concept identically to the other non-IP network types with respect to directed messages and broadcast messages, including local, remote, and global broadcasts, as defined in Clause 6.3.2: a directed message shall be sent directly to the destination node; a "local broadcast" shall reach all nodes on a single B/IP network; a "remote broadcast" shall reach all nodes on a single BACnet network with network number different from that of the originator's network; a "global broadcast" shall reach all nodes on all networks. The management of broadcasts within a single B/IP network, or between multiple B/IP networks, or between B/IP and non-B/IP networks, is described in Clause J.4.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

J.2 BACnet Virtual Link Layer

The BACnet Virtual Link Layer (BVLL) provides the interface between the BACnet Network Layer (Clause 6) and the underlying capabilities of a particular communication subsystem. This Annex specifies the BACnet Virtual Link Control (BVLC) functions required to support BACnet/IP directed and broadcast messages. The purpose and format of each message is described in the following clauses.

Note that each BVLL message has at least three fields. The 1-octet BVLC Type field indicates which underlying communication subsystem or microprotocol is in use. In this case, a BVLC Type of X'81' indicates the use of BACnet/IP as defined in this Annex. The 1-octet BVLC Function field identifies the specific function to be carried out in support of the indicated communication subsystem or microprotocol type. The 2-octet BVLC Length field is the length, in octets, of the entire BVLL message, including the two octets of the length field itself, most significant octet first.

J.2.1 BVLC-Result: Purpose

This message provides a mechanism to acknowledge the result of those BVLL service requests that require an acknowledgment, whether successful (ACK) or unsuccessful (NAK). These are: Write-Broadcast-Distribution-Table (ACK, NAK); Read-Broadcast-Distribution-Table (NAK only); Register-Foreign-Device (ACK, NAK); Read-Foreign-Device-Table (NAK only); Delete-Foreign-Device-Table-Entry (ACK, NAK); and Distribute-Broadcast-To-Network (NAK only).

All non-BBMD B/IP devices shall return the appropriate NAK upon receipt of a unicast BVLL service request intended for BBMDs only (Write-Broadcast-Distribution-Table; Read-Broadcast-Distribution-Table; Register-Foreign-Device; Read-Foreign-Device-Table; Delete-Foreign-Device-Table-Entry; and Distribute-Broadcast-To-Network).

J.2.1.1 BVLC-Result: Format

The BVLC-Result message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'00'	BVLC-Result
BVLC Length:	2-octets	X'0006'	Length, in octets, of the BVLL message
Result Code:	2-octets	X'0000'	Successful completion
		X'0010	Write-Broadcast-Distribution-Table NAK
		X'0020'	Read-Broadcast-Distribution-Table NAK
		X'0030'	Register-Foreign-Device NAK
		X'0040'	Read-Foreign-Device-Table NAK
		X'0050'	Delete-Foreign-Device-Table-Entry NAK
		X'0060'	Distribute-Broadcast-To-Network NAK

J.2.2 Write-Broadcast-Distribution-Table: Purpose

This message provides a mechanism for initializing or updating a Broadcast Distribution Table (BDT) in a BACnet Broadcast Management Device (BBMD).

J.2.2.1 Write-Broadcast-Distribution-Table: Format

The Write-Broadcast-Distribution-Table message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'01'	Write-Broadcast-Distribution-Table
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
List of BDT Entries:	N*10-octets		

N indicates the number of entries in the BDT. Each BDT entry consists of the 6-octet B/IP address of a BBMD followed by a 4-octet field called the broadcast distribution mask that indicates how broadcast messages are to be distributed on the IP subnet served by the BBMD. See Clause J.4.3.2.

Prior to the introduction of the Network Port object in Protocol_Revision 17, this message was the interoperable means of updating BDTs. That function is now performed by writes to the Network Port object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

J.2.3 Read-Broadcast-Distribution-Table: Purpose

The message provides a mechanism for retrieving the contents of a BBMD's BDT.

J.2.3.1 Read-Broadcast-Distribution-Table: Format

The Read-Broadcast-Distribution-Table message consists of three fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'02'	Read-Broadcast-Distribution-Table
BVLC Length:	2-octets	X'0004'	Length, in octets, of the BVLL message

J.2.4 Read-Broadcast-Distribution-Table-Ack: Purpose

This message returns the current contents of a BBMD's BDT to the requester. An empty BDT shall be signified by a list of length zero.

J.2.4.1 Read-Broadcast-Distribution-Table-Ack: Format

The Read-Broadcast-Distribution-Table-Ack message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'03'	Read-Broadcast-Distribution-Table-Ack
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
List of BDT Entries:	N*10-octets		

N indicates the number of entries in the BDT whose contents are being returned.

J.2.5 Forwarded-NPDU: Purpose

This BVLL message is used in broadcast messages from a BBMD as well as in messages forwarded to registered foreign devices. It contains the source address of the original node, or if NAT is being used, the address with which the original node is accessed, as well as the original BACnet NPDU.

Upon receipt of a Forwarded-NPDU with a B/IP Address of Originating Device field whose B/IP address is different from the B/IP address of the sending node, the receiving node shall utilize the contents of that field as the source B/IP address of the sending node.

J.2.5.1 Forwarded-NPDU: Format

The Forwarded-NPDU message consists of five fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'04'	Forwarded-NPDU
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
B/IP Address of Originating Device:			6-octets
BACnet NPDU from Originating Device:			Variable length

J.2.6 Register-Foreign-Device: Purpose

This message allows a foreign device, as defined in Clause J.5.1, to register with a BBMD for the purpose of receiving broadcast messages.

J.2.6.1 Register-Foreign-Device: Format

The Register-Foreign-Device message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'05'	Register-Foreign-Device
BVLC Length:	2-octets	X'0006'	Length, in octets, of the BVLL message
Time-to-Live	2-octets	T	Time-to-Live T, in seconds

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The Time-to-Live value is the number of seconds within which a foreign device must re-register with a BBMD or risk having its entry purged from the BBMD's FDT. This value will be sent most significant octet first. See Clause J.5.2.2.

J.2.7 Read-Foreign-Device-Table: Purpose

The message provides a mechanism for retrieving the contents of a BBMD's FDT.

J.2.7.1 Read-Foreign-Device-Table: Format

The Read-Foreign-Device-Table message consists of three fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'06'	Read-Foreign-Device-Table
BVLC Length:	2-octets	X'0004'	Length, in octets, of the BVLL message

J.2.8 Read-Foreign-Device-Table-Ack: Purpose

This message returns the current contents of a BBMD's FDT to the requester. An empty FDT shall be signified by a list of length zero.

J.2.8.1 Read-Foreign-Device-Table-Ack: Format

The Read-Foreign-Device-Table-Ack message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'07'	Read-Foreign-Device-Table-Ack
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
List of FDT Entries:	N*10-octets		

N indicates the number of entries in the FDT whose contents are being returned. Each returned entry consists of the 6-octet B/IP address of the registrant; the 2-octet Time-to-Live value supplied at the time of registration; and a 2-octet value representing the number of seconds remaining before the BBMD will purge the registrant's FDT entry if no re-registration occurs. The time remaining includes the 30-second grace period as defined in Clause J.5.2.3.

J.2.9 Delete-Foreign-Device-Table-Entry: Purpose

This message is used to delete an entry from the Foreign-Device-Table.

J.2.9.1 Delete-Foreign-Device-Table-Entry: Format

The Delete-Foreign-Device-Table-Entry message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'08'	Delete-Foreign-Device-Table-Entry
BVLC Length:	2-octets	X'000A'	Length, in octets, of the BVLL message
FDT Entry:	6-octets		

The FDT entry is the B/IP address of the table entry to be deleted.

J.2.10 Distribute-Broadcast-To-Network: Purpose

This message provides a mechanism for a foreign device to request to BBMD to distribute a Forwarded-NPDU BVLC to the local IP subnet, to all BBMD's configured in the BBMD's BDT, and to all foreign devices in the BBMD's FDT except the originating node.

J.2.10.1 Distribute-Broadcast-To-Network: Format

The Distribute-Broadcast-To-Network message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'09'	Distribute-Broadcast-To-Network
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
BACnet NPDU from Originating Device:			Variable length

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

J.2.11 Original-Unicast-NPDU: Purpose

This message is used to send directed NPDUs to another B/IP device or router.

J.2.11.1 Original-Unicast-NPDU: Format

The Original-Unicast-NPDU message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'0A'	Original-Unicast-NPDU
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
BACnet NPDU:	Variable length		

J.2.12 Original-Broadcast-NPDU: Purpose

This message is used by B/IP devices and routers which are not foreign devices to broadcast NPDUs on a B/IP network.

J.2.12.1 Original-Broadcast-NPDU: Format

The Original-Broadcast-NPDU message consists of four fields:

BVLC Type:	1-octet	X'81'	BVLL for BACnet/IP
BVLC Function:	1-octet	X'0B'	Original-Broadcast-NPDU
BVLC Length:	2-octets	L	Length L, in octets, of the BVLL message
BACnet NPDU:	Variable length		

J.2.13 Deleted Clause

This clause has been removed.

J.3 BACnet/IP Directed Messages

B/IP devices shall communicate directly with each other by using the B/IP address of the recipient. Each NPDU shall be transmitted in a BVLL Original-Unicast-NPDU.

J.4 BACnet/IP Broadcast Messages

This clause defines how BACnet broadcast messages are managed within a B/IP network.

J.4.1 B/IP Broadcast Management, Single IP Subnet

In this case, the B/IP network consists of a single IP subnet. A "local broadcast" shall use the B/IP broadcast address and the NPDU shall be transmitted in a BVLL Original-Broadcast-NPDU message. Because all nodes are on a single IP subnet, such messages will automatically reach all nodes. See Figure J-1.

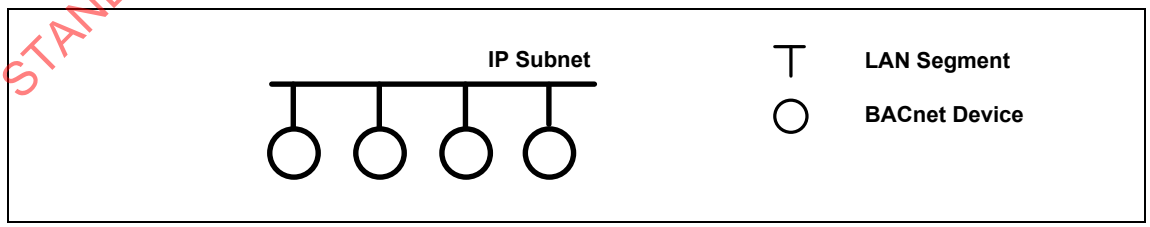


Figure J-1. A B/IP network consisting of a single IP subnet.

J.4.2 B/IP Broadcast Management, Multiple IP Subnets

In this case, the BACnet/IP network consists of two or more IP subnets. A "local broadcast" shall use the B/IP broadcast address, and the NPDU shall be transmitted in a BVLL Original-Broadcast-NPDU message. Because standard IP routers do not forward such broadcasts, an ancillary device is required to perform this function. This device shall be called a BACnet/IP Broadcast Management Device (BBMD). See Figure J-2.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

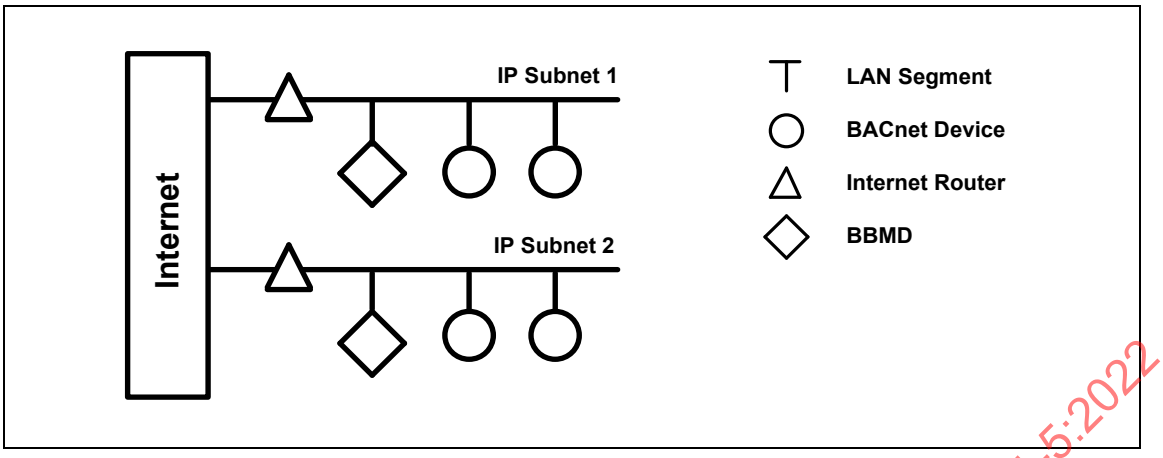


Figure J-2. A B/IP network consisting of two IP subnets.

J.4.3 BBMD Concept

Each IP subnet that contains B/IP devices that do not register as a foreign device and that are part of a B/IP network comprised of two or more IP subnets shall have at least one BBMD. Each BBMD shall possess a table called a Broadcast Distribution Table (BDT). If there are two or more BBMDs on a single subnet, their BDTs shall not contain any common entries in order to avoid a broadcast forwarding loop. The BDT determines which remote IP subnets receive forwarded BACnet broadcasts. To reduce BACnet broadcast traffic, it is possible to configure the BDT to forward broadcasts only to IP subnets where they are required. A BBMD shall be able to be configured to accept Foreign Device registrations, shall support the two-hop broadcast distribution method, and shall support the execution of all FDT read and write messages defined in Clause J.2, and shall support execution of the Read-Broadcast-Distribution-Table message as defined in Clause J.2. Support for the one-hop broadcast distribution method is optional.

A B/IP device which cannot be configured as a BBMD shall be capable of registering as a foreign device with a BBMD.

J.4.3.1 Broadcast Distribution

There are two ways that a BBMD may distribute broadcast messages to remote IP subnets. The first is to use IP "directed broadcasts" (also called "one-hop" distribution). This involves sending the message using a B/IP address in which the network portion of the address contains the subnet of the destination IP subnet and the host portion of the address contains all 1's. While this method of distribution is efficient, it requires that the IP router serving the destination subnet be configured to support the passage of such directed broadcasts.

Since not all IP routers are configured to pass directed broadcasts, a BBMD may be configured to send a directed message to the BBMD on the remote subnet ("two-hop" distribution) which then transmits it using the B/IP broadcast address. Since the use of one-hop distribution requires an IP router configuration that may or may not be possible, while the two-hop method is always available, the choice of which method to use in any given case is a local matter.

J.4.3.2 Broadcast Distribution Table Format

The BDT consists of one entry for the address of the BBMD for the local IP subnet and an entry for the BBMD on each remote IP subnet to which broadcasts are to be forwarded. Each entry consists of the 6-octet B/IP address with which the BBMD is accessed and a 4-octet broadcast distribution mask. If the IP router to the subnet performs network address translation (NAT), then the BDT entry shall contain the global IP address of the IP router. The operation of BBMDs in the presence of NAT is described in Clause J.7.5. If messages are to be distributed on the remote IP subnet using directed broadcasts, the broadcast distribution mask shall be identical to the subnet mask associated with the subnet, i.e. all 1's in the network portion of the 4-octet IP address field and all 0's in the host portion. If messages are to be distributed on the remote IP subnet by sending the message directly to the remote BBMD, the broadcast distribution mask shall be all 1's. The broadcast distribution masks referring to the same IP subnet shall be identical in each BDT. The use of the broadcast distribution mask is described in Clause J.4.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

J.4.4 BBMD Configuration

The configuration of the BACnet-related capability of a BBMD shall consist of supplying it with a BDT. The table shall be supplied by writing to the BBMD_Broadcast_Distribution_Table property of the Network Port Object which represents this B/IP port.

Two BVLL messages support the maintenance of BDTs and are described in Clauses J.4.4.1 and J.4.4.2.

J.4.4.1 Use of the BVLL Read-Broadcast-Distribution-Table Message

Upon receipt of a BVLL Read-Broadcast-Distribution-Table message, a BBMD shall load the contents of its BDT into a BVLL Read-Broadcast-Distribution-Table-Ack message and send it to the originating device. If a table entry contains a host name, then the corresponding entry in the Read-Broadcast-Distribution-Table-Ack message shall contain the B/IP address of the resolved host name or X'000000000000' to indicate that the host name has not been resolved. If the BBMD is unable to perform the read of its BDT, it shall return a BVLC-Result message to the originating device with a result code of X'0020' indicating that the read attempt has failed. A B/IP device which is not configured as a BBMD shall always return a BVLC-Result message to the originating device with a result code of X'0020' indicating that the Read-Broadcast-Distribution-Table BVLL message is not supported.

J.4.4.2 Use of the BVLL Write-Broadcast-Distribution-Table Message

Upon receipt of a BVLL Write-Broadcast-Distribution-Table message, B/IP devices shall always return a BVLC-Result message to the originating device with a result code of X'0010' indicating that the Write-Broadcast-Distribution BVLL message is not supported.

J.4.5 BBMD Operation - Broadcast Distribution

Upon receipt of a BVLL Original-Broadcast-NPDU message, a BBMD shall construct a BVLL Forwarded-NPDU message and send it to each IP subnet in its BDT with the exception of its own. The B/IP address to which the Forwarded-NPDU message is sent is formed by inverting the broadcast distribution mask in the BDT entry and logically ORing it with the BBMD address of the same entry. This process produces either the directed broadcast address of the remote subnet or the unicast address of the BBMD on that subnet depending on the contents of the broadcast distribution mask. See Clause J.4.3.2.. In addition, the received BACnet NPDU shall be sent directly to each foreign device currently in the BBMD's FDT also using the BVLL Forwarded-NPDU message.

Upon receipt of a BVLL Forwarded-NPDU message, a BBMD shall process it according to whether it was received from a peer BBMD as the result of a directed broadcast or a unicast transmission. A BBMD may ascertain the method by which Forwarded-NPDU messages will arrive by inspecting the broadcast distribution mask field in its own BDT entry since masks referring to the same IP address are required to be identical in all BBMDs. If the message arrived via directed broadcast, or if the source is a device located on the same IP subnet, a situation which can occur if two or more BBMDs are installed on the same IP subnet, it was also received by the other devices on the BBMD's subnet. In this case the BBMD merely retransmits the message directly to each foreign device currently in the BBMD's FDT. Otherwise the message arrived via a unicast transmission and it has not yet been received by the other devices on the BBMD's subnet. In this case, the message is sent to the devices on the BBMD's subnet using the B/IP broadcast address as well as to each foreign device currently in the BBMD's FDT. A BBMD on a subnet with no other BACnet devices (such as a NAT-supporting BBMD, see Clause J.7.5) may omit the broadcast using the B/IP broadcast address. The method by which a BBMD determines whether or not other BACnet devices are present is a local matter.

Upon receipt of a BVLL Distribute-Broadcast-To-Network message from a registered foreign device, the receiving BBMD shall transmit a BVLL Forwarded-NPDU message on its local IP subnet using the local B/IP broadcast address as the destination address. In addition, a Forwarded-NPDU message shall be sent to each entry in its BDT as described above in the case of the receipt of a BVLL Original-Broadcast-NPDU as well as directly to each foreign device currently in the BBMD's FDT except the originating node. If the BBMD is unable to perform the forwarding function, or the message was not received from a registered foreign device, then it shall return a BVLC-Result message to the foreign device with a result code of X'0060' indicating that the forwarding attempt was unsuccessful.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

J.5 Addition of Foreign B/IP Devices to an Existing B/IP Network

J.5.1 Foreign Device Definition

A "foreign" device is a BACnet device that has an IP subnet address different from those subnets comprising the BACnet/IP network that the device seeks to join. The foreign device may be a full-time node on the foreign subnet or may be a part-time participant. See Figure J-3.

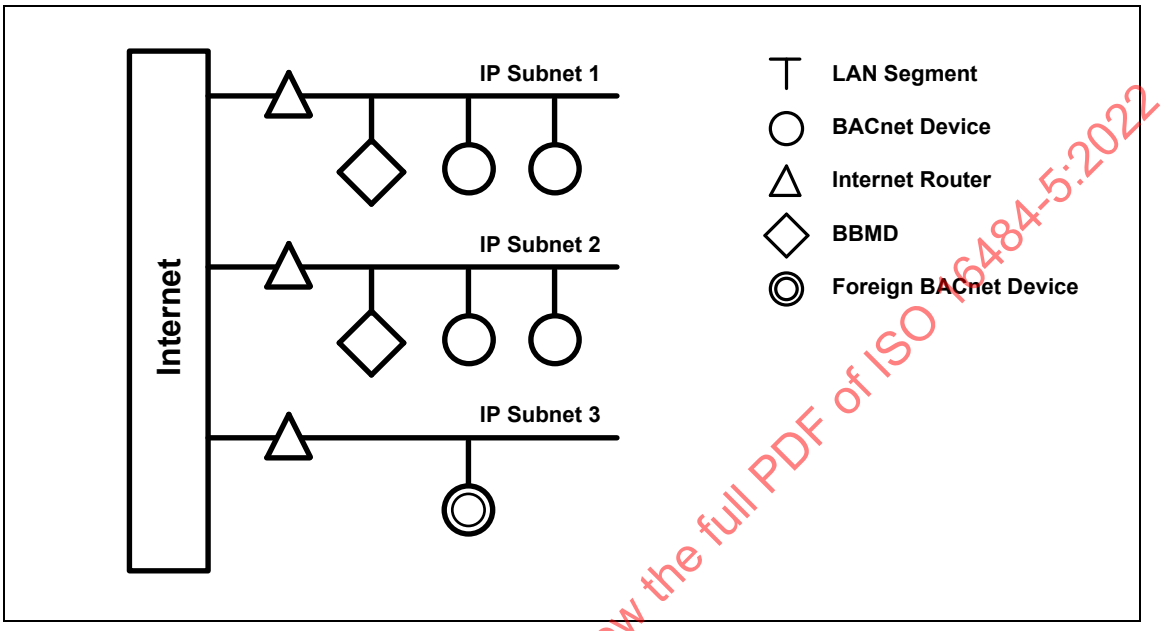


Figure J-3. The "foreign" BACnet device on Subnet 3 can register to receive broadcasts from devices on Subnets 1 and 2 by sending a BVLL Register-Foreign-Device message to a BBMD that supports foreign device registration.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

J.5.2 BBMD Operation - Foreign Devices

In order for a foreign device to fully participate in the activities of a B/IP network, the device must register itself with a BBMD serving one of the IP subnets comprising that network. "Full participation" implies the ability to send and receive both directed and broadcast messages. Registration consists of sending a BVLL Register-Foreign-Device message to an appropriate BBMD and receiving a BVLC-Result message containing a result code of X'0000' indicating the successful completion of the registration. Ascertaining the IP address of such a BBMD is a local matter but could involve the use of a domain nameserver or the distribution of a numeric IP address to authorized users. The UDP port X'BAC0' shall be considered the default, but the use of other port values is permitted if required by the local network architecture, e.g., where two B/IP networks share the same physical LAN.

J.5.2.1 Foreign Device Table

Each device that registers as a foreign device shall be placed in an entry in the BBMD's Foreign Device Table (FDT). Each entry shall consist of the 6-octet B/IP address of the registrant; the 2-octet Time-to-Live value supplied at the time of registration; and a 2-octet value representing the number of seconds remaining before the BBMD will purge the registrant's FDT entry if no re-registration occurs. The number of seconds remaining shall be initialized to the 2-octet Time-to-Live value supplied at the time of registration plus 30 seconds (see Clause J.5.2.3), with a maximum of 65535.

Two BVLL messages support the maintenance of FDTs and are described in Clauses J.5.2.1.1 and J.5.2.1.2.

J.5.2.1.1 Use of the BVLL Read-Foreign-Device-Table Message

Upon receipt of a BVLL Read-Foreign-Device-Table message, a BBMD shall load the contents of its FDT into a BVLL Read-Foreign-Device-Table-Ack message and send it to the originating device. If the BBMD is unable to perform the read of its FDT, it shall return a BVLC-Result message to the originating device with a result code of X'0040' indicating that the read attempt has failed. A B/IP device which is not configured as a BBMD shall always return a BVLC-Result message to the

© ISO 2022 – All rights reserved

originating device with a result code of X'0040' indicating that the Read-Foreign-Device-Table BVLL message is not supported.

J.5.2.1.2 Use of the BVLL Delete-Foreign-Device-Table-Entry Message

Upon receipt of a BVLL Delete-Foreign-Device-Table-Entry message, a BBMD shall search its foreign device table for an entry corresponding to the B/IP address supplied in the message. If an entry is found, it shall be deleted and the BBMD shall return a BVLC-Result message to the originating device with a result code of X'0000'. Otherwise, the BBMD shall return a BVLC-Result message to the originating device with a result code of X'0050' indicating that the deletion attempt has failed. A B/IP device which is not configured as a BBMD shall always return a BVLC-Result message to the originating device with a result code of X'0050' indicating that the Delete-Foreign-Device-Table-Entry BVLL message is not supported.

J.5.2.2 Use of the BVLL Register-Foreign-Device Message

Upon receipt of a BVLL Register-Foreign-Device message, a BBMD configured to accept foreign device registrations and having available table entries, shall add an entry to its FDT as described in Clause J.5.2.1 and reply with a BVLC-Result message containing a result code of 'Successful completion' indicating the successful completion of the registration. A BBMD that does not have an available table entry or that is not configured to accept foreign device registrations shall return a BVLC-Result message containing a result code of 'Register-Foreign-Device NAK' indicating that the registration has failed. A B/IP device which is not configured as a BBMD shall always return a BVLC-Result message containing a result code of 'Register-Foreign-Device NAK' indicating that the Register-Foreign-Device BVLL message is not supported.

J.5.2.3 Foreign Device Table Timer Operation

Upon receipt of a BVLL Register-Foreign-Device message, a BBMD shall start a timer with a value equal to the Time-to-Live parameter supplied plus a fixed grace period of 30 seconds. If, within the period during which the timer is active, another BVLL Register-Foreign-Device message from the same device is received, the timer shall be reset and restarted. If the time expires without the receipt of another BVLL Register-Foreign-Device message from the same foreign device, the FDT entry for this device shall be cleared.

J.6 Routing Between B/IP and non-B/IP BACnet Networks

J.6.1 Router Operation

In concept, a router between a B/IP network and a non-B/IP network functions identically to the routers described in Clause 6. See Figure J-4.

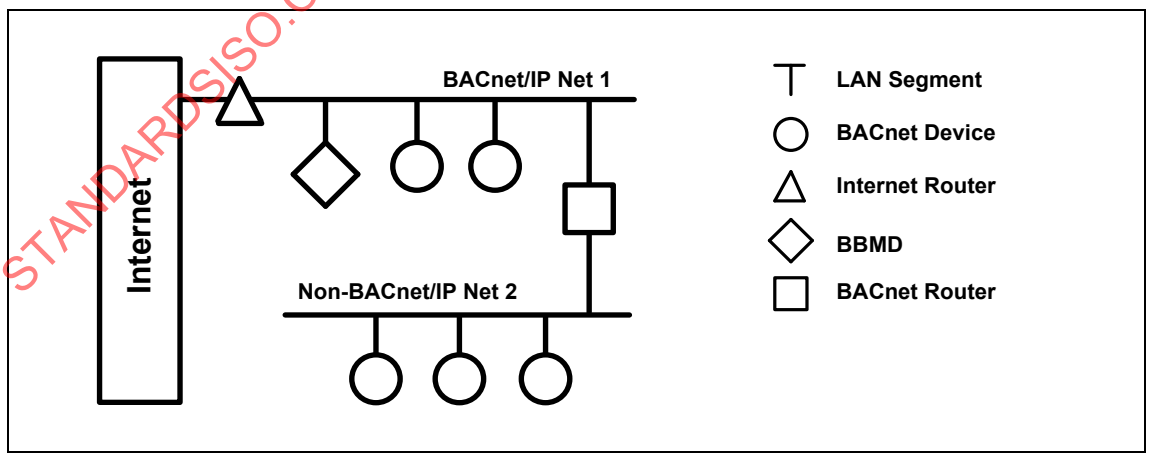


Figure J-4. A BACnet router can be used to convey messages between devices on a B/IP network and non-B/IP network using the procedures in Clause 6.

There are two possible differences. First, on the B/IP side, the B/IP address is used in place of the MAC layer address referred to throughout Clause 6. Second, if B/IP and non-B/IP BACnet devices reside on the same physical LAN, then all traffic is typically sent and received through a single physical port. The collection of B/IP devices would, in such a case,

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

have a network number distinct from the network number of the non-B/IP devices. Such a scenario could easily occur on an Ethernet network where some devices are IP-capable while others are not.

J.7 Routing Between Two B/IP BACnet Networks

Although the foreign registration process provides the ability for remote devices to participate in a particular B/IP network, there may be occasions when it is desirable for two collections of B/IP devices to interoperate more closely. This type of interoperation can only produce results consistent with the assumptions and intent contained in the original BACnet standard if the configuration of the two B/IP networks has been coordinated. For example, it is assumed that Device object identifiers are unique "internetwork wide." If this is not the case, the Who-Is service will produce ambiguous results. Similarly, the Who-Has service may be useless for dynamic configuration applications if multiple instances of objects with identical object identifiers exist.

The BACnet standard also assumes that only a single path exists between devices on different BACnet networks and that this path passes through a BACnet router. The Internet's web topology violates this assumption in that, apart from security constraints such as "firewalls", any IP device can communicate directly with any other IP device if it knows the device's IP address.

This clause specifies how B/IP internetworks may be constructed.

J.7.1 B/IP Internetwork Design Considerations

This standard recognizes that BACnet internetworks containing one or more B/IP networks can be configured in a variety of ways, depending on the requirements of an installation. Any of these configurations shall be deemed to conform to this standard provided they employ the techniques specified in this clause.

1) Depending on local traffic conditions and security requirements, all B/IP subnets can be configured into a single B/IP network. This case is dealt with in Clauses J.1 to J.6.

2) Creating two or more B/IP networks, each with a unique network number, can be useful for limiting the propagation of local broadcast messages and for providing security by confining traffic to a particular geographic or logical area.

3) A single device can be configured to provide all the routing for a B/IP internetwork. See Figure J-5. The advantages include: only a single routing table is required; the possibility of creating multiple paths between B/IP networks is eliminated; the resulting star topology is easy to conceptualize. The disadvantages are: there is a single point of failure; a single device could present a traffic bottleneck under heavy load conditions.

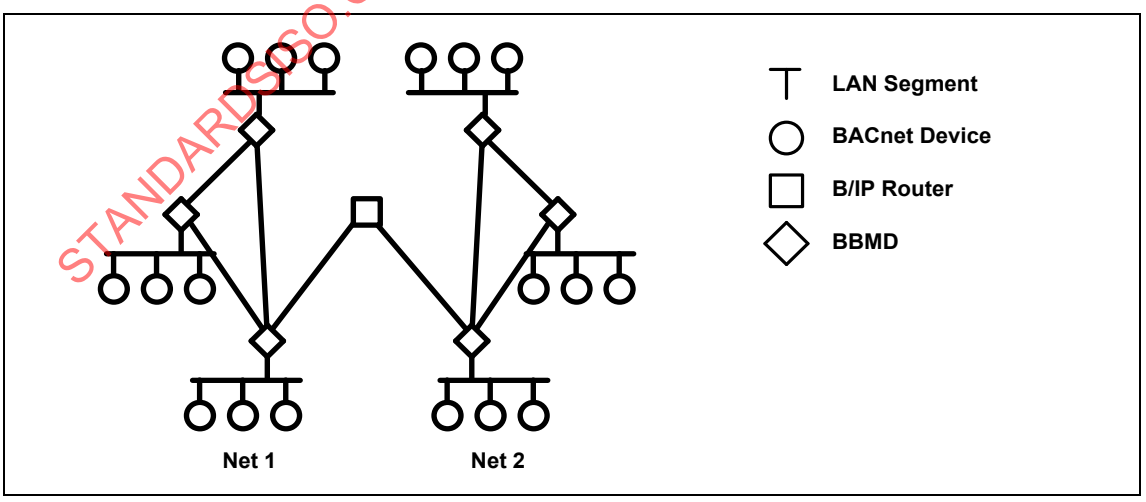


Figure J-5. A single B/IP Router can perform all routing for two or more B/IP networks by registering as a foreign device on each network. It is then "directly connected" to each such network and uses a UDP port unique to that network for the receipt of communications from its individual nodes. See Clause J.7.2. The unique UDP port is required to determine a message's origin for the purpose of appending an SNET to the routed packet. Note that the UDP port associated with the B/IP addresses of the non-router nodes remains in general X'BAC0'.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

4) While the functions of BBMDs as specified in Clause J.4 and of BACnet routers as specified in Clauses 6 and J.7.2 are entirely distinct, this standard does not preclude the implementation of BBMD functionality and router functionality in a single physical device. See Figure J-6.

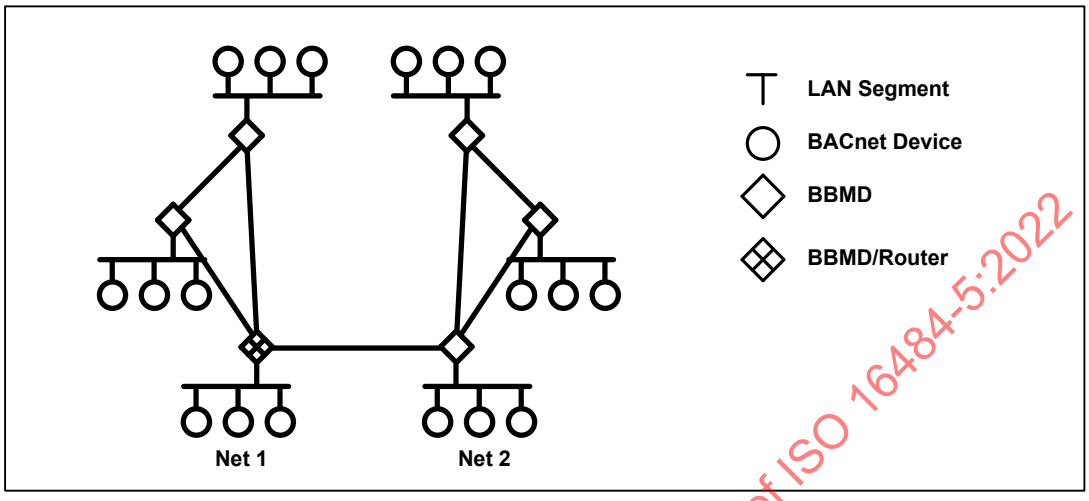


Figure J-6. An alternative to the architecture depicted in Figure J-5 is to combine both BBMD and router functionality in the same physical device as explicitly permitted in Clause J.7.1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

J.7.2 B/IP Routers

B/IP routers adhere to the requirements of Clause 6 with the following differences:

(a) The physical ports of Clause 6 routers are replaced by logical ports. Each logical port is identified by the unique B/IP address of the port's connection to the B/IP network served by the router.
(b) The term "directly connected network" in Clause 6 implies a physical LAN connection between a LAN segment and a physical router port. In this clause "directly connected network" is extended to mean any B/IP network from which a router can receive local broadcast or IP multicast messages. Such networks are: the B/IP network on which a router resides by virtue of having an IP network number in common with one of the IP subnets comprising the B/IP network; a B/IP network in which the router participates as a member of an IP multicast group; or a B/IP network in which a router participates by having registered as a foreign device with a BBMD serving that network.
(c) Networks that are not directly connected are called "remote networks." Remote networks, whether B/IP or non-B/IP, may be reachable by communicating using B/IP with a router serving the remote network.

J.7.3 B/IP Router Tables

B/IP router tables shall contain the following information for each logical port:

(a) the B/IP address for this port,
(b) if the port is to be used to communicate with nodes on a network directly connected by virtue of the router having an IP network number in common with one of the IP subnets comprising the B/IP network, the BACnet network number of the network, else

if the port is to be used to communicate with nodes on a network directly connected by virtue of the router registering as a foreign device with a BBMD, the BACnet network number of the network served by the BBMD and the B/IP address of the BBMD offering foreign device registration,
(c) a list of network numbers reachable through this port along with the B/IP address of the next router on the path to each network number and the reachability status, as defined in Clause 6.6.1, of each such network.

Because internetworks involving multiple B/IP networks may be more dynamic than traditional BACnet internetworks, implementors of B/IP routers may wish to provide a mechanism whereby specific table entries can be selectively activated and deactivated. The mechanism for accomplishing this is deemed to be a local matter.

© ISO 2022 – All rights reserved

J.7.4 B/IP Router Operation

Upon start-up, each B/IP router shall search its routing table for active entries indicating direct connection via foreign registration with a BBMD. The router shall then proceed to register itself with each such network using the procedures specified in Clause J.5. At the conclusion of all such registrations, the router shall follow the procedure of Clause 6.6.2 in that it shall broadcast an I-Am-Router-To-Network message containing the network numbers of each accessible network except the networks reachable via the network on which the broadcast is being made. Note that networks accessed through a given active UDP port that are not directly connected, but are reachable by means of communication with another B/IP router shall, upon router startup, be deemed to be reachable.

Additional router operations with regard to local and remote traffic shall follow the procedures of Clause 6.

J.7.5 BBMD Operation with Network Address Translation

Network Address Translation (NAT) is in widespread use by IP routers and firewalls to connect private subnets to the global Internet. Using NAT, multiple hosts on a subnet can access the Internet using a single public IP address. BBMD operation supporting NAT routers is optional for BBMD devices. A single BACnet device may contain several B/IP network ports, each with its own internal BBMD.

For those B/IP networks that communicate through a NAT router, there are several additional considerations:

(a) For any single B/IP network, only one device on the local side of the NAT router may be accessible from the global side. All other devices on the local side of that NAT router need to be on different BACnet networks so that they can be uniquely addressed using the BACnet network layer. The globally accessible device will contain a BACnet router to those networks. The globally accessible device may be either a BBMD or a foreign device.

(b) The NAT router at each subnet location should be configured to port forward B/IP messages to the BBMD. Port forwarding causes all messages directed to the specified port to be forwarded to a specific local address.

(c) To enable messages to traverse the Internet, the destination IP address and UDP port in all Forwarded-NPDU messages shall be the global IP address and UDP port of the destination subnet. This is facilitated by entering the global B/IP address of each BBMD in the BDT.

(d) Except when propagating a received Forwarded-NPDU message, the "B/IP Address of Originating Device" field in Forwarded-NPDU messages is the global IP address and port of the NAT Router through which the BBMD communicates. This is required so that responding devices on the remote subnet may communicate with the originating device. Received Forwarded-NPDU messages are propagated as-is to foreign devices and to the local IP subnet as defined in Clause J.4.5

(e) A foreign device behind a NAT router should register often with a BBMD to maintain a return path through the NAT router back to the foreign device. The maximum allowed time between registrations is dependent on the NAT router, and may be 30 seconds or less.

(f) Two-hop distribution shall be used in B/IP networks that contain NAT routers, since one-hop distribution is not possible through NAT routers.

A B/IP internetwork containing NAT Routers can be configured several ways. See Figures J-7 and J-8 for example configurations.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

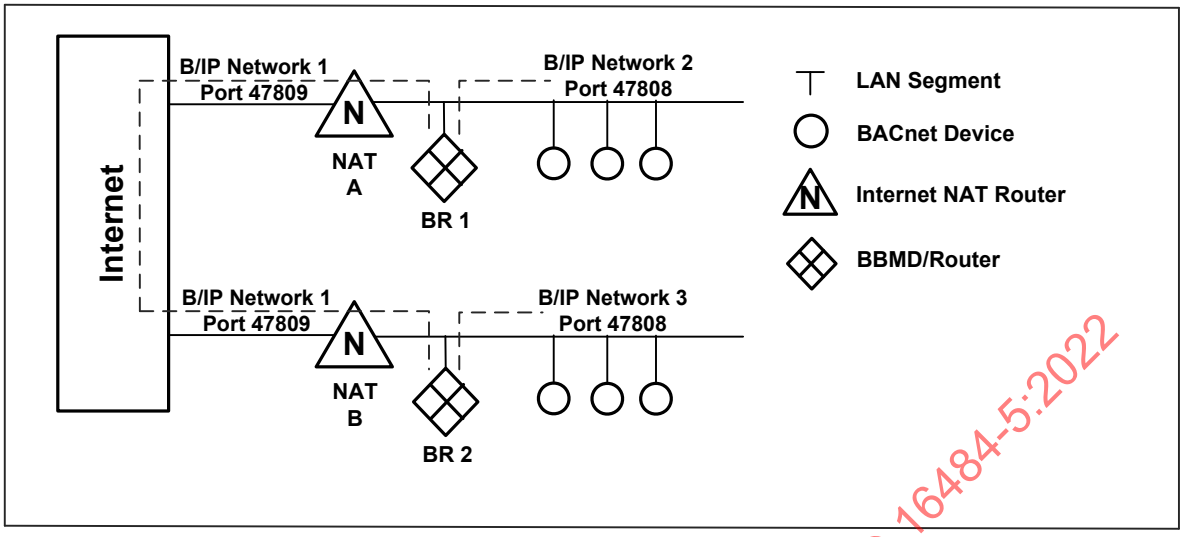


Figure J-7. This figure represents a B/IP internetwork that uses the Internet to connect two remote sites. The NAT devices translate global Internet IP/Port addresses into private addresses. Different networks behind NAT devices may use the same IP/Port address range as demonstrated here. Both B/IP network 2 and B/IP network 3 use locally-assigned IP addresses from the subnet 192.168.1.* and UDP port 47808.

NAT A Configuration
- Internet IP: 201.1.1.1
- Forward: 201.1.1.1:47809 ➔ 192.168.1.10:47809

NAT B Configuration
- Internet IP: 237.2.2.2
- Forward: 237.2.2.2:47809 ➔ 192.168.1.10:47809

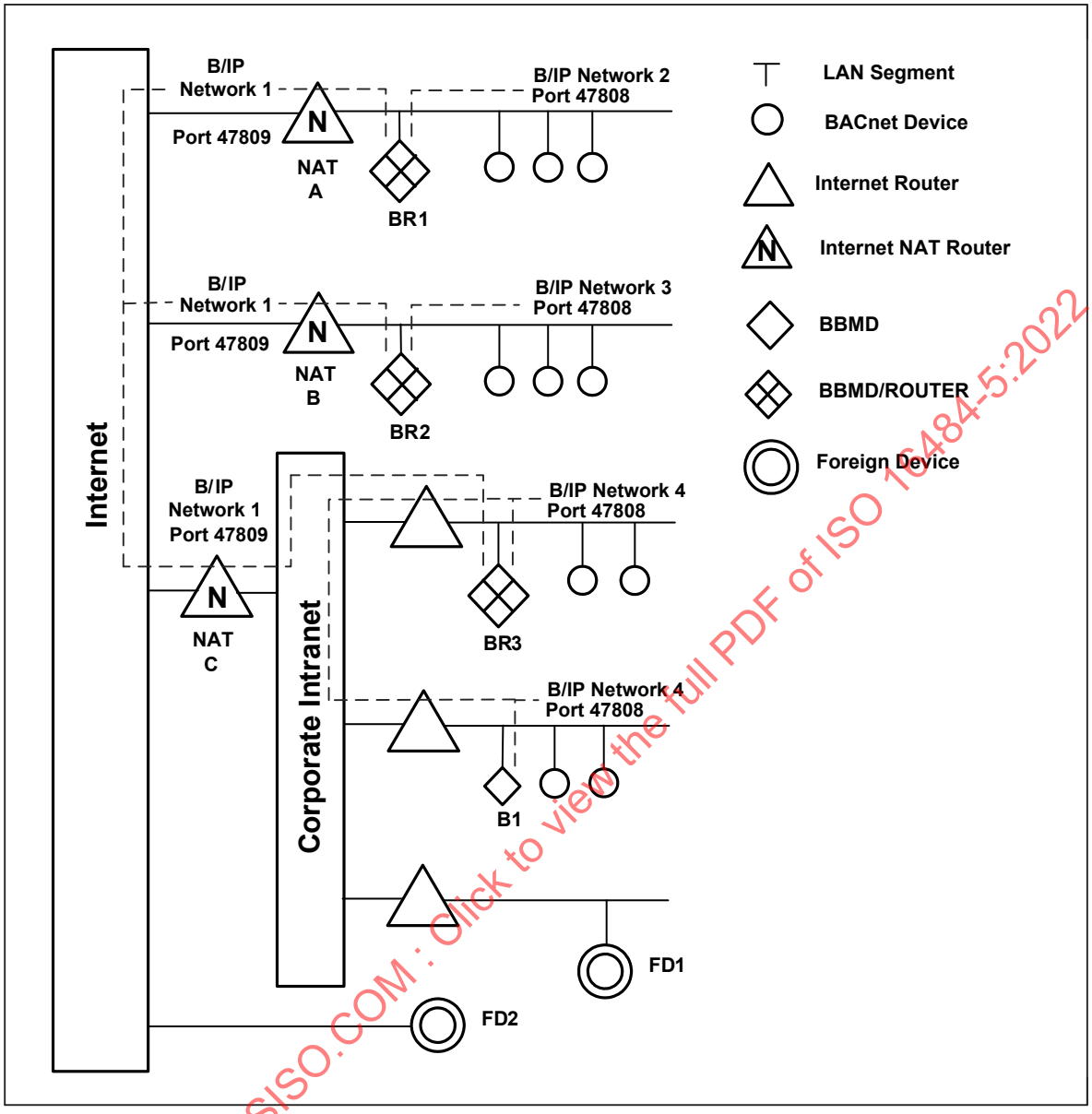
BR1 - BBMD/Router Configuration
- Global IP Address: 201.1.1.1:47809 (global B/IP address of NAT A)
- B/IP Address Net 1: 192.168.1.10:47809
- BDT Net 1: 201.1.1.1:47809 (global B/IP address of NAT A, self), 237.2.2.2:47809 (global B/IP address of NAT B)
- B/IP Address Net 2: 192.168.1.10:47808
- BDT Net 2: 192.168.1.10:47808 (self)

BR2 - BBMD/Router Configuration
- Global IP Address: 237.2.2.2:47809 (global B/IP address of NAT B)
- B/IP Address Net 1: 192.168.1.10:47809
- BDT Net 1: 237.2.2.2:47809 (global B/IP address of NAT B, self), 201.1.1.1:47809 (global B/IP address of NAT A)
- B/IP Address Net 3: 192.168.1.10:47808
- BDT Net 3: 192.168.1.10:47808 (self)

The broadcast distribution masks in the above BDT configurations are 255.255.255.255 indicating two-hop broadcast distribution.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved



STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Figure J-8. This figure represents a potential WAN with multiple remote sites, with BACnet being connected via a corporate intranet. In this configuration, the foreign device FD1 can connect to Network 4 using local addresses and to Networks 2 and 3 using the global IP address of the NAT routers. The foreign device FD2 can only connect to the global IP addresses on the Internet side of the NAT routers.

NAT A Configuration
- Internet IP 201.1.1.1
- Forward 201.1.1.1:47809 ➔ 192.168.1.10:47809

NAT B Configuration
- Internet IP 237.2.2.2
- Forward 237.2.2.2:47809 ➔ 192.168.1.10:47809

NAT C Configuration
- Internet IP 203.3.3.3
- Forward 203.3.3.3:47809 ➔ 192.168.20.10:47809

© ISO 2022 – All rights reserved

BR1 - BBMD/Router Configuration

Global IP Address	201.1.1.1:47809 (global B/IP address of NAT A)
B/IP Address Net 1	192.168.1.10:47809
BDT Net 1	201.1.1.1:47809 (global B/IP address of NAT A, self), 237.2.2.2:47809 (global B/IP address of NAT B), 203.3.3.3:47809 (global B/IP address of NAT C)
B/IP Address Net 2	192.168.1.10:47808
BDT Net 2	192.168.1.10:47808 (self)

BR2 - BBMD/Router Configuration

Global IP Address	237.2.2.2:47809 (global B/IP address of NAT B)
B/IP Address Net 1	192.168.1.10:47809
BDT Net 1	237.2.2.2:47809 (global B/IP address of NAT B, self), 201.1.1.1:47809 (global B/IP of NAT A), 203.3.3.3:47809 (global B/IP of NAT C)
B/IP Address Net 3	192.168.1.10:47808
BDT Net 3	192.168.1.10:47808 (self)

BR3 - BBMD/Router Configuration

Global IP Address	203.3.3.3:47809 (global B/IP address of NAT C)
B/IP Address Net 1	192.168.20.10:47809
BDT Net 1	203.3.3.3:47809 (global B/IP address of NAT C, self), 201.1.1.1:47809 (global B/IP address of NAT A), 237.2.2.2:47809 (global B/IP address of NAT B)
B/IP Address Net 4	192.168.20.10:47808
BDT Net 4	192.168.20.10:47808 (self), 192.168.21.10:47808 (B/IP address of BBMD B1)

B1 - BBMD Configuration

B/IP Address	192.168.21.10:47808
BDT Net 4	192.168.21.10:47808 (self), 192.168.20.10:47808 (B/IP address of BBMD BR3)

The broadcast distribution masks in the above BDT configurations are 255.255.255.255, indicating two-hop broadcast distribution.

J.8 Use of IP Multicast within BACnet/IP

BACnet/IP devices that so desire may alternatively use IP multicasting as a method for distributing BACnet broadcast messages, subject to the constraints imposed in this clause. This is accomplished through the use of an IP class D address which is made up of a single multicast group identifier rather than a combination of network and host IDs. Such devices shall be referred to as B/IP-M devices. The use of IP multicasting also requires that devices comprising a multicast group that reside on more than one IP subnet be served by IP routers capable of supporting IP multicast distribution. (See RFC 1112.) Note that all B/IP-M devices shall also be capable of processing unicast messages and shall each have a unique unicast IP address. All B/IP devices sharing a common IP multicast address should also share a common BACnet network number.

J.8.1 B/IP Multicast (B/IP-M) Concept

For the purposes of BACnet/IP, a B/IP-M group functions logically in the same manner as an IP subnet in the previous clauses. The B/IP multicast group address replaces the B/IP broadcast address for members of the group. The following constraints apply:

1. If the B/IP-M group is part of a BACnet network with B/IP non-multicast devices, there shall be one, and only one, BBMD configured to serve the B/IP-M group devices. A BACnet network comprised solely of B/IP-M devices need not have a BBMD unless foreign devices are to be supported.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

2. In order to prevent the receipt of multiple broadcast messages, devices that are in the B/IP-M group and B/IP non-multicast devices may not share the same IP subnet. Note that this does not necessarily preclude them from sharing the same physical LAN if the IP router serving the LAN can support multiple IP subnets.

J.8.2 B/IP-M Use of BVLL Messages

B/IP-M devices shall use the Original-Unicast-NPDU BVLL message for directed messages to any B/IP device within the BACnet/IP network. B/IP-M devices shall use the Original-Broadcast-NPDU BVLL message for the purpose of transmitting BACnet "local" and "global" broadcasts and shall use the B/IP-M group address as the destination IP address.

J.8.3 B/IP-M BBMD Operation

BBMDs function as described in Clause J.4.5 except that the BBMD serving the B/IP-M group shall also be a member of the group and that the B/IP-M group address is used analogously to the B/IP broadcast address with respect to the B/IP-M group. It is also required that the BDT entry for each BBMD serving a B/IP-M group shall use a broadcast distribution mask of all 1's to force "two-hop" BBMD-to-BBMD broadcast distribution. This is to prevent the multiple receipt of broadcast messages that would occur if the B/IP-M BBMD were on the same IP subnet as any of the B/IP-M devices themselves and a "directed broadcast" were used. The following paragraphs summarize the relevant operations of BBMDs that serve a B/IP-M group:

Upon receipt of an Original-Broadcast-NPDU via its B/IP-M group address, a BBMD shall forward the message to other entries in its BDT (as well as to any devices in its FDT if the BBMD also supports foreign device registration) as described in Clause J.4.5.

Upon receipt of a Forwarded-NPDU from a peer BBMD, the BBMD shall re-transmit the message using the B/IP-M group address (as well as direct it to any devices in its FDT if the BBMD also supports foreign device registration).

Upon receipt of a BVLL Distribute-Broadcast-To-Network message from a registered foreign device, the receiving BBMD shall transmit a BVLL Forwarded-NPDU message using the B/IP-M group address as the destination address. In addition, a Forwarded-NPDU message shall be sent to each entry in its BDT as described in Clause J.4.5 as well as directly to each foreign device currently in the BBMD's FDT except the originating node. Error processing is as described in Clause J.4.5.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX K - BACnet INTEROPERABILITY BUILDING BLOCKS (BIBBs) (NORMATIVE)

(This annex is part of this standard and is required for its use.)

BACnet Interoperability Building Blocks (BIBBs) are collections of one or more BACnet services. These are prescribed in terms of an "A" and a "B" device. Both of these devices are nodes on a BACnet internetwork. In most cases, the "A" device will act as the user of data (client), and the "B" device will be the provider of this data (server). In addition, certain BIBBs may also be predicated on the support of certain, otherwise optional, BACnet objects or properties and may place constraints on the allowable values of specific properties or service parameters.

K.1 Data Sharing BIBBs

These BIBBs prescribe the BACnet capabilities required to interoperably perform the data sharing functions enumerated in Clause 22.2.1.1 for the BACnet devices defined therein.

K.1.1 BIBB - Data Sharing-ReadProperty-A (DS-RP-A)

The A device is a user of data from device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

K.1.2 BIBB - Data Sharing-ReadProperty-B (DS-RP-B)

The B device is a provider of data to device A.

BACnet Service	Initiate	Execute
ReadProperty		x

K.1.3 BIBB - Data Sharing-ReadPropertyMultiple-A (DS-RPM-A)

The A device is a user of data from device B and requests multiple values at one time.

BACnet Service	Initiate	Execute
ReadPropertyMultiple	x	

K.1.4 BIBB - Data Sharing-ReadPropertyMultiple-B (DS-RPM-B)

The B device is a provider of data to device A and returns multiple values at one time.

BACnet Service	Initiate	Execute
ReadPropertyMultiple		x

K.1.5 BIBB - Data Sharing-WriteProperty-A (DS-WP-A)

The A device sets a value in device B.

BACnet Service	Initiate	Execute
WriteProperty	x	

K.1.6 BIBB - Data Sharing-WriteProperty-B (DS-WP-B)

The B device allows a value to be changed by device A.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
WriteProperty		x

K.1.7 BIBB - Data Sharing-WritePropertyMultiple-A (DS-WPM-A)

The A device sets multiple values in device B at one time.

BACnet Service	Initiate	Execute
WritePropertyMultiple	x	

K.1.8 BIBB - Data Sharing-WritePropertyMultiple-B (DS-WPM-B)

The B device allows multiple values to be changed by device A at one time.

BACnet Service	Initiate	Execute
WritePropertyMultiple		x

K.1.9 BIBB - Data Sharing-Change Of Value-A (DS-COV-A)

The A device is a user of COV data from device B.

BACnet Service	Initiate	Execute
SubscribeCOV	x	
ConfirmedCOVNotification		x[1]
UnconfirmedCOVNotification		x[1]

[1] Execution of at least one of these services is required.

Support for subscriptions of a limited lifetime is required, and support for subscriptions of indefinite lifetime is optional. Support for cancellation is optional, except in the case where the device is able to request indefinite subscriptions, in which case it is required.

K.1.10 BIBB - Data Sharing-Change Of Value-B (DS-COV-B)

The B device is a provider of COV data to device A.

BACnet Service	Initiate	Execute
SubscribeCOV		x
ConfirmedCOVNotification	x	
UnconfirmedCOVNotification	x	

Devices claiming conformance to DS-COV-B shall support a minimum of five concurrent subscriptions. Support for subscriptions of a limited lifetime is required, and support for subscriptions of indefinite lifetime is optional.

K.1.11 BIBB - Data Sharing-Change Of Value Property-A (DS-COVP-A)

The A device is a user of COV data from device B.

BACnet Service	Initiate	Execute
SubscribeCOVProperty	x	
ConfirmedCOVNotification		x[1]
UnconfirmedCOVNotification		x[1]

[1] Execution of at least one of these services is required.

Support for cancellation of subscriptions is optional.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.1.12 BIBB - Data Sharing-Change Of Value Property-B (DS-COVP-B)

The B device is a provider of COV data of an arbitrary property of a specified object to device A.

BACnet Service	Initiate	Execute
SubscribeCOVProperty		x
ConfirmedCOVNotification	x	
UnconfirmedCOVNotification	x	

Devices claiming conformance to DS-COVP-B shall support a minimum of five concurrent subscriptions.

K.1.13 BIBB - Data Sharing-Change Of Value Unsubscribed-A (DS-COVU-A)

The A device processes unsubscribed COV data from device B.

BACnet Service	Initiate	Execute
UnconfirmedCOVNotification		x

K.1.14 BIBB - Data Sharing-Change Of Value Unsubscribed-B (DS-COVU-B)

The B device generates unsubscribed COV notifications.

BACnet Service	Initiate	Execute
UnconfirmedCOVNotification	x	

K.1.15 BIBB - Data Sharing-View-A (DS-V-A)

The A device retrieves values from a minimum set of objects and properties and presents them to the user. Devices claiming conformance to DS-V-A shall support DS-RP-A. Device A shall be capable of using ReadProperty to retrieve any of the properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to DS-V-A shall be capable of reading and displaying the object properties listed in Table K-1.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-1. Properties for Which Presentation Is Required

Analog Objects, Integer Value, Large Analog Value, Positive Integer Value	**Binary Objects, Binary Lighting Output, BitString Value, CharacterString Value, Date Pattern Value, Date Value, Multi-state Objects, OctetString Value, Time Pattern Value, Time Value**	**Accumulator**	**Averaging**
Object_Name Present_Value Status_Flags Units	Object_Name Present_Value Status_Flags	Object_Name Present_Value Status_Flags Value_Before_Change Value_Set Pulse_Rate	Object_Name Minimum_Value Average_Value Maximum_Value
Command	**Device**	**Event Enrollment**	**Load Control**
Object_Name Present_Value In_Process All_Writes_Successful	Object_Name System_Status	Object_Name Event_State Object_Property_Reference	Object_Name Present_Value Status_Flags State_Description
Loop	**DateTime Pattern Value, DateTime Value**	**Program**	**Pulse Converter**
Object_Name Present_Value Status_Flags Setpoint	Object_Name Present_Value Status_Flags Is_UTC	Object_Name Program_State	Object_Name Present_Value Status_Flags Adjust_Value Units
Channel	**Lighting Output**	**Timer**	**Staging**
Object_Name Present_Value[1] Write_Status Status_Flags	Object_Name Present_Value Status_Flags In_Progress	Present_Value Timer_State Timer_Running	Object_Name Present_Value Present_Stage Status_Flags Units

[1] Support for the 'Lighting Command' choice option is not required.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

The format of a presented property value is unrestricted; the intent of this BIBB is not to impose how, or in what form, a device displays data values. For example, enumerated values could be displayed as icons, references could be displayed using the referenced object's name, numerical values could be displayed graphically.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for DS-V-A is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Table K-1.

K.1.16 BIBB - Data Sharing-Advanced View-A (DS-AV-A)

The A device retrieves property values and presents them to the user. Device A shall be capable of using ReadProperty to retrieve any standard property of any standard object type excluding the Life Safety, Access Control, and Elevator objects (e.g., Life Safety Point, Life Safety Zone, Access Door, Lift), except for those properties listed in Table K-2 and any

© ISO 2022 – All rights reserved

property defined by the standard as not readable via ReadProperty. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

The information conveyed by the properties in Table K-2 can be otherwise determined and as such need not be read and presented by devices claiming conformance to DS-AV-A.

Table K-2. Excluded Standard Properties

Object_Identifier
Object_Type

In order to ensure that products that claim support for DS-AV-A are capable of presenting accurate data values across the full range of values for each data type, devices claiming support for DS-AV-A shall be able to meet the requirements described in Table K-3.

Table K-3. Presentation Requirements by Datatype

NULL	Present NULL values. The format is unrestricted as long as NULL is distinguishable from other values.
BOOLEAN	Present all valid values. The format is unrestricted as long as each valid value is distinguishable.
Unsigned, Unsigned8, Unsigned16, Unsigned32	Present the complete value range, unless specifically restricted by the standard for the property being displayed. The minimum displayable range for Unsigned by DS-AV-A devices is the same as Unsigned32 with the exception of array indexes, which shall have a minimum displayable range of Unsigned16. In addition, any Unsigned property whose value is also used as an array index, such as a Multi-state object's Present_Value, shall have a minimum displayable range of Unsigned16.
INTEGER	Present the complete value range, unless specifically restricted by the standard for the property being displayed. The minimum displayable range for INTEGER shall be -2147483648...2147483647.
REAL, Double	Present the complete value range, including special values such as +-INF and NaN, unless specifically restricted by the standard for the property being displayed.
OCTET STRING	Present octet strings up to a length of 8 octets. The actual presentation of the values is unrestricted (text, numeric, iconic, etc.), as long as the individual values are distinguishable.
Character String	Present strings up to the length specified in Table K-4, encoded in any of the character sets supported by the A device.
BIT STRING	Present bit strings up to a length of 64 bits. Present the complete range of standard values defined for all standard bit string types for the Protocol_Revision claimed by the A device. The actual presentation of the values is unrestricted (text, numeric, iconic, etc.) as long as the individual values are distinguishable.
Enumerated	Present the standard values defined for all standard enumerated types for the Protocol_Revision claimed by the A device. The actual presentation of the values is unrestricted (text, numeric, iconic, etc.) as long as the individual values are distinguishable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Date	Present all valid dates, including values that contain unspecified octets (X'FF') or special date values (such as 'even days') which are defined for the Protocol_Revision claimed by the A device. Where the month, day and year fields all contain singular specified values, the content of the DayOfWeek field may be ignored. The format is unrestricted as long as each valid value is uniquely presented.
Time	Present all valid times, including values that contain unspecified octets (X'FF'). The format is unrestricted as long as each valid value is uniquely presented.
BACnetObjectIdentifier	Present all valid values. The format is unrestricted as long as each valid value is distinguishable. It is acceptable that BACnetObjectIdentifier values be replaced with unique object identification values such as the object's name, where available.

For Character String property values, the A device shall be capable of presenting string values for specific BACnet properties with at least the number of characters, independent of their encoding, specified in Table K-4.

Table K-4. Minimum Character-String Lengths

Action_Text	32
Application_Software_Version	64
Description	255
Description_Of_Halt	64
Device_Type	64
File_Type	32
Firmware_Revision	64
Inactive_Text/Active_Text	32
Instance_Of	64
Location	64
Model_Name	64
Object_Name	64
Present_Value	64
Profile_Name	64
State_Text	32
Vendor_Name	64
All other character string properties	32

The above presentation requirements are not required to be applied in all circumstances, but rather shall be available for every property value in the system. This should allow a product to restrict its presentation under specific conditions yet still allow the user full access to any specific property value.

The A device shall be capable of reading and presenting all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for DS-AV-A is interoperable with devices that support DS-RP-B.

K.1.17 BIBB - Data Sharing-Modify-A (DS-M-A)

The A device writes properties that are generally expected to be adjusted during normal operation of the system. Devices claiming support for this BIBB are not expected to be capable of fully configuring BACnet devices, although they are not inherently restricted from doing so.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to DS-M-A shall be capable of commanding and relinquishing standard commandable properties at priority 8 (other priorities may also be supported), and writing the properties listed in Table K-5.

Table K-5. Standard Properties That DS-M-A Devices Shall Be Capable of Writing

Analog Objects, Binary Objects, Multi-state Objects, Positive Integer Value, Integer Value, Large Analog Value, OctetString Value, BitString Value, CharacterString Value, Date Value, Date Pattern Value, Time Value, Time Pattern Value, DateTime Value, DateTime Pattern Value, Lighting Output, Binary Lighting Output	**Command**	**Program**	**Loop**
Present_Value Out_Of_Service	Present_Value	Program_Change	Present_Value Out_Of_Service Setpoint
Accumulator	**Pulse Converter**	**Channel**	**Timer**
Present_Value Out_Of_Service Value_Before_Change Value_Set Pulse_Rate	Present_Value Out_Of_Service Adjust_Value	Present_Value[1] Out_Of_Service	Present_Value Timer_State Timer_Running
Staging			
Present_Value Out_Of_Service			

[1] Support for the 'Lighting Command' choice option is not required.

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

Table K-6. Minimum Writable Value Ranges

Datatype	Value Range
NULL	NULL
Boolean	All valid values.
Unsigned8	The complete value range (0..255).
Unsigned16	The complete value range (0..65535).
Unsigned, Unsigned32	The complete value range (0..4294967295) with the exception of array indexes which shall have a minimum writable range of Unsigned16. In addition, any Unsigned property whose value is also used as an array index, such as a Multi-state object's Present_Value, shall have a minimum writable range of Unsigned16.
INTEGER	The complete value range (-2147483648...2147483647)

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Datatype	Value Range
REAL, Double	Valid values across the complete range of the datatype except the special values such as INF, -INF, NaN, etc. The precision of values that can be written may be restricted.
OCTET STRING	Values for octet strings up to a length of 8 octets.
Character String	Strings up to lengths described in Table K-4.
BIT STRING	Values for bit strings up to a length of 64 bits, and all valid values for the complete range of all standard bit string types for the Protocol_Revision claimed by the A device.
Enumerated	The standard values defined for the property being modified as defined by the claimed Protocol_Revision of the A device.
Date	All valid dates, including values that contain unspecified octets (X'FF') or special date values (such as 'even days') which are defined for the Protocol_Revision claimed by the A device.
Time	All valid times, including values that contain unspecified octets (X'FF').
BACnetObjectIdentifier	All valid values.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for DS-M-A is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-5.

K.1.18 BIBB - Data Sharing-Advanced Modify-A (DS-AM-A)

The A device is able to use WriteProperty to modify any standard property of any standard object type excluding the Life Safety, Access Control, and Elevator objects (e.g., Life Safety Point, Life Safety Zone, Access Door, Lift) where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer). Devices shall be capable of commanding and relinquishing standard commandable properties at any priority. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

The A device shall be capable of writing all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for DS-AM-A is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-1.

K.1.19 BIBB - Data Sharing-WriteGroup-A (DS-WG-A)

The A device uses unicast, multicast or broadcast WriteGroup to target Channel object(s) in device B. The A device shall be capable of specifying any group number and any channel number.

BACnet Service	Initiate	Execute
WriteGroup	x	

The A device modifies object property values in device B by initiating WriteGroup service requests that affect Channel objects in device B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.1.20 BIBB - Data Sharing-WriteGroup-Internal-B (DS-WG-I-B)

The B device shall contain one or more Channel objects that may be influenced by WriteGroup service requests from device A.

BACnet Service	Initiate	Execute
WriteGroup		x

Devices claiming conformance to DS-WG-I-B shall support configuration of Channel object BACnetDeviceObjectPropertyReference values that contain references to objects inside of device B only.

When performing datatype coercion of values to be written to objects internal to device B, the B device shall support coercions to all datatypes used by objects implemented in device B but is not required to support coercions for other datatypes.

K.1.21 BIBB - Data Sharing-WriteGroup-External-B (DS-WG-E-B)

The B device shall contain one or more Channel objects that may be influenced by WriteGroup service requests from device A.

BACnet Service	Initiate	Execute
WriteGroup		x
WriteProperty	x	

Devices claiming conformance to DS-WG-E-B shall also support DS-WG-I-B and DS-WP-A. The B device shall also support configuration of Channel object BACnetDeviceObjectPropertyReference values that contain Device Instances outside of device B, and shall be capable of initiating WriteProperty and optionally WritePropertyMultiple.

When performing datatype coercion of values to be written to objects by device B, the B device shall support coercions to all the datatypes shown in Table 12-63 – Datatype Coercion Rules.

K.1.22 BIBB - Data Sharing-Value Source Information-B (DS-VSI-B)

The B device implements the value source mechanism described in Clause 19 in all of its commandable objects and 0 or more of its non-commandable objects. For B devices that do not contain any commandable objects, the device shall implement the value source mechanism for at least one non-commandable object.

K.1.23 BIBB - Data Sharing-Change Of Value Multiple-A (DS-COVM-A)

The A device is a user of COV data from device B that is provided through COV-multiple notifications.

BACnet Service	Initiate	Execute
SubscribeCOVPropertyMultiple	x	
ConfirmedCOVNotificationMultiple		x[1]
UnconfirmedCOVNotificationMultiple		x[1]

[1] Execution of at least one of these services is required.

Support for subscription for time stamped changes is optional. Support for issuing both forms of 'Issue Confirmed Notifications' is not required. Support for execution of ConfirmedCOVNotificationMultiple is not required if 'Issue Confirmed Notifications' in SubscribeCOVPropertyMultiple is never sent as TRUE. Support for execution of UnconfirmedCOVNotificationMultiple is not required if 'Issue Confirmed Notifications' in SubscribeCOVPropertyMultiple is never sent as FALSE.

K.1.24 BIBB - Data Sharing-Change Of Value Multiple-B (DS-COVM-B)

The B device is a provider of COV data of arbitrary properties of specified objects, reported by COV multiple notifications to device A.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
SubscribeCOVPropertyMultiple		x
ConfirmedCOVNotificationMultiple	x	
UnconfirmedCOVNotificationMultiple	x	

Devices claiming conformance to DS-COVM-B shall support a minimum of five COV-multiple contexts, with a minimum of five COV References per such context. Support for timestamped changes is required. Support in SubscribeCOVPropertyMultiple for execution of both 'Issue Confirmed Notifications' as TRUE and 'Issue Confirmed Notifications' as FALSE is required.

K.1.25 BIBB - Data Sharing-Life Safety View-A (DS-LSV-A)

The A device retrieves values from a minimum set of objects and properties, including life safety objects, and presents them to the user. Devices claiming conformance to this BIBB shall support DS-RP-A. Device A shall be capable of using ReadProperty to retrieve any of the properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to this BIBB shall be capable of reading and displaying the object properties listed in Table K-1, excluding properties of Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, and be capable of reading and displaying the object properties listed in Table K-7.

Table K-7. Life Safety Properties for Which Presentation Is Required

Life Safety Point	Life Safety Zone	Structured View
Object_Name	Object_Name	Object_Name
Present_Value	Present_Value	Node_Type
Tracking_Value	Tracking_Value	Node_Subtype
Status_Flags	Status_Flags	Subordinate_List
Mode	Mode	Subordinate_Annotations
Silenced	Silenced	
Operation_Expected	Operation_Expected	
Maintenance_Required	Maintenance_Required	
Direct_Reading		
Units		

The format of a presented property value is unrestricted; the intent of this BIBB is not to impose how, or in what form, a device displays data values. For example, enumerated values could be displayed as icons, references could be displayed using the referenced object's name, and numerical values could be displayed graphically.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Tables K-1 and K-7, except the objects excluded from this BIBB.

K.1.26 BIBB - Data Sharing-Life Safety Advanced View-A (DS-LSAV-A)

The A device retrieves property values and presents them to the user. Device A shall be capable of using ReadProperty to retrieve any standard property of any standard object type listed in Table K-1, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, including the objects listed in Table K-7, except for those properties listed in Table K-2 and any property defined by the standard as

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

not readable via ReadProperty. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

The information conveyed by the properties in Table K-2 can be otherwise determined and as such need not be read and presented by devices claiming conformance to this BIBB.

In order to ensure that products that claim support for this BIBB are capable of presenting accurate data values across the full range of values for each data type, devices claiming support for this BIBB shall be able to meet the requirements described in Table K-3.

For Character String property values, the A device shall be capable of presenting string values for specific BACnet properties with at least the number of characters, independent of their encoding, specified in Table K-4.

The above presentation requirements are not required to be applied in all circumstances, but rather shall be available for every property value in the system. This should allow a product to restrict its presentation under specific conditions yet still allow the user full access to any specific property value.

The A device shall be capable of reading and presenting all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Table K-7.

K.1.27 BIBB - Data Sharing-Life Safety Modify-A (DS-LSM-A)

The A device writes properties of standard objects that are generally expected to be adjusted during normal operation of the life safety system. Devices claiming support for this BIBB are not expected to be capable of fully configuring life safety BACnet devices, although they are not inherently restricted from doing so.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to this BIBB shall be capable of commanding and relinquishing standard commandable properties at priority 8 (other priorities may also be supported) of those objects listed in Table K-5 excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, and writing the properties listed in Table K-5 and Table K-8, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects.

Table K-8. Standard Properties That DS-LSM-A Devices Shall Be Capable of Writing

Life Safety Point	Life Safety Zone
Tracking_Value Mode Out_Of_Service	Tracking_Value Mode Out_Of_Service

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-7.

K.1.28 BIBB - Data Sharing-Life Safety Advanced Modify-A (DS-LSAM-A)

The A device is able to use WriteProperty to modify any standard property of object types listed in Tables K-5 and K-8, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer). Device A shall be capable of commanding and relinquishing standard commandable properties at any priority. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

The A device shall be capable of writing all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-7.

K.1.29 BIBB - Data Sharing-Access Control View-A (DS-ACV-A)

The A device retrieves values from a minimum set of objects and properties, including access control objects, and presents them to the user. Devices claiming conformance to this BIBB shall support DS-RP-A. Device A shall be capable of using ReadProperty to retrieve any of the properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to this BIBB shall be capable of reading and displaying the object properties listed in Table K-1, excluding properties of Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and staging objects, and be capable of reading and displaying the object properties listed in Table K-9.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-9. Properties for Which Presentation Is Required

Access Door	Access Point	Access Zone	Access User
Object_Name Present_Value Status_Flags Secured_Status Door_Alarm_State Out_Of_Service Maintenance_Required Masked_Alarm_Values	Object_Name Status_Flags Authentication_Status Access_Event Access_Event_Time Out_Of_Service Active_Authentication_Policy Authorization_Mode Failed_Attempts Threat_Level	Object_Name Global_Identifier Occupancy_State Status_Flags Occupancy_Count Out_Of_Service Occupancy_Count_Upper_Limit Occupancy_Count_Lower_Limit Credentials_In_Zone Passback_Mode Occupancy_Count_Enable Adjust_Value	Object_Name Global_Identifier Status_Flags User_Name User_Type
Access Rights	**Access Credential**	**Credential Data Input**	
Object_Name Global_Identifier Status_Flags Enable	Object_Name Global_Identifier Status_Flags Credential_Status Reason_For_Disable Credential_Disable Trace_Flag	Object_Name Present_Value Status_Flags Update_Time Out_Of_Service	

The format of a presented property value is unrestricted; the intent of this BIBB is not to impose how, or in what form, a device displays data values. For example, enumerated values could be displayed as icons, references could be displayed using the referenced object's name, and numerical values could be displayed graphically.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Table K-1 and K-9, except the objects excluded from this BIBB.

K.1.30 BIBB - Data Sharing-Access Control Advanced View-A (DS-ACAV-A)

The A device retrieves property values and presents them to the user. Device A shall be capable of using ReadProperty to retrieve any standard property of any standard object types listed in Table K-1, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, including the properties listed in Table K-9, except for those properties listed in Table K-2 and any property defined by the standard as not readable via ReadProperty. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

The information conveyed by the properties in Table K-2 can be otherwise determined and as such need not be read and presented by devices claiming conformance to this BIBB.

In order to ensure that products that claim support for this BIBB are capable of presenting accurate data values across the full range of values for each data type, devices claiming support for this BIBB shall be able to meet the requirements described in Table K-3.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

For Character String property values, the A device shall be capable of presenting string values for specific BACnet properties with at least the number of characters, independent of their encoding, specified in Table K-4.

The above presentation requirements are not required to be applied in all circumstances, but rather shall be available for every property value in the system. This should allow a product to restrict its presentation under specific conditions yet still allow the user full access to any specific property value.

The A device shall be capable of reading and presenting all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Table K-9.

K.1.31 BIBB - Data Sharing-Access Control Modify-A (DS-ACM-A)

The A device writes properties of standard objects that are generally expected to be adjusted during normal operation of the physical access control system. Devices claiming support for this BIBB are not expected to be capable of fully configuring access control BACnet devices, although they are not inherently restricted from doing so.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to this BIBB shall be capable of commanding and relinquishing standard commandable properties at priority 8 (other priorities may also be supported) of those objects listed in Table K-5 excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging objects, and writing the properties listed in Table K-5 and Table K-10, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects.

Table K-10. Standard Properties That DS-ACM-A Devices Shall Be Capable of Writing

Access Door	Access Point	Access Zone
Present_Value Out_Of_Service Maintenance_Required Masked_Alarm_Values	Out_Of_Service Active_Authentication_Policy Authorization_Mode Failed_Attempts Threat_Level	Occupancy_Count Out_Of_Service Occupancy_Count_Enable Adjust_Value Credentials_In_Zone Passback_Mode
Access Credential	**Credential Data Input**	
Credential_Disable Trace_Flag	Present_Value Out_Of_Service	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-9.

K.1.32 BIBB - Data Sharing-Access Control Advanced Modify-A (DS-ACAM-A)

The A device is able to use WriteProperty to modify any standard property of object types listed in Tables K-5 and K-10, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, Binary Lighting Output, and Staging

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

objects, where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g. Log_Buffer). Device A shall be capable of commanding and relinquishing standard commandable properties at any priority. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

The A device shall be capable of writing all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-9.

K.1.33 BIBB - Data Sharing-Access Control User Configuration-A (DS-ACUC-A)

The A device is able to configure device B for access control. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A for the standard access control object types Access Rights, Access User, and Access Credential, related to user configuration. The A device shall be capable of using ReadProperty to retrieve any standard property from those objects. The A device shall be capable of using WriteProperty to modify any standard property of those objects, except those properties whose write access is restricted by the standard. Device A may use alternate services to read or modify properties in Device B where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting user configuration access control objects in Device B that supports creation/deletion of those object types.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

K.1.34 BIBB - Data Sharing-Access Control User Configuration-B (DS-ACUC-B)

The B device responds to requests to read or write standard properties of the supported user configuration access control object types Access Rights, Access User, and Access Credential.

BACnet Service	Initiate	Execute
ReadProperty		x
WriteProperty		x

K.1.35 BIBB - Data Sharing-Access Control Site Configuration-A (DS-ACSC-A)

The A device is able to configure device B for access control. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A for the standard access control object types Access Point, Access Zone, Access Door, and Credential Data Input, related to site configuration. The A device shall be capable of using ReadProperty to retrieve any supported standard property from those objects. The A device shall be capable of using WriteProperty to modify any standard property of those objects, except those properties whose write access is restricted by the standard. Device A may use alternate services to read or modify properties in Device B where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting site configuration access control objects in a B device that supports creation/deletion of those object types.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

K.1.36 BIBB - Data Sharing-Access Control Site Configuration-B (DS-ACSC-B)

The B device responds to requests to read or write standard properties of the supported site configuration access control object types Access Point, Access Zone, Access Door, and Credential Data Input.

BACnet Service	Initiate	Execute
ReadProperty		x
WriteProperty		x

K.1.37 BIBB - Data Sharing-Access Control Access Door-A (DS-ACAD-A)

The A device is able to configure and command Access Door objects. Devices claiming conformance to DS-ACAD-A shall support reading and writing all forms of all standard properties of Access Door objects. Support of DS-RP-A and DS-WP-A is required.

K.1.38 BIBB - Data Sharing-Access Control Access Door-B (DS-ACAD-B)

The B device provides door control functionality for physical access control. Devices claiming conformance to DS-ACAD-B shall support at least one Access Door object. Support of DS-RP-B and DS-WP-B is required.

K.1.39 BIBB - Data Sharing-Access Control Credential Data Input-A (DS-ACCDI-A)

The A device is able to configure and command Credential Data Input objects. Devices claiming conformance to DS-ACCDI-A shall support reading and writing all forms of all standard properties of Credential Data Input objects. Devices claiming conformance to DS-ACCDI-A shall be able to subscribe for COV notifications in Credential Data Input objects and receive all forms of BACnetAuthenticationFactor values conveyed in COV notifications. Support of DS-RP-A, DS-WP-A, and DS-COV-A is required.

K.1.40 BIBB - Data Sharing-Access Control Credential Data Input-B (DS-ACCDI-B)

The B device provides credential data input functionality for physical access control. Devices claiming conformance to DS-ACAD-B shall support at least one Credential Data Input object that supports COV reporting of its Present_Value property. Support of DS-RP-B, DS-WP-B, and DS-COV-B is required.

K.1.41 BIBB - Data Sharing-Lighting Output-A (DS-LO-A)

The A device modifies properties in Lighting Output, Binary Lighting Output, Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to DS-LO-A shall be able to write to the Present_Value property of the Lighting Output, Binary Lighting Output, Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

A device claiming support for DS-LO-A is interoperable with devices that support any combination of DS-LO-B, DS-BLO-B, or DS-WP-B in the case of Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

K.1.42 BIBB - Data Sharing-Lighting Output Status-A (DS-LOS-A)

The A device reads properties in Lighting Output, Binary Lighting Output, Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to DS-LOS-A shall be able to read the Present_Value and Egress_Active properties of Lighting Output and Binary Lighting Output object types and to read the Present_Value property of Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

A device claiming support for DS-LOS-A is interoperable with devices that support any combination of DS-LO-B, DS-BLO-B, or DS-RP-B in the case of Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

K.1.43 BIBB - Data Sharing-Advanced Lighting Output-A (DS-ALO-A)

The A device modifies properties in Lighting Output, Binary Lighting Output, Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

BACnet Service	Initiate	Execute
WriteProperty	x	

The A device shall be capable of modifying all of the standard properties of the Lighting Output, Binary Lighting Output, Channel, Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

A device claiming support for DS-ALO-A is interoperable with devices that support any combination of DS-LO-B, DS-BLO-B, DS-RP-B, or DS-WP-B in the case of Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, and Multi-State Value object types.

K.1.44 BIBB - Data Sharing-Lighting Output-B (DS-LO-B)

The B device implements the Lighting Output object type.

BACnet Service	Initiate	Execute
ReadProperty		x
WriteProperty		x

Devices claiming conformance to DS-LO-B shall support the Lighting Output object type.

A device claiming support for DS-LO-B is interoperable with devices that support DS-LO-A, or DS-ALO-A.

K.1.45 BIBB - Data Sharing-Binary Lighting Output-B (DS-BLO-B)

The B device implements the Binary Lighting Output object type.

BACnet Service	Initiate	Execute
ReadProperty		x
WriteProperty		x

Devices claiming conformance to DS-BLO-B shall support the Binary Lighting Output object type.

A device claiming support for DS-BLO-B is interoperable with devices that support DS-LO-A, or DS-ALO-A.

K.1.46 BIBB - Data Sharing-Lighting View-A (DS-LV-A)

The A device retrieves values from a minimum set of lighting objects and properties and presents them to the user. Devices claiming conformance to this BIBB shall support DS-RP-A. The A device shall be capable of using ReadProperty to retrieve any of the properties listed below. The A device may use alternate services where support for execution of the alternate service is supported by the B device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to this BIBB shall be capable of reading and displaying the object properties listed in Table K-11.

Table K-11. Object Properties for Which Presentation Is Required

Lighting Output	Binary Lighting Output	Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, Multi-State Value	Channel
Object_Name Present_Value Tracking_Value Lighting_Command Status_Flags Blink_Warn_Enable Current_Command_Priority	Object_Name Present_Value Status_Flags Blink_Warn_Enable Current_Command_Priority	Object_Name Present_Value Status_Flags	Object_Name Present_Value Status_Flags

The format of a presented property value is unrestricted; the intent of this BIBB is not to impose how, or in what form, a device displays data values. For example, enumerated values could be displayed as icons, references could be displayed using the referenced object's name, and numerical values could be displayed graphically.

Actions taken by the A device when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and one or more of the objects listed in Table K-11.

K.1.47 BIBB - Data Sharing-Lighting Advanced View-A (DS-LAV-A)

The A device retrieves property values and presents them to the user. The A device shall be capable of using ReadProperty to retrieve any standard property of any standard object type listed in Table K-1, excluding Averaging, Loop, Accumulator, and Pulse Converter objects, except for those properties listed in Table K-2 and any property defined by the standard as not readable via ReadProperty. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

The information conveyed by the properties in Table K-2 can be otherwise determined and as such need not be read and presented by devices claiming conformance to this BIBB.

In order to ensure that products that claim support for this BIBB are capable of presenting accurate data values across the full range of values for each data type, devices claiming support for this BIBB shall be able to meet the requirements described in Table K-3. In addition, the device shall be able to present all valid values for fields of the Lighting_Command property. The format is unrestricted as long as each valid value is distinguishable.

For Character String property values, the A device shall be capable of presenting string values for specific BACnet properties with at least the number of characters, independent of their encoding, specified in Table K-4.

The above presentation requirements are not required to be applied in all circumstances, but rather shall be available for every property value in the system. This should allow a product to restrict its presentation under specific conditions yet still allow the user full access to any specific property value.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The A device shall be capable of reading and presenting all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Actions taken by the A device when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and one or more of the objects listed in Table K-1.

K.1.48 BIBB - Data Sharing-Lighting Modify-A (DS-LM-A)

The A device writes properties of standard objects that are generally expected to be adjusted during normal operation of the lighting system. Devices claiming support for this BIBB are not expected to be capable of fully configuring lighting controller BACnet devices, although they are not inherently restricted from doing so.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to this BIBB shall be capable of commanding and relinquishing standard commandable properties at priority 8 (other priorities may also be supported) of those objects listed in Table K-12 and writing the properties listed in Table K-12.

Table K-12. Standard Properties That DS-LM-A Devices Shall Be Capable of Writing

Lighting Output	**Binary Lighting Output**	**Analog Output, Analog Value, Binary Output, Binary Value, Multi-State Output, Multi-State Value**	**Channel**
Present_Value Lighting_Command	Present_Value	Present_Value	Present_Value

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6. In addition, the device shall be able to manipulate all of the fields of the Lighting_Command property.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-12.

K.1.49 BIBB - Data Sharing-Lighting Advanced Modify-A (DS-LAM-A)

The A device is able to use WriteProperty to modify any standard property of object types listed in Tables K-5, excluding Averaging, Command, Program, Loop, Accumulator, Timer, and Pulse Converter objects, where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer). The A device shall be capable of commanding and relinquishing standard commandable properties at any priority. The A device may use alternate services where support for execution of the alternate service is supported by the B device.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6. In addition, the device shall be able to manipulate all of the fields of the Lighting_Command property.

The A device shall be capable of writing all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of the objects listed in Table K-5.

K.1.50 BIBB - Data Sharing-Elevator View-A (DS-EV-A)

The A device retrieves values from a minimum set of objects and properties, including elevator objects, and presents them to the user. Devices claiming conformance to this BIBB shall support DS-RP-A. Device A shall be capable of using ReadProperty to retrieve any of the properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

Devices claiming conformance to this BIBB shall be capable of reading and displaying the object properties listed in Table
K-1, excluding properties of Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects, and be capable of reading and displaying the object properties listed in Table K-13

Table K-13. Elevator Object Properties for which Presentation is Required

Elevator Group	Lift	Escalator
Object_Name Group_ID Group_Mode	Object_Name Status_Flags Car_Position Car_Moving_Direction Car_Assigned_Direction	Object_Name Status_Flags Operation_Direction Escalator_Mode

The format of a presented property value is unrestricted; the intent of this BIBB is not to impose how, or in what form, a device displays data values. For example, enumerated values could be displayed as icons, references could be displayed using the referenced object's name, and numerical values could be displayed graphically.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Tables K-1 and K-13, except the objects excluded from this BIBB.

K.1.51 BIBB - Data Sharing-Elevator Advanced View-A (DS-EAV-A)

The A retrieves property values and presents them to the user. Device A shall be capable of using ReadProperty to retrieve any standard property of any standard object type listed in Table K-1 and K-13, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects, except for those properties listed in Table K-2 and any property defined by the standard as not readable via ReadProperty. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	

The information conveyed by the properties in Table K-2 can be otherwise determined and as such need not be read and presented by devices claiming conformance to this BIBB.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

In order to ensure that products that claim support for this BIBB are capable of presenting accurate data values across the full range of values for each data type, devices claiming support for this BIBB shall be able to meet the requirements described in Table K-3.

For Character String property values, the A device shall be capable of presenting string values for specific BACnet properties with at least the number of characters, independent of their encoding, specified in Table K-4.

The above presentation requirements are not required to be applied in all circumstances, but rather shall be available for every property value in the system. This should allow a product to restrict its presentation under specific conditions yet still allow the user full access to any specific property value.

The A device shall be capable of reading and presenting all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-RP-B and support one or more of the objects listed in Table K-13.

K.1.52 BIBB - Data Sharing-Elevator Modify-A (DS-EM-A)

The A device writes properties of standard objects that are generally expected to be adjusted during normal operation of the elevator system. Devices claiming support for this BIBB are not expected to be capable of fully configuring elevator controller BACnet devices, although they are not inherently restricted from doing so.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming conformance to this BIBB shall be capable of commanding and relinquishing standard commandable properties at priority 8 (other priorities may also be supported) of those objects listed in Table K-5 and Table K-13 excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects, and writing the properties listed in Table K-5 and Table K-14, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects.

Table K-14. Standard Properties That DS-EM-A Devices Shall Be Capable of Writing

Elevator Group	Lift	Escalator
Group_Mode	Making_Car_Call Car_Door_Command Car_Mode	Escalator_Mode

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of
the objects listed in Table K-13.

K.1.53 BIBB - Data Sharing-Elevator Advanced Modify-A (DS-EAM-A)

The A device is able to use WriteProperty to modify any standard property of object types listed in Tables K-5 and K-14, excluding Averaging, Loop, Accumulator, Pulse Converter, Channel, Lighting Output, and Binary Lighting Output objects, where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer). Device A shall be capable of commanding and relinquishing standard commandable properties at any

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

priority. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
WriteProperty	x	

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Table K-6.

The A device shall be capable of writing all standard forms of the datatypes as defined per the A device's claimed Protocol_Revision.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for this BIBB is interoperable with devices that support DS-WP-B and support one or more of
the objects listed in Table K-13.

K.2 Alarm and Event Management BIBBs

These BIBBs prescribe the BACnet capabilities required to interoperably perform the alarm and event management functions enumerated in Clause 22.2.1.2 for the BACnet devices defined therein.

K.2.1 BIBB - Alarm and Event Management-Notification-A (AE-N-A)

The A device processes notifications about alarms and other events.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x

Devices claiming conformance to AE-N-A shall be able to process notifications from any standard or proprietary event-generating object of any standard or proprietary event type (excluding the CHANGE_OF_LIFE_SAFETY, ACCESS_EVENT, and/or BUFFER_READY event types).

K.2.2 BIBB - Alarm and Event Management-Notification-Internal-B (AE-N-I-B)

Device B generates notifications about alarms and other events.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x[1]	
UnconfirmedEventNotification	x	

[1] Devices that contain only read-only Recipient_List properties and no Notification Forwarder objects are not required to have the capability to initiate ConfirmedEventNotification requests.

Devices claiming support for AE-N-I-B shall also claim support for AE-INFO-B.

Devices claiming support for AE-N-I-B shall also support either Intrinsic or Algorithmic reporting. Any device that supports the generation of event notifications that require operator acknowledgment shall support AE-ACK-B and AE-INFO-B. Any device that supports the generation of TO_FAULT or TO_OFFNORMAL event notifications shall support AE-INFO-B.

Devices that only support generation of CHANGE_OF_LIFE_SAFETY, ACCESS_EVENT, and/or BUFFER_READY notifications shall not claim support for this BIBB.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.2.3 BIBB - Alarm and Event Management-Notification-External-B (AE-N-E-B)

Device B contains an Event Enrollment object that monitors values in another device. Device B is capable of generating event notifications for alarm conditions based on value(s) in another device. Devices conforming to this BIBB shall conform to DS-RP-A, AE-N-I-B, and shall support at least 1 Event Enrollment object with an Object_Property_Reference property that supports references to properties in objects contained in other devices. Any device that supports the generation of event notifications that require operator acknowledgment shall support AE-ACK-B and AE-INFO-B. Any device that supports the generation of TO_FAULT or TO_OFFNORMAL event notifications shall support AE-INFO-B.

Devices claiming support for this BIBB shall either support writable Recipient_List properties or support Notification_Forwarder objects.

Devices that only support Event Enrollment objects that only support generation of CHANGE_OF_LIFE_SAFETY, ACCESS_EVENT, and/or BUFFER_READY notifications shall not claim support for this BIBB.

K.2.4 BIBB - Alarm and Event Management-Acknowledge-A (AE-ACK-A)

Device A acknowledges alarm/event notifications.

BACnet Service	Initiate	Execute
AcknowledgeAlarm	x	

Devices claiming conformance to AE-ACK-A shall also support the Local_Date and Local_Time properties in the Device object.

K.2.5 BIBB - Alarm and Event Management-Acknowledge-B (AE-ACK-B)

Device B processes acknowledgments of previously transmitted alarm/event notifications.

BACnet Service	Initiate	Execute
AcknowledgeAlarm		x

To support this BIBB the device is also required to support acknowledgeable alarms.

K.2.6 BIBB - Alarm and Event Management-Alarm Summary-A (AE-ASUM-A)

Device A requests summaries of alarms from device B.

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.

BACnet Service	Initiate	Execute
GetAlarmSummary	x	

K.2.7 BIBB - Alarm and Event Management-Alarm Summary-B (AE-ASUM-B)

The functionality covered by this BIBB has been deprecated. This BIBB is included solely for historical purposes.

Device B provides summaries of alarms to device A.

BACnet Service	Initiate	Execute
GetAlarmSummary		x

K.2.8 BIBB - Alarm and Event Management-Enrollment Summary-A (AE-ESUM-A)

Device A requests event enrollments from device B.

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
GetEnrollmentSummary	x	

K.2.9 BIBB - Alarm and Event Management-Enrollment Summary-B (AE-ESUM-B)

The functionality covered by this BIBB has been deprecated. This BIBB is included solely for historical purposes.

Device B provides event enrollments to device A.

BACnet Service	Initiate	Execute
GetEnrollmentSummary		x

K.2.10 BIBB - Alarm and Event Management-Information-A (AE-INFO-A)

Device A requests event information from device B.

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.

BACnet Service	Initiate	Execute
GetEventInformation	x	

K.2.11 BIBB - Alarm and Event Management-Information-B (AE-INFO-B)

Device B provides event information to device A.

BACnet Service	Initiate	Execute
GetEventInformation		x

K.2.12 BIBB - Alarm and Event Management-LifeSafety-A (AE-LS-A)

Life safety device A is able to process and acknowledge life safety notifications and is able to request silence and reset operations from life safety device B.

BACnet Service	Initiate	Execute
LifeSafetyOperation	x	
ConfirmedEventNotification		x
UnconfirmedEventNotification		x
AcknowledgeAlarm	x	

K.2.13 BIBB - Alarm and Event Management-LifeSafety-B (AE-LS-B)

Life safety device B is able to generate life safety notifications and is able to process silence and reset operations on its life safety objects.

BACnet Service	Initiate	Execute
LifeSafetyOperation		x
ConfirmedEventNotification	x	
UnconfirmedEventNotification	x	

Devices claiming conformance to AE-LS-B shall support at least one instance of a Life Safety Point or Life Safety Zone object and shall be able to generate ConfirmedEventNotification and UnconfirmedEventNotification service requests describing CHANGE_OF_LIFE_SAFETY event transitions.

Any device that supports the generation of event notifications that require operator acknowledgment shall support AE-ACK-B and AE-INFO-B. Any device that supports the generation of TO_FAULT or TO_OFFNORMAL event notifications shall support AE-INFO-B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.2.14 BIBB - Alarm and Event Management-View Notifications-A (AE-VN-A)

Device A presents basic alarm and event notifications to the user. Device A shall support AE-N-A and shall be capable of presenting any alarm or event notifications covered by AE-N-A to the user. The information conveyed to the user shall include identification of the event-generating object or the monitored object, the event's timestamp and the event's Message Text. Any other information conveyed to the user shall be consistent with the data contained in the notification.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x

Device A shall be capable of presenting at least 32 characters of Message Text, although it is suggested that devices claiming this BIBB be capable of displaying 255 characters of Message Text.

A device claiming support for AE-VN-A is interoperable with devices that support AE-N-I-B.

K.2.15 BIBB - Alarm and Event Management-Advanced View Notifications-A (AE-AVN-A)

Device A presents complete alarm and event notifications to the user. Device A shall support AE-VN-A. In addition to the requirements of AE-VN-A, Device A shall be capable of presenting the event Notification Class, Priority, Notify Type, Ack Required, To State and Event Values to the user. Device A shall be capable of presenting at least 255 characters of Message Text.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x

A device claiming support for AE-AVN-A is interoperable with devices that support AE-N-I-B.

K.2.16 BIBB - Alarm and Event Management-View and Modify-A (AE-VM-A)

Device A displays and modifies limits and related parameters in standard event-initiating objects.

Device A shall support DS-RP-A and DS-WP-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the event and fault algorithm parameters listed in Tables K-11 and K-12. Such parameters may be present in individual properties, in event parameter properties, or in fault parameter properties. See the respective property specifications. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-VM-A shall be capable of reading, presenting and writing all standard properties in standard objects that are configuration parameters of standard event and/or fault algorithms that have high and low numerical limits, as listed in Tables K-15 and K-16.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-15. Event Algorithm Parameters That Devices Shall Be Capable of Presenting and Modifying

Event Algorithm	Event Algorithm Parameter
DOUBLE_OUT_OF_RANGE	pLowLimit pHighLimit pDeadband
FLOATING_LIMIT	pLowDiffLimit pHighDiffLimit pDeadband
OUT_OF_RANGE	pLowLimit pHighLimit pDeadband
SIGNED_OUT_OF_RANGE	pLowLimit pHighLimit pDeadband
UNSIGNED_OUT_OF_RANGE	pLowLimit pHighLimit pDeadband
UNSIGNED_RANGE	pLowLimit pHighLimit

Table K-16. Fault Algorithm Parameters That Devices Shall Be Capable of Presenting and Modifying

Fault Algorithm	Fault Algorithm Parameter
FAULT_OUT_OF_RANGE	pMinimumNormalValue pMaximumNormalValue

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Tables K-4 and K-6.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects, properties, and parameters of event and fault algorithms that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-VM-A is interoperable with devices that support AE-N-I-B.

K.2.17 BIBB - Alarm and Event Management-Advanced View and Modify-A (AE-AVM-A)

Device A configures standard event-initiating objects, Notification Class objects, and Notification Forwarder objects in Device B. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the properties listed below and all forms of standard properties that contain parameters, or references to parameters, of event and/or fault algorithms. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Event Enrollment, Notification Class, and Notification Forwarder objects in the B device.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-AVM-A are not required to perform any of these functions for event and fault algorithms ACCESS_EVENT, BUFFER_READY, CHANGE_OF_LIFE_SAFETY, and FAULT_LIFE_SAFETY.

In addition to being able to interoperate with algorithm parameters, devices claiming conformance to AE-AVM-A shall also be capable of reading, presenting, and writing all standard forms of all common properties related to event-state-detection and alarm-acknowledgement, as listed in Table K-17, and those related to event-distribution as listed in Table

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K-18, for all standard object types that include these properties, except for objects that only apply ACCESS_EVENT, CHANGE_OF_LIFE_SAFETY, or FAULT_LIFE_SAFETY event and fault algorithms.

Table K-17. Common Event-State-Detection and Alarm-Acknowledgement Properties

Event-State-Detection Properties	Alarm-Acknowledgement Properties
Event_State[1] Event_Enable Event_Time_Stamps[1] Notify_Type Event_Detection_Enable Event_Message_Texts[1] Event_Message_Texts_Config Event_Algorithm_Inhibit_Ref Event_Algorithm_Inhibit Reliability_Evaluation_Inhibit	Acked_Transitions[1]

[1] AE-AVM-A devices need only be capable of presenting these properties; not modifying them.

Table K-18. Event-Notification-Distribution Properties

All Standard Object Types	Notification Class	Notification Forwarder
Notification_Class	Priority Ack_Required Recipient_List	Recipient_List Subscribed_Recipients Process_Identifier_Filter Port_Filter Local_Forwarding_Only

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Tables K-4 and K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects, properties, and forms of event and fault algorithms parameters that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-AVM-A is interoperable with devices that support AE-N-I-B or AE-N-E-B and is interoperable with devices that support AE-NF-B or AE-NF-I-B.

K.2.18 BIBB - Alarm and Event Management-Alarm Summary View-A (AE-AS-A)

Device A presents alarm summary information to the user. Device A uses GetEventInformation to retrieve or update alarm summary information presented to the user. When confronted with a device that does not support execution of GetEventInformation, Device A uses GetAlarmSummary instead. Device A may use alternate alarm and event summary services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
GetEventInformation	x	
GetAlarmSummary	x	

Device A is not required to rely solely on the event summary services for retrieval of event information. It may use the information contained in received event notifications to build the alarm summary. In such a case, a device claiming conformance to this BIBB shall use the summarization services to update this information. Presentation content and format is a local matter.

A device claiming support for AE-AS-A is interoperable with devices that support AE-INFO-B, or AE-ASUM-B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.2.19 BIBB - Alarm and Event Management-Event Log View-A (AE-ELV-A)

The A device displays event log data from the B device.

BACnet Service	Initiate	Execute
ReadRange	x	

The A device uses ReadRange to retrieve and display the Event Log object's Log_Buffer property. Devices claiming support for this BIBB shall be capable of presenting Event Logs containing any type event notifications but are not required to display the Event_Values field of event notifications for Event_Types that are defined in a Protocol_Revision newer than that of the A device.

The A device has to be able to display the information, with the same data requirments, indicated in AE-VN-A.

A device claiming support for AE-ELV-A is interoperable with devices that support AE-EL-I-B or AE-EL-E-B.

K.2.20 BIBB - Alarm and Event Management-Event Log View and Modify-A (AE-ELVM-A)

The A device displays event log data from the B device and manipulates event log collection parameters in the B device. Devices claiming support for this BIBB shall support DS-RP-A and DS-WP-A.

BACnet Service	Initiate	Execute
ReadRange	x	
ReadProperty	x	
WriteProperty	x	

The A device shall be capable of using ReadRange to retrieve and display the Event Log's Log_Buffer property, ReadProperty to retrieve and display Event Log properties, WriteProperty to modify Event Log properties. The properties that the A device shall be capable of reading and writing are listed below. Device A may use alternate services where support for execution of the alternate service is supported by the B Device.

Table K-19. Event Log Object Properties That AE-ELVM-A Devices Shall Be Capable of Presenting and Modifying

Enable	Notification_Threshold
Start_Time	Last_Notify_Record (retrieve only)
Stop_Time	Event_State (retrieve only)
Stop_When_Full	Notification_Class
Buffer_Size	Event_Enable
Record_Count	Event_Time_Stamps (retrieve only)
Total_Record_Count (retrieve only)	

Devices claiming support for this BIBB shall be capable of presenting all of the fields of the Log_Buffer records, and all types of event notifications but are not required to display the Event_Values field of event notifications for Event_Types that are defined in a Protocol_Revision newer than that claimed by the A device.

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Tables K-4 and K-6.

A device claiming support for AE-ELVM-A is interoperable with devices that support AE-EL-I-B or AE-EL-E-B.

K.2.21 BIBB - Alarm and Event Management-Event Log-Internal-B (AE-EL-I-B)

The B device collects the event notifications in an internal buffer. Each device claiming conformance to AE-EL-I-B shall be able to support at least one Event Log object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
ReadRange		x

K.2.22 BIBB - Alarm and Event Management-Event Log-External-B (AE-EL-E-B)

The B device collects, in an internal buffer, confirmed and unconfirmed event notifications that are received from other devices. Each device claiming conformance to AE-EL-E-B must be able to support at least one Event Log object and shall be capable of logging all forms of event notifications.

BACnet Service	Initiate	Execute
ReadRange		x
ConfirmedEventNotification		x
UnconfirmedEventNotification		x

K.2.23 BIBB - Alarm and Event Management-Notification Forwarder-B (AE-NF-B)

Device B forwards alarm and event notifications for other devices.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	x
UnconfirmedEventNotification	x	x
AddListElement		x
RemoveListElement		x

Devices claiming conformance to AE-NF-B shall support the Notification Forwarder object type with configurable Recipient_List properties that can contain at least 8 entries, and with a Subscribed_Recipients property with at least 8 entries.

The Notification Forwarder objects shall be capable of forwarding all event notifications received by the device. If the Process_Identifier_Filter properties of the Notification Forwarder objects are not configurable, then at least one Notification Forwarder object shall have a Process_Identifier_Filter property with a value of NULL or 0 in order to forward notifications from devices that only send event notifications using local broadcasts.

K.2.24 BIBB - Alarm and Event Management-Notification Forwarder-Internal-B (AE-NF-I-B)

Device B forwards alarm and event notifications for the local device.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	
UnconfirmedEventNotification	x	
AddListElement		x
RemoveListElement		x

Devices claiming conformance to AE-NF-I-B shall support the Notification Forwarder object type with configurable Recipient_List properties that can contain at least 8 entries, and with a Subscribed_Recipients property with at least 8 entries.

K.2.25 BIBB - Alarm and Event Management-Configurable Recipient Lists-B (AE-CRL-B)

Device B supports configuration of its Recipient_List properties.

BACnet Service	Initiate	Execute
WriteProperty		x
Who-Is	x	
I-Am		x

Devices claiming conformance to AE-CRL-B shall support at least one instance of a Notification Class or Notification Forwarder object.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The Recipient_List properties of all Notification Class and Notification Forwarder objects present in the device shall be writable, support all forms of BACnetDestination, and support a minimum of 8 entries. The Recipient_List property shall be writable using the WriteProperty service. Support of DM-LM-B to modify the Recipient_List property is optional.

A device claiming support for AE-CRL-B is interoperable with devices that support AE-AVM-A.

K.2.26 BIBB - Alarm and Event Management-Temporary Event Subscription-A (AE-TES-A)

Device A subscribes in Notification Forwarder objects for temporary reception of event notifications from Device B, is able to renew the subscription, and to unsubscribe respectively.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x
AddListElement	x	
RemoveListElement	x	

Devices claiming conformance to AE-TES-A shall provide the ability to add itself as an event notification recipient in the Subscribed_Recipients property of any Notification Forwarder object present in any device, to renew, and to cancel such subscriptions respectively. Devices claiming conformance to this BIBB shall use AddListElement and RemoveListElement to update Notification Forwarder objects' Subscribed_Recipients property for the purpose of self-subscription, renewal, and cancellation.

A device claiming support for AE-TES-A is interoperable with devices that support AE-NF-B or AE-NF-I-B.

K.2.27 BIBB - Alarm and Event Management-Life Safety View Notifications-A (AE-LSVN-A)

Device A presents alarm and event state information for events which the A device is configured to receive, including life safety alarms. Devices claiming conformance to this BIBB shall support AE-N-A and AE-LS-A and shall support presentation of alarm states of event type CHANGE_OF_LIFE_SAFETY.

A device claiming support for AE-LSVN-A is interoperable with devices that support AE-N-I-B, AE-N-E-B or AE-LS-B.

K.2.28 BIBB - Alarm and Event Management-Life Safety Advanced View Notifications-A (AE-LSAVN-A)

Device A presents complete alarm and event notifications to the user, including life safety alarms. Devices claiming conformance to this BIBB shall support AE-AVN-A and AE-LS-A and shall support presentation of complete alarm and event notifications of event type CHANGE_OF_LIFE_SAFETY.

A device claiming support for AE-LSAVN-A is interoperable with devices that support AE-N-I-B, AE-N-E-B or AE-LS-B.

K.2.29 BIBB - Alarm and Event Management-Life Safety View and Modify-A (AE-LSVM-A)

Device A displays and modifies limits and other event parameters in event-initiating objects in Device B, including life safety objects.

Device A shall support DS-RP-A and DS-WP-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the event and fault algorithm parameters listed below. Such parameters may be present in individual properties, in event parameter properties, or in fault parameter properties. See the respective property specifications. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-LSVM-A shall be capable of reading, presenting, and writing standard properties that are configuration parameters or references to configuration parameters of standard and life safety event and/or fault algorithms, as listed in Tables K-15, K-16, K-20, and K-21.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-20. Additional Event Algorithm Parameters That Life Safety Devices Shall Be Capable of Presenting and Modifying

Event Algorithm	Event Algorithm Parameter
CHANGE_OF_LIFE_SAFETY	pAlarmValues pLifeSafetyAlarmValues

Table K-21. Additional Fault Algorithm Parameters That Life Safety Devices Shall Be Capable of Presenting and Modifying

Fault Algorithm	Fault Algorithm Parameter
FAULT_LIFE_SAFETY	pFaultValues

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-LSVM-A is interoperable with devices that support AE-N-I-B, AE-N-E-B, or AE-LS-B.

K.2.30 BIBB - Alarm and Event Management-Life Safety Advanced View and Modify-A (AE-LSAVM-A)

Device A configures standard event-initiating objects, Notification Class objects, and Notification Forwarder objects in Device B. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the properties listed below and all forms of standard properties that contain parameters, or references to parameters, of event and/or fault algorithms. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Event Enrollment, Notification Class, and Notification Forwarder objects in the B device.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-LSAVM-A are required to read, present, and modify any properties or particular forms of properties that contain parameters, or references to parameters, related to the event and fault algorithms as required by AE-AVM-A, including CHANGE_OF_LIFE_SAFETY and FAULT_LIFE_SAFETY algorithm parameters.

Devices claiming conformance to AE-LSAVM-A shall be capable of reading, presenting, and writing all standard forms of all common properties related to event-state-detection and alarm-acknowledgement, as listed in Table K-17.

Devices claiming conformance to AE-LSAVM-A shall be capable of reading, presenting, and writing all standard forms of properties that are related to event-notification-distribution, listed in Table K-18.

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-LSAVM-A is interoperable with devices that support AE-N-I-B, AE-N-E-B, or AE-LS-B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.2.31 BIBB - Alarm and Event Management-Access Control-A (AE-AC-A)

Access control device A is able to process and acknowledge ACCESS_EVENT event notifications.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x
AcknowledgeAlarm	x	

K.2.32 BIBB - Alarm and Event Management-Access Control-B (AE-AC-B)

Access control device B is able to generate ACCESS_EVENT event notifications.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	
UnconfirmedEventNotification	x	

Devices claiming conformance to AE-AC-B shall support at least one instance of an Access Point object and shall be able to generate ConfirmedEventNotification and UnconfirmedEventNotification service requests describing ACCESS_EVENT event transitions.

Any device that supports the generation of event notifications that require operator acknowledgment shall support AE-ACK-B and AE-INFO-B. Any device that supports the generation of TO_FAULT or TO_OFFNORMAL event notifications shall support AE-INFO-B.

K.2.33 BIBB - Alarm and Event Management-Access Control Advanced View Notifications-A (AE-ACAVN-A)

Device A presents complete alarm and event notifications to the user, including access control events. Devices claiming conformance to this BIBB shall support AE-AVN-A and AE-AC-A and shall support presentation of complete alarm and event notifications of event type ACCESS_EVENT.

A device claiming support for AE-ACAVN-A is interoperable with devices that support AE-N-I-B, AE-N-E-B, or AE-AC-B.

K.2.34 BIBB - Alarm and Event Management-Access Control View and Modify-A (AE-ACVM-A)

Device A displays and modifies limits and other event parameters in event-initiating objects in Device B, including access control objects.

Device A shall support DS-RP-A and DS-WP-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the event and fault algorithm parameters listed below. Such parameters may be present in individual properties, in event parameter properties, or in fault parameter properties. See the respective property specifications. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-ACVM-A shall be capable of reading, presenting, and writing standard properties that are configuration parameters or references to configuration parameters of standard and access control event and/or fault algorithms, as listed in Tables K-15, K-16, K-22, and K-23.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-22. Additional Event Algorithm Parameters That Devices Shall Be Capable of Presenting and Modifying

Event Algorithm	Event Algorithm Parameter
ACCESS_EVENT	pAccessEvents pAccessEventTime

Table K-23. Additional Fault Algorithm Parameters That Devices Shall Be Capable of Presenting and Modifying

Fault Algorithm	Fault Algorithm Parameter
<none at this time>	

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-ACVM-A is interoperable with devices that support AE-N-I-B, AE-N-E-B, or AE-AC-B.

K.2.35 BIBB - Alarm and Event Management-Access Control Advanced View and Modify-A (AE-ACAVM-A)

Device A configures standard event-initiating objects, Notification Class objects, and Notification Forwarder objects in Device B. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the properties listed below and all forms of standard properties that contain parameters, or references to parameters, of event and/or fault algorithms. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Event Enrollment, Notification Class, and Notification Forwarder objects in the B device.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-ACAVM-A are required to read, present, and modify any properties or particular forms of properties that contain parameters, or references to parameters, related to the event and fault algorithms as required by AE-AVM-A, including ACCESS_EVENT algorithm parameters.

Devices claiming conformance to AE-ACAVM-A shall be capable of reading, presenting, and writing all standard forms of all common properties related to event-state-detection and alarm-acknowledgement, as listed in Table K-17.

Devices claiming conformance to AE-ACAVM-A shall be capable of reading, presenting, and writing all standard forms of properties that are related to event-notification-distribution, listed in Table K-18.

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-ACAVM-A is interoperable with devices that support AE-N-I-B, AE-N-E-B, or AE-AC-B.

K.2.36 BIBB - Alarm and Event Management-Elevator View Notifications-A (AE-EVN-A)

Device A presents alarm and event state information for events which the A device is configured to receive, including elevator faults. Devices claiming conformance to this BIBB shall support AE-N-A and shall support presentation of faults of fault algorithm FAULT_LISTED.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

A device claiming support for AE-EVN-A is interoperable with devices that support AE-N-I-B or AE-N-E-B.

K.2.37 BIBB - Alarm and Event Management-Elevator Advanced View Notifications-A (AE-EAVN-A)

Device A presents complete alarm and event notifications to the user, including elevator faults and alarms. Devices claiming conformance to this BIBB shall support AE-AVN-A and shall support presentation of complete fault notifications from the FAULT_LISTED fault algorithm.

A device claiming support for AE-EAVN-A is interoperable with devices that support AE-N-I-B or AE-N-E-B.

K.2.38 BIBB - Alarm and Event Management-Elevator View and Modify-A (AE-EVM-A)

Device A displays and modifies limits and other event parameters in event-initiating objects in Device B, including fault parameters for the FAULT_LISTED algorithm.

Device A shall support DS-RP-A and DS-WP-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any of the event and fault algorithm parameters listed below. Such parameters may be present in individual properties, in event parameter properties, or in fault parameter properties. See the respective property specifications. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-EVM-A shall be capable of reading, presenting, and writing standard properties that
are configuration parameters or references to configuration parameters of standard event and/or fault algorithms, as listed
in Tables K-11, K-12, and K-24.

Table K-24. Additional Event Algorithm Parameters That Device A Shall Be Capable of Presenting and Modifying

Event Algorithm	Event Algorithm Parameter
FAULT_LISTED	pMonitoredList

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-EVM-A is interoperable with devices that support AE-N-I-B or AE-N-E-B.

K.2.39 BIBB - Alarm and Event Management-Elevator Advanced View and Modify-A (AE-EAVM-A)

Device A configures standard event-initiating objects, Notification Class objects, and Notification Forwarder objects in Device B. Device A shall support DS-RP-A, DS-WP-A, and DM-OCD-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify properties and all forms of standard properties that contain parameters, or references to parameters, of event and/or fault algorithms. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Event Enrollment, Notification Class, and Notification Forwarder objects in the B device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Devices claiming conformance to AE-EAVM-A are required to read, present, and modify any properties or particular forms of properties that contain parameters, or references to parameters, related to the event and fault algorithms as required by AE-AVM-A, including FAULT_LISTED algorithm parameters.

Devices claiming conformance to AE-EAVM-A shall be capable of reading, presenting, and writing all standard forms of all common properties related to event-state-detection and alarm-acknowledgement, as listed in Table K-13.

Devices claiming conformance to AE-EAVM-A shall be capable of reading, presenting, and writing all standard forms of properties that are related to event-notification-distribution, listed in Table K-14.

Devices claiming support for this BIBB shall be capable of writing the full range of values as defined in Table K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

Devices claiming conformance to this BIBB are not required to support presentation and modification of objects and properties that are introduced in a Protocol_Revision newer than that claimed by the A device.

A device claiming support for AE-EAVM-A is interoperable with devices that support AE-N-I-B or AE-N-E-B.

K.3 Scheduling BIBBs

These BIBBs prescribe the BACnet capabilities required to interoperably perform the scheduling functions enumerated in Clause 22.2.1.3 for the BACnet devices defined therein.

K.3.1 BIBB - Scheduling-A (SCHED-A)

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.
The A device manipulates schedules and calendars on the B device. The A device is required to support the DS-RP-A and DS-WP-A BIBBs.

K.3.2 BIBB - Scheduling-Internal-B (SCHED-I-B)

The B device provides date and time scheduling of the values of specific properties of specific objects within the device. In addition to supporting the DS-RP-B and DS-WP-B BIBBs, each device claiming conformance to SCHED-I-B shall also be capable of possessing at least one Calendar and one Schedule object. Devices claiming conformance to SCHED-I-B shall also support either DM-TS-B or DM-UTC-B.

The Schedule object shall support a writable Weekly_Schedule property and at least 6 entries per day. The Schedule object shall support a non-empty Exception_Schedule property. If the List_Of_Object_Property_Reference is writable, the Priority_For_Writing property in the Schedule object shall also be writable.

K.3.3 BIBB - Scheduling-External-B (SCHED-E-B)

The B device provides date and time scheduling of the values of specific properties of specific objects in other devices. Devices at protocol_Revision 20 or higher, and claiming conformance to SCHED-E-B shall at a minimum support the writing of primitive datatypes BOOLEAN, REAL, Enumerated, and Unsigned32 to all standard properties of those datatypes in any standard objects. A Schedule object in the device shall support the writing of primitive datatype NULL to commandable standard properties in standard objects. This occurs when the value in the Relinquish_Default property is NULL and no schedule actions are in effect from the Weekly_Schedule and Exception_Schedule properties. Scheduling of other datatypes may also be supported.

The device shall support SCHED-I-B and DS-WP-A. The List_Of_Object_Property_References property shall support references to objects in external devices and be writable.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.3.4 BIBB - Scheduling-Readonly-B (SCHED-R-B)

The B device provides read-only Schedule object(s). Each device claiming conformance to SCHED-R-B shall be capable of possessing at least one Schedule object, support DS-RP-B and either DM-TS-B or DM-UTC-B. The Weekly_Schedule and Exception_Schedule properties of the Schedule object shall be read-only when accessed directly via BACnet services, but may be modifiable by other means.

This BIBB is primarily included in the BACnet standard to allow gateway devices to indicate support for exposing the content of schedules found in devices from other protocols.

K.3.5 BIBB - Scheduling-Advanced View and Modify-A (SCHED-AVM-A)

The A device manipulates schedules, calendars, and timers on the B device. The A device shall support the DM-OCD-A, DS-RP-A and DS-WP-A BIBBs.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

The A device uses ReadProperty to retrieve for presentation and WriteProperty to modify each of the Schedule, Calendar, and Timer properties listed below. The A device shall be capable of using the CreateObject and DeleteObject services to create and delete Schedule, Calendar, and Timer objects on the B device. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

Table K-25. Properties That SCHED-AVM-A Devices Shall Be Capable of Presenting and Modifying

Schedule	Calendar	Timer
Effective_Period Weekly_Schedule Exception_Schedule Schedule_Default List_Of_Object_Property_References Priority_For_Writing Out_Of_Service	Date_List	Present_Value Timer_State Timer_Running Default_Timeout State_Change_Values List_Of_Object_Property_References Priority_For_Writing Out_Of_Service

The A device shall support the creation, presentation and modification of all forms of the Weekly_Schedule, Exception_Schedule and Date_List properties with the following limitations. At a minimum, the A device shall be capable of handling Exception_Schedule properties with up to 255 entries, and 12 BACnetTimeValue tuples per entry, Weekly_Schedule properties with up to 6 entries per day, and Date_List properties with up to 32 entries.

Devices claiming support for this BIBB shall be capable of creating, deleting, presenting, and modifying Schedule objects that schedule any of the following types:

BOOLEAN, REAL, ENUMERATED, Unsigned32

and which may contain NULL values and shall be capable of changing the datatype that a Schedule object schedules. Schedule objects contain a number of properties that need to be consistent in the datatype of the values they contain. Devices claiming support for this BIBB shall be prepared to interact with, as well as allow display and modification of, Schedule objects that are self-inconsistent. A self-inconsistent Schedule object is one in which the scheduled values in the Weekly_Schedule, Exception_Schedule, and Schedule_Default properties are not all of the same datatype or in which the controlled objects are not all of the same datatype or are of a datatype different than the scheduled values.

Devices claiming support for this BIBB shall be capable of creating, deleting, presenting, and modifying Timer objects that write any of the following types:

BOOLEAN REAL, ENUMERATED, Unsigned32, NULL

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

and shall be capable of changing the datatype that a Timer object writes, and shall be capable of setting the 'No Value' option. Timer objects contain a number of properties that need to be consistent in the datatype of the values they contain. Devices claiming support for this BIBB shall be prepared to interact with, as well as allow display and modification of, Timer objects that are self-inconsistent. A self-inconsistent Timer object is one in which the values to write in the State_Change_Values property are not all of the same datatype or in which the controlled objects are not all of the same datatype or are of a datatype different than the values to be written.

The A device shall be capable of creating, deleting, presenting and modifying schedules in any B device regardless of the B device's claimed Protocol_Revision.

The A device shall be capable of creating, deleting, presenting and modifying schedule objects that do not contain an Exception_Schedule.

Devices claiming support for this BIBB shall be capable of providing times and values in the time/value pairs within the full range as defined in Tables K-4 and K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

A device claiming support for SCHED-AVM-A is interoperable with devices that support any of the B side schedule BIBBs.

Support for timer objects is not required for A side devices that claim a protocol revision less than 17.

K.3.6 BIBB - Scheduling-View and Modify-A (SCHED-VM-A)

The A device manipulates schedules, calendars, and timers on the B device. The A device shall support the DS-RP-A and DS-WP-A BIBBs.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

Device A shall be capable of using ReadProperty to retrieve for presentation and WriteProperty to modify each of the Schedule, Calendar, and Timer properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

Table K-26. Properties That SCHED-VM-A Devices Shall Be Capable of Presenting and Modifying

Schedule	Calendar	Timer
Effective_Period Weekly_Schedule Exception_Schedule	Date_List	Present_Value Timer_State Timer_Running Default_Timeout State_Change_Values

The A device shall support the presentation and modification of all forms of the Weekly_Schedule, Exception_Schedule and Date_List properties with the following limitations. At a minimum, the A device shall be capable of handling Exception_Schedule properties with up to 255 entries, and 12 BACnetTimeValue tuples per entry, Weekly_Schedule properties with up to 6 entries per day, and Date_List properties with up to 32 entries.

Devices claiming support for this BIBB shall be capable of presenting, and modifying Schedule objects that schedule any of the following types:

BOOLEAN, REAL, ENUMERATED, Unsigned32

and which may contain NULL values.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Devices claiming support for this BIBB shall be capable of presenting and modifying Timer objects that write any of the following types:

BOOLEAN, REAL, ENUMERATED, Unsigned32, NULL

and which may contain the 'No-Value' option.

The A device shall be capable of presenting and modifying schedules in any B devices regardless of the B device's claimed Protocol_Revision.

The A device shall be capable of presenting and modifying schedule objects that do not contain an Exception_Schedule.

Devices claiming support for this BIBB shall be capable of providing times and values in the time/value pairs within the full range as defined in Table K-6.

A device claiming support for SCHED-VM-A is interoperable with devices that support any of the B-side schedule BIBBs.

Support for timer objects is not required for A side devices that claim a protocol revision less than 17.

K.3.7 BIBB - Scheduling-Weekly Schedule-A (SCHED-WS-A)

The A device manipulates the weekly schedule portion of schedules on the B device. The A device shall support the DS-RP-A and DS-WP-A BIBBs.

BACnet Service	Initiate	Execute
ReadProperty	x	
WriteProperty	x	

The A device shall be capable of using ReadProperty to retrieve for presentation and WriteProperty to modify each of the Schedule properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

Table K-27. Properties That SCHED-WS-A Devices Shall Be Capable of Presenting and Modifying

Schedule
Weekly_Schedule
Schedule_Default

The A device shall support the presentation and modification of all forms of the Weekly_Schedule, property with the following limitations. At a minimum, the A device shall be capable of handling Weekly_Schedule properties with up to 6 entries per day.

Devices claiming support for this BIBB shall be capable of presenting, and modifying Schedule objects that schedule any of the following types:

BOOLEAN, REAL, ENUMERATED, Unsigned32

and which may contain NULL values.

The A device shall be capable of presenting and modifying Schedule objects of the forms defined in Protocol_Revision 0 and Protocol_Revision 4.

K.3.8 BIBB - Scheduling-Weekly Schedule-Internal-B (SCHED-WS-I-B)

The B device provides weekly scheduling of the values of specific properties of specific objects within the device via Schedule objects that do not contain Exception_Schedules. In addition to supporting the DS-RP-B and DS-WP-B BIBBs, each device claiming conformance to SCHED-WS-I-B shall also be capable of possessing at least one Schedule object. Devices claiming conformance to SCHED-WS-I-B shall also support either DM-TS-B or DM-UTC-B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
ReadProperty		x
WriteProperty		x
TimeSynchronization		x
UTCTimeSynchronization		x

The Schedule object shall support at least 6 entries per day in the Weekly_Schedule property. The schedule shall support the scheduling of BACnetBinaryPV values. The Priority_For_Writing property in the Schedule object shall be writable.

K.3.9 BIBB - Scheduling-Timer-Internal-B (SCHED-TMR-I-B)

The B device provides timed modification of the values of specific properties of specific objects within the device. In addition to supporting the DS-RP-B and DS-WP-B BIBBs, each device claiming conformance to SCHED-TMR-I-B shall also be capable of possessing at least one Timer object.

The Present_Value, the Timer_State, and the Timer_Running properties in the Timer object shall be writable.

The Default_Timeout property in the Timer object shall be present and writable. The Timer object shall support a writable State_Change_Values property. The Priority_For_Writing property in the Timer object shall be writable.

K.3.10 BIBB - Scheduling-Timer-External-B (SCHED-TMR-E-B)

The B device provides timed modification of the values of specific properties of specific objects in other devices. Devices claiming conformance to SCHED-TMR-E-B shall also support SCHED-TMR-I-B and DS-WP-A. The List_Of_Object_Property_References property of the Timer object shall support references to objects in external devices and be writable.

K.4 Trending BIBBs

These BIBBs prescribe the BACnet capabilities required to interoperably perform the trending functions enumerated in Clause 22.2.1.4 for the BACnet devices defined therein.

K.4.1 BIBB - Trending-Viewing and Modifying Trends-A (T-VMT-A)

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.

The A device displays trend data from the B device and manipulates trend log collection parameters in the B device.

BACnet Service	Initiate	Execute
ReadRange	x	

K.4.2 BIBB - Trending-Viewing and Modifying Trends-Internal-B (T-VMT-I-B)

The B device collects the trend log data records in an internal buffer. Each device claiming conformance to T-VMT-I-B shall be able to support at least one Trend Log object.

BACnet Service	Initiate	Execute
ReadRange		x

K.4.3 BIBB - Trending-Viewing and Modifying Trends-External-B (T-VMT-E-B)

The B device is capable of trending properties of objects contained in other devices. The B device shall support T-VMT-I-B and DS-RP-A. The Log_Interval and Log_DeviceObjectProperty properties shall be writable.

The Trend Log objects shall be capable of trending REAL, Unsigned, INTEGER, BOOLEAN, Bit String, Enumerated and NULL values.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

K.4.4 BIBB - Trending-Automated Trend Retrieval-A (T-ATR-A)

The A device responds to a notification that a trend log is ready with new data and acquires the new data from the log.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x
ReadRange	x	

Devices claiming conformance to T-ATR-A shall be able to process BUFFER_READY event notifications generated by Trend Log objects and Event Enrollment objects.

K.4.5 BIBB - Trending-Automated Trend Retrieval-B (T-ATR-B)

The B device notifies the A device that a trending buffer has acquired a predetermined number of data samples using the BUFFER_READY event algorithm either intrinsically in the Trend Log object or algorithmically using an Event Enrollment object.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	
UnconfirmedEventNotification	x	
ReadRange		x

Devices claiming conformance to T-ATR-B shall support the Trend Log object.

K.4.6 BIBB - Trending-Viewing and Modifying Multiple Values-A (T-VMMV-A)

This BIBB has been deprecated and is included solely for historical purposes. This BIBB should not be used when describing the functionality of BACnet devices.

The A device displays data from a Trend Log Multiple object in the B device and manipulates Trend Log Multiple object collection parameters in the B device.

BACnet Service	Initiate	Execute
ReadRange	x	

K.4.7 BIBB - Trending-Viewing and Modifying Multiple Values-Internal-B (T-VMMV-I-B)

The B device collects the multiple-data log records in an internal buffer. Each device claiming conformance to T-VMMV-I-B shall be able to support at least one Trend Log Multiple object.

BACnet Service	Initiate	Execute
ReadRange		x

K.4.8 BIBB - Trending-Viewing and Modifying Multiple Values-External-B (T-VMMV-E-B)

The B device is capable of logging multiple properties of multiple objects contained in other devices. The B device shall support T-VMMV-I-B and DS-RPM-A. The Log_Interval and Log_DeviceObjectProperty properties shall be writable.

K.4.9 BIBB - Trending-Automated Multiple Value Retrieval-A (T-AMVR-A)

The A device responds to a notification that a Trend Log Multiple object is ready with new data and acquires the new data from the log.

BACnet Service	Initiate	Execute
ConfirmedEventNotification		x
UnconfirmedEventNotification		x
ReadRange	x	

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Devices claiming conformance to T-AMVR-A shall be able to process BUFFER_READY event notifications generated by Trend Log Multiple objects and Event Enrollment objects.

K.4.10 BIBB - Trending-Automated Multiple Value Retrieval-B (T-AMVR-B)

The B device notifies the A device that a Trend Log Multiple object's buffer has acquired a predetermined number of data samples using the BUFFER_READY event algorithm either intrinsically in the Trend Log Multiple object or algorithmically using an Event Enrollment object.

BACnet Service	Initiate	Execute
ConfirmedEventNotification	x	
UnconfirmedEventNotification	x	
ReadRange		x

Devices claiming conformance to T-AMVR-B shall support the Trend Log Multiple object.

K.4.11 BIBB - Trending-View-A (T-V-A)

The A device displays trend data from the B device. Within the context of this BIBB, the term "trend object" shall refer to both Trend Log and Trend Log Multiple objects.

BACnet Service	Initiate	Execute
ReadRange	x	

The A device uses ReadRange to retrieve and display trend object Log_Buffer properties.

Devices claiming support for this BIBB shall be capable of presenting data from trend objects for the following types of data:

BOOLEAN, REAL, ENUMERATED, Unsigned32, INTEGER, BIT STRING, NULL

Devices claiming conformance to a Protocol_Revision less than 7 are not required to support these interactions with Trend Log Multiple objects. The A device need not be capable of interoperating with Trend Logs of the form defined in Protocol_Revision 1.

A device claiming support for T-V-A is interoperable with devices that support T-VMT-I-B.

K.4.12 BIBB - Trending-Advanced View and Modify-A (T-AVM-A)

The A device displays trend data from the B device and manipulates trend log collection parameters in the B device. Devices claiming support for this BIBB shall support DS-RP-A and DS-WP-A. Within the context of this BIBB, the term "trend object" shall refer to both Trend Log and Trend Log Multiple objects.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
ReadRange	x	
WriteProperty	x	

The A device shall be capable of using ReadRange to retrieve and display trend object Log_Buffer properties, ReadProperty to retrieve and display trend object properties, WriteProperty to modify trend object properties, and CreateObject and DeleteObject to create and delete trend objects, Event Enrollment and Notification Class objects. The properties that the A device shall be capable of reading and writing are listed below. Device A may use alternate services where support for execution of the alternate service is supported by the B Device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Table K-28. Trend Object Properties That T-AVM-A Devices Shall Be Capable of Presenting and Modifying

Enable	Stop_When_Full
Start_Time	Buffer_Size
Stop_Time	Record_Count
Log_DeviceObjectProperty	Total_Record_Count (retrieve only)
Logging_Type	Notification_Threshold
Log_Interval	Last_Notify_Record (retrieve only)
Align_Intervals	Event_State (retrieve only)
Interval_Offset	Notification_Class
COV_Resubscription_Interval	Event_Enable
Client_COV_Increment	Event_Time_Stamps (retrieve only)

In addition, devices claiming support for T-AVM-A shall be capable of creating and configuring Event Enrollment objects to monitor trend objects using the BUFFER_READY algorithm. The A device shall also be capable of creating and configuring Notification Class objects (as described in AE-AVM-A) for setup of Automated Trend Retrieval.

Devices claiming support for this BIBB shall be capable of presenting trend object data of the following types:

BOOLEAN, REAL, ENUMERATED, Unsigned32, INTEGER, BIT STRING, NULL

Devices claiming support for this BIBB shall be capable of writing values within the full range as defined in Tables K-4 and K-6.

A device claiming support for T-AVM-A is interoperable with devices that support T-VMT-I-B or T-VMMV-I-B.

Devices claiming conformance to a Protocol_Revision less than 7, are not required to support these interactions with Trend Log Multiple objects nor properties added to the Trend Log object in Protocol_Revision 7. The A device need not be capable of interoperating with Trend Logs of the form defined in Protocol_Revision 1.

K.4.13 BIBB - Trending-Archival-A (T-A-A)

The A device archives trend data from Trend Log and Trend Log Multiple objects in the B device. The A device shall support T-ATR-A and T-AMVR-A and shall be capable of using the BUFFER_READY notifications to ensure that trend data is retrieved using ReadRange and archived before the trend data is removed from the Log_Buffer property due to the addition of newly collected samples. The archived data shall be stored in non-volatile storage for future access.

A device claiming support for T-A-A is interoperable with devices that support T-ATR-B or T-AMVR-B.

Devices claiming conformance to a Protocol_Revision less than 7, are not required to support these interactions with Trend Log Multiple objects and thus do not need to support T-AMVR-A.

K.5 Device Management BIBBs

These device management BIBBs prescribe the BACnet capabilities required to interoperably perform the device management functions enumerated in Clause 22.2.1.5 for the BACnet devices defined therein.

K.5.1 BIBB - Device Management-Dynamic Device Binding-A (DM-DDB-A)

The A device seeks information about device attributes of other devices and interprets device announcements.

BACnet Service	Initiate	Execute
Who-Is	x	
I-Am		x

K.5.2 BIBB - Device Management-Dynamic Device Binding-B (DM-DDB-B)

The B device provides information about its device attributes and responds to requests to identify itself.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
Who-Is		x
I-Am	x	

K.5.3 BIBB - Device Management-Dynamic Object Binding-A (DM-DOB-A)

The A device seeks address information about objects.

BACnet Service	Initiate	Execute
Who-Has	x	
I-Have		x

K.5.4 BIBB - Device Management-Dynamic Object Binding-B (DM-DOB-B)

The B device provides address information about its objects upon request.

BACnet Service	Initiate	Execute
Who-Has		x
I-Have	x	

K.5.5 BIBB - Device Management-DeviceCommunicationControl-A (DM-DCC-A)

The A device exercises communication control over the B device.

BACnet Service	Initiate	Execute
DeviceCommunicationControl	x	

Support for requests of a limited duration is required, and support for requests of an indefinite duration is optional.

K.5.6 BIBB - Device Management-DeviceCommunicationControl-B (DM-DCC-B)

The B device responds to communication control exercised by the A device.

BACnet Service	Initiate	Execute
DeviceCommunicationControl		x

Support for requests of a limited duration is required, and support for requests of an indefinite duration is optional.

K.5.9 BIBB - Device Management-Text Message-A (DM-TM-A)

The A device initiates the transmission of text messages. The interpretation and subsequent processing of the messages is a local matter.

BACnet Service	Initiate	Execute
ConfirmedTextMessage	x[1]	
UnconfirmedTextMessage	x[1]	

[1] The A device shall support initiation of at least one of these services.

K.5.10 BIBB - Device Management-Text Message-B (DM-TM-B)

The B device processes the text messages.

BACnet Service	Initiate	Execute
ConfirmedTextMessage		x
UnconfirmedTextMessage		x

K.5.11 BIBB - Device Management-TimeSynchronization-A (DM-TS-A)

The A device provides time synchronization to B devices. The time parameter contained in the service request contains the date and time as determined by the clock in the device issuing the service request. Normally this will be "local time," i.e. the time in the local time zone corrected for daylight savings time as appropriate.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

BACnet Service	Initiate	Execute
TimeSynchronization	x	

Devices claiming conformance to DM-TS-A shall support the Time_Synchronization_Recipients and UTC_Time_Synchronization_Recipients properties of the Device object.

K.5.12 BIBB - Device Management-TimeSynchronization-B (DM-TS-B)

The B device interprets time synchronization messages from the A device.

BACnet Service	Initiate	Execute
TimeSynchronization		x

Devices claiming conformance to DM-TS-B shall support the Local_Time and Local_Date properties of the Device object.

K.5.13 BIBB - Device Management-UTCTimeSynchronization-A (DM-UTC-A)

The A device provides time synchronization to B devices. The time parameter contained in the service request contains "Coordinated Universal Time" (UTC). For all practical purposes, UTC is synonymous with Greenwich Mean Time, the time at the zero or Greenwich meridian.

BACnet Service	Initiate	Execute
UTCTimeSynchronization	x	

Devices claiming conformance to DM-UTC-A shall support the Time_Synchronization_Recipients and UTC_Time_Synchronization_Recipients properties of the Device object.

K.5.14 BIBB - Device Management-UTCTimeSynchronization-B (DM-UTC-B)

The B device interprets time synchronization messages from the A device.

BACnet Service	Initiate	Execute
UTCTimeSynchronization		x

Devices claiming conformance to DM-UTC-B shall support the Local_Time, Local_Date, UTC_Offset, and Daylight_Saving_Status properties of the Device object.

K.5.15 BIBB - Device Management-ReinitializeDevice-A (DM-RD-A)

The A device is authorized to reinitialize the B device.

BACnet Service	Initiate	Execute
ReinitializeDevice	x	

Devices claiming conformance to DM-RD-A shall be able to initiate ReinitializeDevice requests containing the Password parameter. Devices claiming conformance to DM-RD-A are only required to support the ACTIVATE_CHANGES, WARMSTART and COLDSTART service choices.

K.5.16 BIBB - Device Management-ReinitializeDevice-B (DM-RD-B)

The B device performs reinitialization requests from the A device. The optional password field shall be supported.

BACnet Service	Initiate	Execute
ReinitializeDevice		x

Devices claiming conformance to DM-RD-B are only required to support the WARMSTART and COLDSTART service choices. If the device supports a configurable Network Port Object, then it shall also support the restart choice ACTIVATE_CHANGES.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.5.17 BIBB - Device Management-Backup and Restore-A (DM-BR-A)

The A device reads the files that contain the configuration of the B device and writes those files to the B device should it need to be restored to its previously backed-up state.

BACnet Service	Initiate	Execute
AtomicReadFile	x	
AtomicWriteFile	x	
CreateObject	x	
ReinitializeDevice	x	

Devices claiming conformance to DM-BR-A are required to support COLDSTART, WARMSTART, STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, and ABORTRESTORE choices of the ReinitializeDevice service. In addition, devises claiming conformance to DM-BR-A shall support the device A capabilities as described in Clause 19.1.

K.5.18 BIBB - Device Management-Backup and Restore-B (DM-BR-B)

The B device provides its configuration file to the A device and allows the A device to write this file to recover its configuration in the event of a failure.

BACnet Service	Initiate	Execute
AtomicReadFile		x
AtomicWriteFile		x
ReinitializeDevice		x

Devices claiming conformance to DM-BR-B are required to support COLDSTART, WARMSTART, STARTBACKUP, ENDBACKUP, STARTRESTORE, ENDRESTORE, and ABORTRESTORE choices of the ReinitializeDevice service. In addition, devices claiming conformance to DM-BR-B shall support the device B capabilities as described in Clause 19.1. Once a Restore procedure has been initiated on the device, the Read_Only property of configuration File objects shall contain the value FALSE and the File_Size property of the configuration File objects shall be writable if the size of the configuration file can change based on the device's configuration.

If the configuration file objects are not guaranteed to exist once a Restore procedure has been initiated, then the device shall support execution of the CreateObject service.

K.5.19 BIBB - Device Management-Restart-A (DM-R-A)

The A device processes restart notifications.

BACnet Service	Initiate	Execute
UnconfirmedCOVNotification		x

K.5.20 BIBB - Device Management-Restart-B (DM-R-B)

The B device informs the A device(s) each time it restarts.

BACnet Service	Initiate	Execute
UnconfirmedCOVNotification	x	

Devices claiming conformance to DM-R-B shall support the Last_Restart_Reason, Restart_Notification_Recipients, and Time_Of_Device_Restart properties of the Device object.

K.5.21 BIBB - Device Management-List Manipulation-A (DM-LM-A)

Many BACnet object types have properties that are lists of a particular datatype. The A device can add and remove list elements in properties of objects in the B device.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
AddListElement	x	
RemoveListElement	x	

K.5.22 BIBB - Device Management-List Manipulation-B (DM-LM-B)

The B device responds to requests to add or remove list elements.

BACnet Service	Initiate	Execute
AddListElement		x
RemoveListElement		x

K.5.23 BIBB - Device Management-Object Creation and Deletion-A (DM-OCD-A)

BACnet allows object instances to be dynamically created and deleted. The A device is capable of dynamically creating and deleting object instances of at least one object type.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	

K.5.24 BIBB - Device Management-Object Creation and Deletion-B (DM-OCD-B)

The B device creates and deletes object instances based on requests from the A device. The object types whose dynamic creation and deletion is supported shall be enumerated in the Standard Object Types Supported section of device B's PICS.

BACnet Service	Initiate	Execute
CreateObject		x
DeleteObject		x

K.5.25 BIBB - Device Management-Virtual Terminal-A (DM-VT-A)

The A device initiates a virtual terminal session and exchanges data with the B device.

BACnet Service	Initiate	Execute
VT-Open	x	
VT-Close	x	x
VT-Data	x	x

K.5.26 BIBB - Device Management-Virtual Terminal-B (DM-VT-B)

The B devices permits the establishment of a virtual terminal session and exchanges data with the A device.

BACnet Service	Initiate	Execute
VT-Open		x
VT-Close	x	x
VT-Data	x	x

K.5.27 BIBB - Device Management-Automatic Network Mapping-A (DM-ANM-A)

The A device finds all devices currently connected to the BACnet internetwork that support DM-DDB-B and presents the list of those devices to the user. A device claiming support for this BIBB shall support DM-DDB-A.

The A device is not required to report the presence of devices located on the far side of a non-connected PTP link.

A device claiming support for DM-ANM-A is interoperable with devices that support DM-DDB-B.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.5.28 BIBB - Device Management-Automatic Device Mapping-A (DM-ADM-A)

The A device is capable of determining and presenting a list of all objects contained in any BACnet device. Devices claiming support for this BIBB shall also support DS-RP-A to retrieve and display the Object_Name property of any object in any BACnet device. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

A device claiming support for DM-ADM-A is interoperable with all BACnet devices.

K.5.29 BIBB - Device Management-Automatic Time Synchronization-A (DM-ATS-A)

The A device provides periodic time synchronization to B devices. In order to support all types of BACnet devices, the A device shall be capable of periodically sending TimeSynchronization and UTCTimeSynchronization services to recipients listed in the A device's Time_Synchronization_Recipients and UTC_Time_Synchronization_Recipients properties.

BACnet Service	Initiate	Execute
TimeSynchronization	x	
UTCTimeSynchronization	x	

Devices claiming conformance to DM-ATS-A shall support non-empty Time_Synchronization_Recipients and UTC_Time_Synchronization_Recipients properties in its Device object and shall support all forms of the BACnetRecipient in both properties.

A device claiming support for DM-ATS-A is interoperable with devices that support DM-TS-B or DM-UTC-B.

K.5.30 BIBB - Device Management-Manual Time Synchronization-A (DM-MTS-A)

The A device provides time synchronization to B devices at the request of the operator. In order to support all types of BACnet devices, the A device shall be capable of sending both TimeSynchronization and UTCTimeSynchronization services to any, or all, BACnet devices in the BACnet internetwork.

BACnet Service	Initiate	Execute
TimeSynchronization	x	
UTCTimeSynchronization	x	

A device claiming support for DM-MTS-A is interoperable with devices that support DM-TS-B or DM-UTC-B.

K.5.31 BIBB - Device Management-Slave Proxy-View and Modify-A (DM-SP-VM-A)

The A device displays and modifies the slave proxy related properties in a slave proxy device.

BACnet Service	Initiate	Execute
ReadProperty	x	
ReadRange	x	
AddListElement	x	
RemoveListElement	x	
WriteProperty	x	

Devices claiming conformance to DM-SP-VM-A shall be able to read and present the Slave_Proxy_Enable, Manual_Slave_Address_Binding, Auto_Slave_Discovery and the Slave_Address_Binding properties of Device and Network Port object types (support for interaction with Network Port object types is only required for devices claiming conformance to a Protocol_Revision of 17 or greater.)

When a device is proxying for a large number of MS/TP slave devices, the Manual_Slave_Address_Binding and Slave_Address_Binding properties can contain very large values. For this reason, the A device shall support initiation of the ReadRange service so as to be able to retrieve large values for these properties.

The A device shall be capable of configuring the Slave_Proxy_Enable, Manual_Slave_Address_Binding, Auto_Slave_Discovery, and Slave_Address_Binding properties.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

A device claiming support for DM-SP-VM-A is interoperable with devices that support DM-SP-B.

K.5.32 BIBB - Device Management-Slave Proxy-B (DM-SP-B)

The B device implements the slave proxy functionality and provides I-Am messages on behalf of MS/TP slave devices.

BACnet Service	Initiate	Execute
ReadProperty	x	x
ReadRange		x
AddListElement		x
RemoveListElement		x
WriteProperty		x
Who-Is		x
I-Am	x	

Devices claiming conformance to DM-SP-B shall support the Slave_Proxy_Enable, Manual_Slave_Address_Binding, Auto_Slave_Discovery and the Slave_Address_Binding properties. Slave proxies shall be capable of performing the slave proxy function on any or all of the directly connected MS/TP networks.

Devices claiming conformance to this BIBB shall be capable of proxying for at least 32 MS/TP slaves devices per directly connected MS/TP network.

When the B device is proxying for a large number of MS/TP slave devices, the Manual_Slave_Address_Binding and Slave_Address_Binding properties can contain very large values. For this reason, the B device shall support execution of the ReadRange service.

A device claiming support for DM-SP-B is interoperable with devices that support DM-SP-VM-A and with MS/TP slave devices.

K.5.33 BIBB - Device Management- Lighting Output Management-A (DM-LOM-A)

BACnet allows lighting object instances to be dynamically created and deleted. The A device shall be able to dynamically create and delete Lighting Output, Binary Lighting Output, and Channel object types supported by the B device.

BACnet Service	Initiate	Execute
CreateObject	x	
DeleteObject	x	

K.5.34 BIBB - Device Management-Dynamic Device Assignment-A (DM-DDA-A)

The A device assigns other devices their device attributes and responds to requests for device attributes.

BACnet Service	Initiate	Execute
Who-Am-I		x
You-Are	x	
Who-Is	x	

K.5.35 BIBB - Device Management-Dynamic Device Assignment-B (DM-DDA-B)

The B device seeks information about device attributes of itself and interprets device assignments.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
Who-Am-I	x	
You-Are		x
Who-Is		x
I-Am	x	

K.6 Network Management BIBBs

The network management BIBBs prescribe the BACnet capabilities required to interoperably perform network management functions described in Clause 22.2.1.5.

K.6.1 BIBB - Network Management-Connection Establishment-A (NM-CE-A)

The A device commands a half-router to establish and terminate connections as needed for communication with other devices.

BACnet Network Layer Message	Initiate	Execute
Establish-Connection-To-Network	x	
Disconnect-Connection-To-Network	x	

K.6.2 BIBB - Network Management-Connection Establishment-B (NM-CE-B)

The B device executes commands to establish and terminate connections as needed.

BACnet Network Layer Message	Initiate	Execute
Establish-Connection-To-Network		x
Disconnect-Connection-To-Network		x

K.6.3 BIBB - Network Management-Router Configuration-A (NM-RC-A)

The A device is capable of querying, changing and verifying the configuration of routers and half-routers.

The A device is capable of creating and deleting Network Port objects; reading all standard properties of the Network Port object type; and writing all standard properties of the Network Port object type except those defined as read-only by this standard.

BACnet Network Layer Message	Initiate	Execute
Who-Is-Router-To-Network	x	
I-Am-Router-To-Network		x
I-Could-Be-Router-To-Network		x
Establish-Connection-To-Network	x	
Disconnect-Connection-To-Network	x	

Devices claiming a Protocol_Revision less than 17 and conformance to this BIBB are not required to be able to create, delete or modify Network Port objects.

K.6.4 BIBB - Network Management-Router Configuration-B (NM-RC-B)

The B device responds to router management commands and shall meet the requirements for BACnet Routers as stated in Clause 6. Devices claiming conformance to this BIBB shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Network Layer Message	Initiate	Execute
Who-Is-Router-To-Network	x	x
I-Am-Router-To-Network	x	x
Reject-Message-To-Network	x	x
Router-Busy-To-Network	x	x
Router-Available-To-Network	x	x
Network-Number-Is	x	x
What-Is-Network-Number		x

BACnet Service	Initiate	Execute
Read-Property		x
Write-Property		x

Devices claiming a Protocol_Revision less than 11 and conformance to this BIBB are not required to support What-Is-Network-Number nor Network-Number-Is.

Devices claiming a Protocol_Revision less than 17 and conformance to this BIBB are not required to support WriteProperty nor the Network Port object type.

Devices shall meet the requirements in Clause 6 for the forwarding of Router-Busy-To-Network messages. The initiation of Router-Busy-To-Network messages for other conditions remains optional.

K.6.5 BIBB - Network Management-BBMD Configuration-A (NM-BBMDC-A)

The A device is able to query and change the configuration of BBMDs.

BACnet Network Layer Message	Initiate	Execute
Write-Broadcast-Distribution-Table	x	
Read-Broadcast-Distribution-Table	x	
Read-Foreign-Device-Table	x	
Delete-Foreign-Device-Table-Entry	x	

Devices claiming conformance to this BIBB are required to initiate Write-Broadcast-Distribution-Table for interoperability with devices implementing Protocol_Revisions older than 17.

K.6.6 BIBB - Network Management-BBMD Configuration-B (NM-BBMDC-B)

The B device responds to BBMD management commands and shall meet the requirements for BBMDs as stated in Annex J.

- Supports 2-hop distribution (support for 1-hop is optional).
- Supports a minimum of five peer BBMDs.
- Supports a minimum of five simultaneous foreign device connections.
- At a minimum, supports foreign device registration lifetimes in the range 30 seconds – 9 hours.

Devices claiming conformance to this BIBB shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Virtual Link Layer Message	Initiate	Execute
Write-Broadcast-Distribution-Table		x
Read-Broadcast-Distribution-Table		x
Read-Foreign-Device-Table		x
Delete-Foreign-Device-Table-Entry		x
Register-Foreign-Device		x
Forwarded-NPDU	x	x
Distribute-Broadcast-To-Network		x
Original-Unicast-NPDU	x	x
Original-Broadcast-NPDU	x	x

K.6.7 BIBB - Network Management-Foreign Device Registration-A (NM-FDR-A)

Devices claiming conformance to this BIBB shall be able to register as foreign device.

- Supports, at a minimum, foreign device registration lifetimes in the range 30 seconds – 9 hours.

BACnet Virtual Link Layer Message	Initiate	Execute
Register-Foreign-Device	x	
Distribute-Broadcast-To-Network	x	
Forwarded-NPDU		x
Original-Unicast-NPDU	x	x

K.6.8 BIBB - Network Management-Secure Connect Hub-B (NM-SCH-B)

The B device supports at least one BACnet/SC network port and is capable of supporting the BACnet/SC hub function on at least one of these ports. The B device is capable of accepting hub connections and performing the BACnet/SC hub function.

Devices claiming conformance to this BIBB shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

BACnet/SC BVLL Function	Initiate	Execute
Encapsulated-NPDU	x	x
Advertisement	x	x
Advertisement-Solicitation		x
Connect-Request		x
Connect-Accept	x	
Disconnect-Request	x	x
Disconnect-ACK	x	x
Heartbeat-Request		x
Heartbeat-ACK	x	

K.6.9 BIBB - Network Management-Secure Connect Direct Connect -A (NM-SCDC-A)

The A device supports at least one BACnet/SC network port that implements the node switch and is able to initiate at least one direct connection.

Devices claiming conformance to this BIBB shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet/SC BVLL Function	Initiate	Execute
Encapsulated-NPDU	x	x
Address-Resolution	x	
Address-Resolution-ACK		x
Advertisement	x	x
Advertisement-Solicitation	x	x
Connect-Request	x	
Connect-Accept		x
Disconnect-Request	x	x
Disconnect-ACK	x	x
Heartbeat-Request	x	x
Heartbeat-ACK	x	x

K.6.10 BIBB - Network Management-Secure Connect Direct Connect - B (NM-SCDC-B)

The B device supports at least one BACnet/SC network port that implements the node switch and is able to accept at least one direct connection.

Devices claiming conformance to this BIBB shall support the following BACnet/SC BVLL functions.

BACnet/SC BVLL Function	Initiate	Execute
Encapsulated-NPDU	x	x
Address-Resolution		x
Address-Resolution-ACK	x	
Advertisement	x	x
Advertisement-Solicitation	x	x
Connect-Request		x
Connect-Accept	x	
Disconnect-Request	x	x
Disconnect-ACK	x	x
Heartbeat-Request		x
Heartbeat-ACK	x	

K.7 Gateway BIBBs

K.7.1 BIBB - Gateway-Virtual Network-B (GW-VN-B)

The B device provides access to data and functionality in non-BACnet devices. The B device models the devices as a collection of virtual BACnet devices and appears to the BACnet network as a router as described in Annex H. The B device shall, at a minimum, support NM-RC-B and DS-RP-B.

The gateway and each virtual BACnet device shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

K.7.2 BIBB - Gateway-Embedded Objects-B (GW-EO-B)

The B device provides access to data and functionality in non-BACnet devices. The B device includes the data and functionality of the other devices through BACnet objects and services within the B device.

The B device shall support DS-RP-B.

The gateway shall meet the minimum requirements for a BACnet device as described by this standard and specifically by Clause 22.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

K.8 Audit Reporting BIBBs

K.8.1 BIBB - Audit Reporting-Logging- A (AR-L-A)

The A device is an audit logger. It logs received audit notifications from operation sources and operation targets for merging into its Audit Logs and forwards notifications to parent audit loggers.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x	x
UnconfirmedAuditNotification	x	x
ReadRange		x
AuditLogQuery		x

The audit logger receives audit event notifications and places them into local Audit Log objects.

The audit logger performs all functions of an audit logger as described in Clause 19.6.

Devices claiming conformance to this BIBB are interoperable with devices claiming conformance to AR-R-B.

K.8.2 BIBB - Audit Reporting-Reporter-B (AR-R-B)

The B device generates audit notifications and supports Audit Reporter objects. It is an operation target which generates audit notifications for local objects, and, if it performs actions, it is an operation source which generates audit notifications for its actions. See Clause 19.6 for more information on audit logging.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x[1]	
UnconfirmedAuditNotification	x[1]	

[1] At least one of these shall be supported.

A device claiming this BIBB supports modification of its audit reporting configuration. Specifically, the following Audit Reporter properties shall be writable:

Table K-29. Writable Audit Reporter Properties

Audit_Level
Auditable_Operations
Audit_Priority_Filter

K.8.3 BIBB - Audit Reporting-Reporter-Simple-B (AR-R-S-B)

The B device generates audit notifications. It is an operation target which generates audit notifications for local objects, and, if it performs actions, it is an operation source which generates audit notifications for its actions. See Clause 19.6 for more information on audit logging.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x[1]	
UnconfirmedAuditNotification	x[1]	

[1] At least one of these shall be supported.

K.8.4 BIBB - Audit Reporting-Forwarder-B (AR-F-B)

The B device is an audit logger which receives audit notifications and forwards them to parent audit loggers. See Clause 19.6 for more information on audit logging.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x	x
UnconfirmedAuditNotification	x	x
ReadRange		x

K.8.5 BIBB - Audit Reporting-View-A (AR-V-A)

The A device queries audit loggers using AuditLogQuery or ReadRange and presents query results to the user.

BACnet Service	Initiate	Execute
AuditLogQuery	x[1]	
ReadRange	x[1]	

[1] At least one of these shall be supported.

K.8.6 BIBB - Audit Reporting-Advanced View and Modify-A (AR-AVM-A)

Device A configures auditing in all standard objects in Device B. Device A shall support DS-RP-A, DS-WP-A, DM-OCD-A, and AR-V-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any auditing related properties. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Audit Reporter and Audit Log objects in the B device.

BACnet Service	Initiate	Execute
AuditLogQuery	x	
ReadRange	x	
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Table K-30. Auditing Related Properties

Audit_Level
Auditable_Operations
Audit_Priority_Filter

The A device is able to use ReadProperty to retrieve and present all standard properties of the Audit Reporter and Audit Log object types, except those listed in Table K-30.

The A device is able to use WriteProperty to modify any standard property of the Audit Reporter and Audit Log object types where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer).

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

ANNEX L - DESCRIPTIONS AND PROFILES OF STANDARDIZED BACnet DEVICES (NORMATIVE)

(This annex is part of this standard and is required for its use.)

This annex provides descriptions of "standardized" types of BACnet devices. Any device that implements all the required BACnet capabilities for a particular device type and interoperability area may claim to be a device of that particular type. Devices may also provide additional capabilities and shall indicate these capabilities in their PICS.

BACnet device profiles are categorized into families:

- Operator Interfaces. This family is composed of B-XAWS, B-AWS, B-OWS, and B-OD.
- Lighting Operator Interfaces. This family is composed of B-XAWS, B-ALWS, and B-LOD.
- Life Safety Operator Interfaces. This family is composed of B-ALSWS, B-LSWS, and B-LSAP.
- Access Control Operator Interfaces. This family is composed of B-XAWS, B-AACWS, B-ACWS, and B-ACSD.
- Elevator Operator Interfaces. This family is composed of B-XAWS, B-AEWS, B-EWS, and B-ED.
- Lighting Control Stations. This family is composed of B-ALCS and B-LCS.
- Controllers. This family is composed of B-BC, B-AAC, B-ASC, B-SA, and B-SS.
- Lighting Controllers. This family is composed of B-LS and B-LD.
- Life Safety Controllers. This family is composed of B-ALSC and B-LSC.
- Access Control Controllers. This family is composed of B-AACC and B-ACC.
- Elevator Controllers. This family is composed of B-AEC, B-EC, and B-EM.
- Miscellaneous. This family is composed of B-RTR, B-GW, B-BBMD, B-ACDC, B-ACCR, and B-SCHUB.

Devices may claim to be multiple device types. For example, a device may claim to be both a B-BC and the B-RTR. Devices that claim multiple device profiles shall only combine capabilities from different device families, with the exception that multiple profiles may be selected from the Miscellaneous family. For example, a device may claim the B-BC, B-RTR, and B-BBMD profiles, but a device may not claim both the B-BC and B-SS profiles.

The B-GENERAL device profile is not included in any of the profile families and is never claimed in conjunction with any other device profile, except those from the Miscellaneous family.

L.1 Operator Interface Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area. The B-XAWS is not shown in this table and is described in Clause L.1.1.

Data Sharing

B-AWS	B-OWS	B-OD
DS-RP-A,B	DS-RP-A,B	DS-RP-A,B
DS-RPM-A	DS-RPM-A	
DS-WP-A	DS-WP-A	DS-WP-A
DS-WPM-A	DS-WPM-A	
DS-AV-A	DS-V-A	DS-V-A
DS-AM-A	DS-M-A	DS-M-A

Alarm & Event Management

B-AWS	B-OWS	B-OD
AE-N-A	AE-N-A	AE-N-A
AE-ACK-A	AE-ACK-A	
AE-AS-A	AE-AS-A	
AE-AVM-A	AE-VM-A	
AE-AVN-A	AE-VN-A	AE-VN-A
AE-ELVM-A[1]		

[1] Not required for devices claiming conformance to a Protocol_Revision less than 7

Scheduling

B-AWS	B-OWS	B-OD
SCHED-AVM-A	SCHED-VM-A	

Trending

B-AWS	B-OWS	B-OD
T-AVM-A	T-V-A	

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device & Network Management

B-AWS	B-OWS	B-OD
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-A,B
DM-ANM-A		
DM-ADM-A		
DM-DOB-B	DM-DOB-B	DM-DOB-B
DM-DCC-A		
DM-MTS-A	DM-MTS-A	
DM-OCD-A		
DM-RD-A		
DM-BR-A		
DM-DDA-A		

Audit Reporting

B-AWS	B-OWS	B-OD
AR-AVM-A		

L.1.1 BACnet Cross-Domain Advanced Workstation (B-XAWS)

The B-XAWS workstation is an advanced operator workstation for all building automation domains except life safety that includes the functionality of the following device profiles:

- B-AWS, see Clause L.1.2
- B-AACWS, see Clause L.3.1
- B-ALWS, see Clause L.9.1
- B-AEWS, see Clause L.12.1

L.1.2 BACnet Advanced Operator Workstation (B-AWS)

The B-AWS is the advanced operator's window into a BACnet system. It is primarily used to monitor the performance of a system and to modify parameters that affect the operation of a system. It may also be used for configuration activities that are beyond the scope of this standard.

The B-AWS profile is targeted at a building operator or technician with a higher level of technical ability. It provides support for limited configuration actions and ongoing commissioning activities.

The B-AWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e. reports and graphics)
- Ability to monitor the value of selected BACnet object types, including all required and optional properties
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits
- Adjustment of alarm routing
- Creation of new Event Enrollment and Notification Class objects
- Presentation of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars
- Creation of new calendars and schedules

Trending
- Modification of the parameters of a trend log
- Display of trend data
- Creation of new Trend Log objects

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices
- Ability to command half-routers to establish and terminate connections
- Ability to perform dynamic device assignment

Audit Reporting
- Query and display of logged audit records
- Ability to configure audit reporting in BACnet devices

L.1.3 BACnet Operator Workstation (B-OWS)

'The B-OWS is an operator interface with limited capabilities relative to a B-AWS. The B-OWS is used for monitoring and basic control of a system, but differs from a B-AWS in that it does not support configuration activities, nor does it provide advanced troubleshooting capabilities.

The B-OWS profile is targeted at the daily operator who needs the ability to monitor basic system status and to perform simple modifications to the operation of the system.

The B-OWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e. reports and graphics)
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of analog alarm limits

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars

Trending
- Display of trend data

Device and Network Management
- Ability to find other BACnet devices
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator

L.1.4 BACnet Operator Display (B-OD)

The B-OD is a basic operator interface with limited capabilities relative to a B-OWS. It is not intended to perform direct digital control. The B-OD profile could be used for wall-mounted LCD devices, displays affixed to BACnet devices; hand-held terminals or other very simple user interfaces.

The B-OD profile enables the specification of the following:

Data Sharing
- Presentation of basic data
- Ability to modify setpoints and parameters

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Alarm and Event Management
- Operator notification and presentation of event information

Scheduling
- No minimum requirements

Trending
- No minimum requirements

Device and Network Management
- Ability to find other BACnet devices

L.2 Life Safety Operator Interface Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-ALSWS	B-LSWS	B-LSAP
DS-RP-A,B	DS-RP-A,B	DS-RP-A,B
DS-RPM-A	DS-RPM-A	
DS-WP-A	DS-WP-A	DS-WP-A
DS-WPM-A	DS-WPM-A	
DS-LSAV-A	DS-LSV-A	DS-LSV-A
DS-LSAM-A	DS-LSM-A	

Alarm & Event Management

B-ALSWS	B-LSWS	B-LSAP
AE-N-A	AE-N-A	AE-N-A
AE-LS-A	AE-LS-A	AE-LS-A
AE-ACK-A	AE-ACK-A	AE-ACK-A
AE-AS-A	AE-AS-A	
AE-LSAVM-A	AE-LSVM-A	
AE-LSAVN-A	AE-LSAVN-A	AE-LSVN-A
AE-ELVM-A[1]	AE-ELV-A[1]	

[1] Not required for devices claiming conformance to a Protocol_Revision less than 7

Scheduling

B-ALSWS	B-LSWS	B-LSAP
SCHED-AVM-A	SCHED-VM-A	

Trending

B-ALSWS	B-LSWS	B-LSAP
T-AVM-A	T-V-A	

Device & Network Management

B-ALSWS	B-LSWS	B-LSAP
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-A,B
DM-ANM-A		
DM-ADM-A		
DM-DOB-B	DM-DOB-B	DM-DOB-B
DM-DCC-A	DM-DCC-A	
DM-MTS-A	DM-MTS-A	
DM-OCD-A		
DM-RD-A	DM-RD-A	
DM-BR-A	DM-BR-A	

Audit Reporting

B-ALSWS	B-LSWS	B-LSAP
AR-AVM-A		

L.2.1 BACnet Advanced Life Safety Workstation (B-ALSWS)

The B-ALSWS is an advanced life safety operator workstation that provides full support of the life-safety features of BACnet.

The B-ALSWS profile is targeted at a life safety operator or technician with a higher level of technical ability. It provides support for limited configuration actions and ongoing commissioning activities.

The B-ALSWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Presentation of life-safety data
- Ability to monitor the value of BACnet objects relevant for life safety, including all required and optional properties
- Ability to modify setpoints and parameters

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Alarm and Event Management
- Operator notification and presentation of event information, including life safety events
- Alarm acknowledgment by operators
- Life-safety silence and reset operations by operators
- Alarm summarization
- Adjustment of alarm limits and conditions, including life safety alarm parameters
- Adjustment of alarm routing
- Ability to create, delete and configure Event Enrollment, Notification Class and Notification Forwarder objects
- Presentation and modification of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars
- Creation and deletion of calendars and schedules

Trending
- Modification of the parameters of a trend log
- Display of trend data
- Creation of new Trend Log objects

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

Audit Reporting:
- Query and display of logged audit records
- Ability to configure audit reporting in BACnet devices

L.2.2 BACnet Life Safety Workstation (B-LSWS)

The B-LSWS is a life safety operator interface with limited capabilities relative to a B-ALSWS. The B- LSWS is used for monitoring and basic control of a BACnet life safety system, but differs from a B-ALSWS in that it does not support configuration activities, nor does it provide advanced troubleshooting capabilities.

The B-LSWS profile is targeted at the daily life safety operator who needs the ability to monitor basic system status and to perform simple modifications to the operation of the system.

The B-LSWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Presentation of life-safety data
- Ability to monitor the value of BACnet objects relevant for life safety, including all required and optional properties
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information, including life safety events
- Alarm acknowledgment by operators
- Life safety silence and reset operations by operators
- Alarm summarization
- Adjustment of alarm limits and conditions, including life safety alarm parameters
- Presentation of Event Logs

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars

Trending
- Display and archive of trend data

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

L.2.3 BACnet Life Safety Annunciator Panel (B-LSAP)

The B-LSAP is a life safety operator interface for the indication of life safety events and status. The B-LSAP is used for monitoring and event handling of a BACnet life safety system.

The B-LSAP profile is targeted at the daily life safety operator who needs the ability to monitor basic system status and to perform event handling of life safety events.

The B-LSAP profile enables the specification of the following:

Data Sharing
- Presentation of life safety object data

Alarm and Event Management
- Operator notification and presentation of event information, including life safety events
- Alarm acknowledgment by operators
- Life safety silence and reset operations by operators

Scheduling
- No minimum requirements

Trending
- No minimum requirements

Device and Network Management
- Ability to find other BACnet devices

L.3 Access Control Operator Interface Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-AACWS	B-ACWS	B-ACSD
DS-RP-A,B	DS-RP-A,B	DS-RP-A,B
DS-RPM-A	DS-RPM-A	DS-RPM-A
DS-WP-A	DS-WP-A	DS-WP-A
DS-WPM-A	DS-WPM-A	DS-WPM-A
DS-ACAV-A	DS-ACAV-A	DS-ACV-A
DS-ACAM-A	DS-ACM-A	DS-ACM-A
DS-ACUC-A	DS-ACUC-A	
DS-ACSC-A		

Alarm & Event Management

B-AACWS	B-ACWS	B-ACSD
AE-N-A	AE-N-A	AE-N-A
AE-AC-A	AE-AC-A	AE-AC-A
AE-ACK-A	AE-ACK-A	AE-ACK-A
AE-AS-A	AE-AS-A	AE-AS-A
AE-ACAVM-A	AE-ACVM-A	-
AE-ACAVN-A	AE-ACAVN-A	AE-ACAVN-A
AE-ELVM-A[1]	AE-ELV-A[1]	AE-ELV-A[1]

[1] Not required for devices claiming conformance to a Protocol_Revision less than 7

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Scheduling

B-AACWS	B-ACWS	B-ACSD
SCHED-AVM-A	SCHED-VM-A	SCHED-VM-A

Trending

B-AACWS	B-ACWS	B-ACSD

Device & Network Management

B-AACWS	B-ACWS	B-ACSD
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-A,B
DM-ANM-A		
DM-ADM-A		
DM-DOB-B	DM-DOB-B	DM-DOB-B
DM-DCC-A	DM-DCC-A	
DM-MTS-A	DM-MTS-A	
DM-OCD-A	DM-OCD-A	
DM-RD-A		
DM-BR-A		

Audit Reporting

B-AACWS	B-ACWS	B-ACSD
AR-AVM-A		

L.3.1 BACnet Advanced Access Control Workstation (B-AACWS)

The B-AACWS workstation is an advanced operator workstation for physical access control systems that is able to enroll and manage access control users and their credentials and rights. It is able to create, delete, and configure access points, access zones, access doors, and credential data inputs. In addition, it provides all functions required for a security operator who operates the physical access control system.

The B-AACWS profile is targeted at an access control administrator who has the maximum privileges on the physical access control system, including operation of the access control system.

The B-AACWS profile enables the specification of the following:

Data Sharing

- Presentation of data (i.e., reports and graphics)
- Ability to monitor the value of BACnet objects relevant for access control, including all required and optional properties
- Ability to modify setpoints and parameters
- Ability to create, delete, and configure Access User, Access Credential, and Access Rights objects
- Ability to create, delete, and configure Access Zone and Access Point objects
- Ability to create, delete, and configure Access Door and Credential Data Input objects

Alarm and Event Management

- Operator notification and presentation of event information, including access events
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits and conditions
- Adjustment of alarm routing
- Ability to create, delete, and configure Event Enrollment, Notification Class, and Notification Forwarder objects
- Presentation and modification of Event Logs

Scheduling

- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars
- Creation and deletion of calendars and schedules

Trending

- No requirements

Device and Network Management

- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

Audit Reporting:
- Query and display of logged audit records
- Ability to configure audit reporting in BACnet devices

L.3.2 BACnet Access Control Workstation (B-ACWS)

The B-ACWS workstation is an operator workstation for physical access control systems that is able to enroll and manage access control users and their credentials. It is able to create, delete, and configure access users, access credentials, and access rights. In addition, it provides all functions required for a security operator who operates the physical access control system.

The B-ACWS profile is targeted at an access control user administrator who has the privileges to enroll and manage users in the physical access control system and to operate the access control system.

The B-ACWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Ability to monitor the value of BACnet objects relevant for access control, including all required and optional properties
- Ability to modify setpoints and parameters
- Ability to create, delete, and configure Access User, Access Credential, and Access Rights objects

Alarm and Event Management
- Operator notification and presentation of event information, including access events
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits and conditions
- Presentation of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars

Trending
- No requirements

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

L.3.3 BACnet Access Control Security Display (B-ACSD)

The B-ACSD device is a security operator display device. It is able to monitor and control the access control devices. Some minimal level of configuration support is considered a capability of such devices, e.g., setting threat levels, etc. It is also able to receive and handle access events and retrieve and present event logs from access control controllers.

The B-ACSD profile is targeted at security personnel to monitor and control the devices of a physical access control system.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The B-ACSD profile enables the specification of the following:

Data Sharing
- Presentation of basic data of BACnet object types relevant for access control
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information, including access events
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits and conditions
- Presentation of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars

Trending
- No requirements

Device and Network Management
- Ability to find other BACnet devices
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator

L.4 Controller Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-BC	B-AAC	B-ASC	B-SA	B-SS
DS-RP-A,B	DS-RP-B	DS-RP-B	DS-RP-B	DS-RP-B
DS-RPM-A,B	DS-RPM-B			
DS-WP-A,B	DS-WP-B	DS-WP-B	DS-WP-B	
DS-WPM-B	DS-WPM-B			

Alarm & Event Management

B-BC	B-AAC	B-ASC	B-SA	B-SS
AE-N-I-B	AE-N-I-B			
AE-ACK-B	AE-ACK-B			
AE-INFO-B	AE-INFO-B			
AE-ESUM-B[1]				
AE-CRL-B[3]	AE-CRL-B[3]			

[1] Not required for devices claiming conformance to Protocol_Revision 13 or greater.
[3] Not required for devices claiming conformance to a Protocol_Revision less than 19.

Scheduling

B-BC	B-AAC	B-ASC	B-SA	B-SS
SCHED-E-B	SCHED-I-B			

Trending

B-BC	B-AAC	B-ASC	B-SA	B-SS
T-VMT-I-B				
T-ATR-B				

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device & Network Management

B-BC	B-AAC	B-ASC	B-SA	B-SS
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-B	DM-DDB-B[2]	DM-DDB-B[2]
DM-DOB-B	DM-DOB-B	DM-DOB-B	DM-DOB-B[2]	DM-DOB-B[2]
DM-DCC-B	DM-DCC-B	DM-DCC-B		
DM-TS-B or DM-UTC-B	DM-TS-B or DM-UTC-B			
DM-RD-B	DM-RD-B			
DM-BR-B				

[2] Not required if the device is a BACnet MS/TP Slave.

L.4.1 BACnet Building Controller (B-BC)

A B-BC is a general-purpose, field-programmable device capable of carrying out a variety of building automation and control tasks. It enables the specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to retrieve the values of BACnet objects from other devices
- Ability to allow modification of some or all of its BACnet objects by another device
- Ability to modify some BACnet objects in other devices

Alarm and Event Management
- Supports configuration of event recipient lists
- Generation of alarm / event notifications and the ability to direct them to recipients
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters

Scheduling
- Ability to schedule output actions, both in the local device and in other devices, both binary and analog, based on date and time

Trending
- Collection and delivery of (time, value) pairs

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request
- Ability to upload its configuration and allow it to be subsequently restored

L.4.2 BACnet Advanced Application Controller (B-AAC)

A B-AAC is a control device with limited resources relative to a B-BC. It may be intended for specific applications and supports some degree of programmability.

Data Sharing
- Ability to provide values for any of its BACnet objects upon request
- Ability to allow modification of some or all of its BACnet objects by another BACnet device

Alarm and Event Management
- Supports configuration of event recipient lists
- Generation of limited alarm and event notifications and the ability to direct them to recipients
- Tracking acknowledgments of alarms from human operators
- Adjustment of alarm parameters

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Scheduling
- Ability to schedule actions in the local device based on date and time

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request

L.4.3 BACnet Application Specific Controller (B-ASC)

A B-ASC is a controller with limited resources relative to a B-AAC. It is intended for use in a specific application and supports limited programmability. It enables specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.4.4 BACnet Smart Actuator (B-SA)

A B-SA is a simple control device with limited resources; it is intended for specific applications.

Data Sharing
- Ability to provide values for any of its BACnet objects upon request
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

L.4.5 BACnet Smart Sensor (B-SS)

A B-SS is a simple sensing device with very limited resources.

Data Sharing
- Ability to provide values for any of its BACnet objects upon request

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects

L.5 Life Safety Controller Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-ALSC	B-LSC
DS-RP-B	DS-RP-B
DS-RPM-B	
DS-WP-B	DS-WP-B
DS-WPM-B	
DS-COV-B	DS-COV-B

Alarm & Event Management

B-ALSC	B-LSC
AE-LS-B	AE-LS-B
AE-ACK-B	AE-ACK-B
AE-INFO-B	AE-INFO-B
AE-EL-I-B[1]	

[1] Not required for devices claiming conformance to a Protocol_Revision less than 7

Scheduling

B-ALSC	B-LSC
SCHED-I-B	

Trending

B-ALSC	B-LSC

Device & Network Management

B-ALSC	B-LSC
DM-DDB-A,B	DM-DDB-A,B
DM-DOB-B	DM-DOB-B
DM-DCC-B	DM-DCC-B
DM-TS-B or DM-UTC-B	
DM-RD-B	

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

L.5.1 BACnet Advanced Life Safety Controller (B-ALSC)

A B-ALSC device performs life safety alarm detection and control. It supports life safety objects and limited modification of its alarm and event reporting, including event distribution by another device. Also, the device supports scheduling of internal values and event logging.

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device for control and configuration purposes

© ISO 2022 – All rights reserved

Alarm and Event Management
- Generation of alarm / event notifications including life safety alarms and the ability to direct them to recipients
- Execution of all forms of life safety operations
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters including notification distribution
- Logging of event notifications of the local device in an Event Log object

Scheduling
- Ability to schedule actions in the local device for binary, analog, and enumerated values, based on date and time

Trending
- No requirements

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request

L.5.2 BACnet Life Safety Controller (B-LSC)

A B-LSC device performs life safety alarm detection and control. It supports life safety objects. The B-LSC device is not required to support modification of its alarm and event reporting by another device.

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of a limited set of its properties by another device for control purposes

Alarm and Event Management
- Generation of alarm / event notifications including life safety alarms and the ability to direct them to recipients
- Execution of all forms of life safety operations
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.6 Access Control Controller Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

Data Sharing

B-AACC	B-ACC
DS-RP-A,B	DS-RP-B
DS-RPM-A,B	DS-RPM-B
DS-WP-A,B	DS-WP-B
DS-WPM-B	DS-WPM-B
DS-COV-A,B	DS-COV-B
DS-ACAD-A	
DS-ACCDI-A	
DS-ACUC-B	DS-ACUC-B
DS-ACSC-B	DS-ACSC-B

Alarm & Event Management

B-AACC	B-ACC
AE-AC-B	AE-AC-B
AE-ACK-B	AE-ACK-B
AE-INFO-B	AE-INFO-B
AE-EL-I-B[1]	

[1] Not required for devices claiming conformance to a Protocol_Revision less than 7

Scheduling

B-AACC	B-ACC
SCHED-I-B	SCHED-I-B

Trending

B-AACC	B-ACC

Device & Network Management

B-AACC	B-ACC
DM-DDB-A,B	DM-DDB-A,B
DM-DOB-B	DM-DOB-B
DM-DCC-B	DM-DCC-B
DM-TS-B or DM-UTC-B	DM-TS-B or DM-UTC-B
DM-RD-B	DM-RD-B
DM-BR-B	

L.6.1 BACnet Advanced Access Control Controller (B-AACC)

A B-AACC device performs physical access control authentication and authorization for physical access. It supports the modification of its Access User, Access Credential, and Access Rights objects by another device. It also supports the modification by another device of its Access Point and Access Zone objects. It is capable to use remote Access Control Door Controller devices and remote Access Control Credential Reader devices.

Data Sharing

- Ability to provide the values of any of its BACnet objects
- Ability to retrieve the values of BACnet objects from other devices
- Ability to allow creation, deletion, and modification of some or all of its BACnet objects by another device
- Ability to modify some BACnet objects in other devices

Alarm and Event Management

- Generation of alarm / event notifications, including access events and the ability to direct them to recipients
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters
- Logging of event notifications of the local device in an Event Log object

Scheduling

- Ability to schedule values, based on date and time

Trending

- No requirements

Device and Network Management

- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request
- Ability to upload its configuration and allow it to be subsequently restored

L.6.2 BACnet Access Control Controller (B-ACC)

A B-ACC device performs authentication and authorization for physical access. It supports the modification of its Access Credential and Access Rights objects by another device. The support of Access User objects is optional. It also supports the modification by another device of its Access Point and Access Zone objects for and at which the authentication and authorization is performed.

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- Generation of alarm / event notifications, including access events and the ability to direct them to recipients
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters

Scheduling
- Ability to schedule values, based on date and time

Trending
- No requirements

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request

L.7 Miscellaneous Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-RTR	B-GW	B-BBMD	B-ACDC	B-ACCR	B-SCHUB
DS-RP-B	DS-RP-B	DS-RP-B	DS-RP-B	DS-RP-B	DS-RP-B
DS-WP-B	DS-WP-B	DS-WP-B	DS-WP-B	DS-WP-B	
				DS-COV-B	
			DS-ACAD-B		
				DS-ACCDI-B	

Alarm & Event Management

B-RTR	B-GW	B-BBMD	B-ACDC	B-ACCR	B-SCHUB

Scheduling

B-RTR	B-GW	B-BBMD	B-ACDC	B-ACCR	B-SCHUB

Trending

B-RTR	B-GW	B-BBMD	B-ACDC	B-ACCR	B-SCHUB

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device & Network Management

B-RTR	B-GW	B-BBMD	B-ACDC	B-ACCR	B-SCHUB
DM-DDB-B	DM-DDB-B	DM-DDB-B	DM-DDB-B	DM-DDB-B	DM-DDB-B
DM-DOB-B	DM-DOB-B	DM-DOB-B	DM-DOB-B	DM-DOB-B	DM-DOB-B
			DM-DCC-B	DM-DCC-B	
NM-RC-B					
		NM-BBMDC-B			
	GW-EO-B[1]				
	GW-VN-B[1]				
					NM-SCH-B

[1] One of these BIBBs shall be supported.

L.7.1 BACnet Router (B-RTR)

A B-RTR is a BACnet network router. It connects two or more BACnet networks at the network layer. It enables specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to respond to network layer messages
- Ability to connect two or more data links

L.7.2 BACnet Gateway (B-GW)

A B-GW is a BACnet gateway. It is intended for connecting one or more non-BACnet networks into a BACnet internetwork. There are two types of B-GW devices, those that represent non-BACnet devices as a network of virtual BACnet devices, and those that represent non-BACnet devices as BACnet objects. It enables specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to respond to network layer messages
- Ability to respond to queries about the status of a virtual BACnet device
- Ability to respond to requests for information about any of the objects of a virtual BACnet device

L.7.3 BACnet Broadcast Management Device (B-BBMD)

A B-BBMD is a BACnet broadcast management device as defined in Annex J. It is intended for transporting broadcasts between different IP subnets. Devices which provide BBMD functionality along with some other BACnet functionality shall be capable of being reconfigured as a non-BBMD. It enables specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to respond to network layer messages
- Ability to manage BVLC functions

L.7.4 BACnet Access Control Door Controller (B-ACDC)

A B-ACDC is a device that controls doors. It represents the doors that it controls through the respective Access Door objects. It does not itself perform authentication and authorization for physical access.

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

L.7.5 BACnet Access Control Credential Reader (B-ACCR)

This is a device that performs credential data reading and provides access to credential reader elements and functions. It represents its credential data reading functionality through Credential Data Input objects, and may use any other objects to represent additional credential reader device functions. Support of COV reporting of its Credential Data Input objects is required.

Data Sharing
- Ability to provide the values of any of its BACnet objects
- Ability to provide authentication factors through COV reporting
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.7.6 BACnet Secure Connect Hub (B-SCHUB)

A B-SCHUB is a device that is able to perform the BACnet Secure Connect hub function as defined in Annex AB. The BACnet/SC hub function accepts secured Web Socket connections as defined for BACnet/SC hub connections.

Data Sharing
- Ability to provide the values of any of its BACnet objects.

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to perform the BACnet/SC hub function.

L.8 BACnet General (B-GENERAL) Profile

The following table indicates which BIBBs shall be supported by this device type for each interoperability area.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Data Sharing

B-GENERAL
DS-RP-B

Alarm & Event Management

B-GENERAL

Scheduling

B-GENERAL

Trending

B-GENERAL

Device & Network Management

B-GENERAL
DM-DDB-B[1]
DM-DOB-B[1]

[1] Not required if the device is a BACnet MS/TP Slave.

A B-GENERAL device is a BACnet device whose main function does not fall under any of the other device profiles. While the device may meet the functional requirements of one or more device profiles, the general description of the other profiles do not match the intended application of the device. It enables specification of the following:

Data Sharing
- Ability to provide the values of any of its BACnet objects

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects

L.9 Lighting Operator Interface Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area. The B-XAWS is not shown in this table. See Clause L.1.1.

Data Sharing

B-ALWS	B-LOD
DS-RP-A,B	DS-RP-A,B
DS-RPM-A	
DS-WP-A	DS-WP-A
DS-WPM-A	
DS-LAV-A	DS-LV-A
DS-LAM-A	DS-LM-A
DS-WG-A	DS-WG-A
DS-ALO-A	DS-ALO-A

Alarm & Event Management

B-ALWS	B-LOD
AE-N-A	
AE-ACK-A	
AE-AS-A	
AE-AVM-A	
AE-AVN-A	
AE-ELVM-A	

Scheduling

B-ALWS	B-LOD
SCHED-AVM-A	

Trending

B-ALWS	B-LOD
T-AVM-A	

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device & Network Management

B-ALWS	B-LOD
DM-DDB-A,B	DM-DDB-A,B
DM-ANM-A	
DM-ADM-A	
DM-DOB-B	DM-DOB-B
DM-DCC-A	
DM-MTSA	
DM-OCD-A	
DM-RD-A	
DN-BR-A	

L.9.1 BACnet Advanced Lighting Workstation (B-ALWS)

The B-ALWS workstation is an advanced operator workstation with abilities to monitor, control, and configure lighting devices.

The B-ALWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Ability to monitor the value of BACnet objects relevant for lighting, including all required and optional properties
- Ability to modify lighting parameters
- Ability to create, delete, and configure Lighting Output, Binary Lighting Output, and Channel objects
- Ability to command lighting devices using WriteGroup
- Ability to configure advanced lighting parameters in lighting devices

Alarm and Event Management
- Operator notification and presentation of event information
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits and conditions
- Adjustment of alarm routing
- Ability to create, delete, and configure Event Enrollment, Notification Class, and Notification Forwarder objects
- Presentation and modification of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars
- Creation and deletion of calendars and schedules

Trending
- Modification of the parameters of a Trend Log object
- Display of trend data
- Creation and deletion of new Trend Log objects

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

L.9.2 BACnet Lighting Operator Display (B-LOD)

The B-LOD is a basic operator interface with limited capabilities relative to a B-ALWS. The B-LOD profile could be used for wall-mounted display devices, simple web server gateways to BACnet lighting devices, displays affixed to BACnet devices; handheld terminals or other very simple control stations.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

The B-LOD profile enables the specification of the following:

Data Sharing
- Presentation of basic data
- Ability to monitor the value of BACnet objects relevant for lighting
- Ability to modify lighting parameters
- Ability to command lighting devices using WriteGroup
- Ability to configure advanced lighting parameters in lighting devices

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to find other BACnet devices
- Ability to choose among discovered BACnet devices
- Ability to find objects in other BACnet devices
- Ability to choose among discovered BACnet device's objects

L.10 Lighting Control Station Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-ALCS	B-LCS
DS-RP-A,B	DS-RP-A,B
DS-RPM-A	
DS-WP-A	DS-WP-A
DS-WPM-A	
DS-WG-A	
DS-ALO-A	DS-LO-A

Alarm & Event Management

B-ALCS	B-LCS

Scheduling

B-ALCS	B-LCS
SCHED-E-B	

Trending

B-ALCS	B-LCS

Device & Network Management

B-ALCS	B-LCS
DM-DDB-A,B	DM-DDB-A,B
DM-DOB-B	DM-DOB-B
DM-DCC-B	DM-DCC-B
DM-TS-B or DM-UTC-B	

L.10.1 BACnet Advanced Lighting Control Station (B-ALCS)

A B-ALCS is any device that needs to interact with Lighting Output, Binary Lighting Output, and Channel objects in other devices for the purpose of monitoring and/or controlling them. B-ALCS devices are not Workstations and Operator Displays. Instead, they are more limited devices nonetheless used for configuration. Examples might include smart wall switches/dimmers, scene preset consoles, etc. B-ALCS devices differ from B-LCS devices in B-ALCS's ability to configure and command advanced lighting parameters in lighting devices.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Data Sharing
- Ability to monitor and command BACnet lighting devices
- Ability to provide the values of any of its BACnet objects
- Ability to command lighting devices using WriteGroup

Alarm and Event Management
- No requirement

Scheduling
- Ability to schedule output actions, both in the local device and in other devices, both binary and analog, based on date and time

Trending
- No requirement

Device and Network Management
- Ability to find other BACnet devices
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.10.2 BACnet Lighting Control Station (B-LCS)

A B-LCS is any device that needs to interact with Lighting Output or Binary Lighting Output objects in other devices for the purpose of monitoring and/or controlling them. B-LCS devices are different from Workstations and Operator Displays in that their interactions with humans are more limited. Examples might include smart wall switches/dimmers, scene preset consoles, etc.

Data Sharing
- Ability to command BACnet lighting objects' Present_Value properties
- Ability to provide the values of any of its BACnet objects

Alarm and Event Management
- No requirement

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to find other BACnet devices
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.11 Lighting Controller Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Data Sharing

B-LS	B-LD
DS-RP-B	DS-RP-B
DS-WP-A,B	DS-WP-B
DS-WG-E-B	
DS-ALO-A	DS-BLO-B or DS-LO-B

Alarm & Event Management

B-LS	B-LD

Scheduling

B-LS	B-LD
SCHED-E-B	

Trending

B-LS	B-LD

Device & Network Management

B-LS	B-LD
DM-DDB-A,B	DM-DDB-B
DM-DOB-B	DM-DOB-B
DM-DCC-B	DM-DCC-B
DM-TS-B or DM-UTC-B	

L.11.1 BACnet Lighting Supervisor (B-LS)

A B-LS is any device that implements Channel objects and optionally Lighting Output and/or Binary Lighting Output objects with the ability to forward channel writes to other BACnet devices.

Data Sharing
- Ability to provide values for any of its BACnet objects upon request
- Ability to allow modification of some or all of its BACnet objects by another device
- Ability to execute WriteGroup commands
- Ability to propagate Channel values to objects external to the device

Alarm and Event Management
- No requirement

Scheduling
- Ability to schedule output actions, both in the local device and in other devices, both binary and analog, based on date and time

Trending
- No requirement

Device and Network Management
- Ability to find other BACnet devices
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request

L.11.2 BACnet Lighting Device (B-LD)

A B-LD is any device that implements Binary Lighting Output and/or Lighting Output objects

Data Sharing
- Ability to provide values for any of its BACnet objects upon request
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- No requirement

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Scheduling
- No requirement

Trending
- No requirement

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

L.12 Elevator Operator Interface Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area. The B-XAWS is not shown in this table. See Clause L.1.1

Data Sharing

B-AEWS	B-EWS	B-ED
DS-RP-A,B	DS-RP-A,B	DS-RP-A,B
DS-RPM-A	DS-RPM-A	
DS-WP-A	DS-WP-A	DS-WP-A
DS-WPM-A	DS-WPM-A	
DS-COVM-A	DS-COVM-A	
DS-EAV-A	DS-EV-A	DS-EV-A
DS-EAM-A	DS-EM-A	

Alarm & Event Management

B-AEWS	B-EWS	B-ED
AE-N-A	AE-N-A	AE-N-A
AE-ACK-A	AE-ACK-A	AE-ACK-A
AE-AS-A	AE-AS-A	
AE-EAVM-A	AE-EVM-A	
AE-EAVN-A	AE-EAVN-A	AE-EVN-A
AE-ELVM-A		

Scheduling

B-AEWS	B-EWS	B-ED
SCHED-AVM-A	SCHED-VM-A	

Trending

B-AEWS	B-EWS	B-ED
T-AVM-A	T-V-A	

Device & Network Management

B-AEWS	B-EWS	B-ED
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-A,B
DM-ANM-A		
DM-ADM-A		
DM-DOB-B	DM-DOB-B	DM-DOB-B
DM-DCC-A		
DM-MTS-A	DM-MTS-A	
DM-OCD-A		
DM-RD-A		
DM-BR-A		

L.12.1 BACnet Advanced Elevator Workstation (B-AEWS)

The B-AEWS is an advanced elevator operator workstation that provides full support of the elevator features of BACnet.

The B-AEWS profile is targeted at an elevator operator or technician with a higher level of technical ability. It provides support for limited configuration actions and ongoing commissioning activities.

The B-AEWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Presentation of elevator data
- Ability to monitor the value of BACnet objects relevant for elevator, including all required and optional properties
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information, including events from elevator objects.
- Alarm acknowledgment by operators
- Alarm summarization

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

- Adjustment of alarm and fault conditions, including elevator alarm and fault parameters
- Adjustment of alarm routing
- Ability to create, delete and configure Event Enrollment, Notification Class and Notification Forwarder objects
- Presentation and modification of Event Logs

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars
- Creation and deletion of calendars and schedules

Trending
- Modification of the parameters of a trend log
- Display of trend data
- Creation of new Trend Log objects

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to silence a device on the network that is transmitting erroneous data
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator
- Ability to cause a remote device to reinitialize itself
- Ability to backup and restore the configuration of other devices

L.12.2 BACnet Elevator Workstation (B-EWS)

The B-EWS is an elevator operator interface with limited capabilities relative to a B-AEWS. The B-EWS is used for monitoring and basic control of a BACnet elevator system, but differs from a B-AEWS in that it does not support configuration activities, nor does it provide advanced troubleshooting capabilities.

The B-EWS profile is targeted at the daily elevator operator who needs the ability to monitor basic system status and to perform simple commands and modifications to the operation of the system.

The B-EWS profile enables the specification of the following:

Data Sharing
- Presentation of data (i.e., reports and graphics)
- Presentation of elevator data
- Ability to monitor the value of BACnet objects relevant for elevators, including all required and optional properties
- Ability to modify setpoints and parameters

Alarm and Event Management
- Operator notification and presentation of event information, including elevator events
- Alarm acknowledgment by operators
- Alarm summarization
- Adjustment of alarm limits and conditions, including elevator alarm and fault parameters

Scheduling
- Modification of calendars and schedules
- Display of the start and stop times (schedule) of scheduled devices
- Display of calendars

Trending
- Display and archive of trend data

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

Device and Network Management
- Ability to find other BACnet devices
- Ability to find all objects in BACnet devices
- Ability to synchronize the time in devices across the BACnet internetwork at the request of the operator

L.12.3 BACnet Elevator Display (B-ED)

The B-ED is an elevator interface for the indication of elevator events and status. The B-ED is used for displaying the status of an elevator system.

The B-ED profile is targeted at the elevator operator and user who needs the ability to view the basic elevator system status.

The B-ED profile enables the specification of the following:

Data Sharing
- Presentation of elevator object data
- Ability to modify setpoints and parameters

Alarm and Event Management
- Presentation of event information, including elevator events and faults.

Scheduling
- No minimum requirements

Trending
- No minimum requirements

Device and Network Management
- Ability to find other BACnet devices

L.13 Elevator Controller Profiles

The following table indicates which BIBBs shall be supported by the device types of this family, for each interoperability area.

Data Sharing

B-AEC	B-EC	B-EM
DS-RP-B	DS-RP-B	DS-RP-B
DS-RPM-B	DS-RPM-B	DS-RPM-B
DS-WP-B	DS-WP-B	
DS-WPM-B	DS-WPM-B	
DS-COV-B	DS-COV-B	DS-COV-B
DS-COVM-B	DS-COVM-B	DS-COVM-B

Alarm & Event Management

B-AEC	B-EC	B-EM
AE-N-I-B	AE-N-I-B	AE-N-I-B
AE-ACK-B	AE-ACK-B	AE-ACK-B
AE-INFO-B	AE-INFO-B	AE-INFO-B
AE-EL-I-B		

Scheduling

B-AEC	B-EC	B-EM
SCHED-I-B		

Trending

B-AEC	B-EC	B-EM

Device & Network Management

B-AEC	B-EC	B-EM
DM-DDB-A,B	DM-DDB-A,B	DM-DDB-B
DM-DOB-B	DM-DOB-B	DM-DOB-B
DM-DCC-B	DM-DCC-B	DM-DCC-B
DM-TS-B or DM-UTC-B	DM-TS-B or DM-UTC-B	
DM-OCD-B		
DM-RD-B	DM-RD-B	
DM-BR-B		

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

L.13.1 BACnet Advanced Elevator Controller (B-AEC)

A B-AEC device performs control of elevators. It supports the modification and commanding of its elevator objects by another device.

Data Sharing
- Ability to contain elevator objects.
- Ability to provide the values of any of its BACnet objects
- Ability to allow creation, deletion, and modification of some or all of its BACnet objects by another device

Alarm and Event Management
- Generation of alarm / event notifications of internal objects and the ability to direct notifications to recipients.
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters
- Logging of event notifications of the local device in an Event Log object

Scheduling
- Ability to schedule internal values, based on date and time

Trending
- No requirements

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request
- Ability to upload its configuration and allow it to be subsequently restored

L.13.2 BACnet Elevator Controller (B-EC)

A B-EC device performs elevator control. It supports limited modification of its elevator objects by another device.

Data Sharing
- Ability to contain elevator objects.
- Ability to provide the values of any of its BACnet objects
- Ability to allow modification of some or all of its BACnet objects by another device

Alarm and Event Management
- Generation of alarm / event notifications of internal objects and the ability to direct notifications to recipients.
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received
- Adjustment of alarm / event parameters

Scheduling
- No requirements

Trending
- No requirements

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages
- Ability to synchronize its internal clock upon request
- Ability to perform re-initialization upon request

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved

L.13.3 BACnet Elevator Monitor (B-EM)

A B-EM device performs monitoring of an elevator control system. It supports presentation of its elevator objects by another device, but is not required to support modifications of any of its elevator objects.

Data Sharing
- Ability to contain elevator objects.
- Ability to provide the values of any of its BACnet objects

Alarm and Event Management
- Generation of alarm / event notifications of internal objects and the ability to direct notifications to recipients.
- Maintain a list of unacknowledged alarms / events
- Notifying other recipients that the acknowledgment has been received

Scheduling
- No requirements

Trending
- No requirements

Device and Network Management
- Ability to respond to queries about its status
- Ability to respond to requests for information about any of its objects
- Ability to respond to communication control messages

STANDARDSISO.COM : Click to view the full PDF of ISO 16484-5:2022

© ISO 2022 – All rights reserved