IEC 62948

Edition 1.0 2017-07

INTERNATIONAL STANDARD

colour inside

Industrial networks – Wireless communication network and communication profiles – WIA-FA

IEC 62948:2017-07(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing 20 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
65 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

IEC 62948

Edition 1.0 2017-07

INTERNATIONAL STANDARD

colour
inside

Industrial networks – Wireless communication network and communication profiles – WIA-FA

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 25.040.40; 35.110

ISBN 978-2-8322-4607-8

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

INDUSTRIAL NETWORKS – WIRELESS COMMUNICATION NETWORK AND COMMUNICATION PROFILES – WIA-FA

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.

3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this publication.

7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.

8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 62948 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation. The text of this standard is based on the following documents:

This first edition cancels and replaces the IEC PAS 62948 published in 2015. This edition constitutes a technical revision.

The text of this standard is based on the following documents:

FDIS	Report on voting
65C/877/FDIS	65C/885/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2.

The committee has decided that the contents of this publication will remain unchanged until the stability dateindicated on the IEC website under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

The contents of the corrigendum of March 2021 have been included in this copy.

INDUSTRIAL NETWORKS – WIRELESS COMMUNICATION NETWORK AND COMMUNICATION PROFILES – WIA-FA

1 Scope

This International Standard specifies the system architecture and communication protocol of WIA-FA (Wireless Networks for Industrial Automation – Factory Automation) based on IEEE STD 802.11-2012 physical layer (PHY).

This document applies to wireless network systems for factory automation measuring, monitoring and control.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61588, *Precision clock synchronization protocol for networked measurement and control systems*

IEEE STD 802.11-2012, *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*

3 Terms, definitions, abbreviated terms, and conventions

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1.1
absolute timeslot number
number of timeslots counted from the start of the network

Note 1 to entry: The value is the sequence number of the current timeslot, which is incremented by one each timeslot. Once the maximum value ($2^{48}-1$) is reached, the value is reset to 0.

3.1.2
access device
device installed in the industrial field, which forwards the sensor data, alarms and network management related information of the field devices to the gateway device, or forwards control signals, management information and configuration information of the gateway device to field devices

3.1.3
aggregation
process of aggregating data of multiple user application objects into one packet, or aggregating several frames into one frame

3.1.4
application configuration
configuration procedure for user application processes in field devices to finish a factory automation task

3.1.5
application sub-layer
protocol sub-layer that provides data and management services for the application layer

3.1.6
backoff
process of retrying a frame in pre-determined retransmission timeslots if the previous transmission fails

3.1.7
beacon
special frame broadcasted by the access devices in the WIA-FA network

Note 1 to entry: To join the WIA-FA network, a new field device has to first listen to beacons.

3.1.8
channel
RF medium used to convey a frame from a sender to a receiver

3.1.9
coexistence
wireless communication coexistence
state in which all wireless communication solutions of a plant using shared medium fulfill all their application communication requirements

Note 1 to entry: This is consistent with the definition of coexistence in IEEE 802.15.2-2003.

[SOURCE: IEC 62657-2:2013, 3.1.12]

3.1.10
communication resource
channels and timeslots used to transport a frame

3.1.11
disaggregation
process of dividing the aggregated packet into data of multiple user application objects, or dividing the aggregated frame into multiple frames

3.1.12
field device
device installed in the industrial field and connected to sensors or actuators, which is used for transmitting field data and receiving control commands

3.1.13
gateway device
device that connects the WIA-FA network to other plant networks

3.1.14
handheld device
portable device used for network provisioning and firmware updating

3.1.15
heartbeat signal
signal sent by the primary gateway device to the redundant ones, which indicates the gateway device is operating properly

3.1.16
host computer
computer that is used for configuration, monitoring, and controlling

3.1.17
joining
process by which a WIA-FA device attempts and is allowed to participate in the WIA-FA network

3.1.18
link
interconnecting path between two neighbouring devices in the WIA-FA network, which consists of a set of parameters for one hop communication

Note 1 to entry: Parameters of link include link identifier, link type, active timeslot, peer address, relative timeslot number, current channel index, and superframe identifier.

3.1.19
long address
EUI-64 bits uniquely identifying the device in the WIA-FA network

Note 1 to entry: A long address is assigned by a manufacturer.

3.1.20
network address
8-bit or 16-bit unsigned integer uniquely identifying the device in the WIA-FA network, which is also called short address

3.1.21
network configuration
process of configuring parameters for WIA-FA devices to maintain network operation and communication

3.1.22
network manager
logical role configuring the network, allocating communication resources, monitoring and reporting network status

Note 1 to entry: There is only one network manager in a WIA-FA network.

3.1.23
passive leaving
process by which an online field device is instructed by the gateway device to leave the WIA-FA network

3.1.24
primary gateway device
gateway device currently operating in a WIA-FA network

Note 1 to entry: A WIA-FA network has only one primary gateway device.

3.1.25
provisioning
process of pre-configuring some static information that includes network identifier, security level, join key, and shared key for the WIA-FA devices

3.1.26
redundant gateway device
hot backup of the primary gateway device

3.1.27
relative timeslot number
number of timeslots counted from the start of a superframe

3.1.28
security manager
logical role configuring the security policies of the whole network, managing keys, and authenticating devices

3.1.29
superframe
collection of timeslots cyclically repeating at a constant rate

3.1.30
timeslot
basic time unit used for data exchange in WIA-FA network

Note 1 to entry: Its duration is configurable in the WIA-FA network.

3.2 Abbreviated terms

For the purposes of this document, the following abbreviated terms apply.

ACK	Acknowledgement
AI	Analog Input
AD	Access Device
AL	Application Layer
ASLDE	Application Sub-layer Digital Entity
ASLM	ASL State Machine
AMCL	ASL State Machine of Client
AMSV	ASL State Machine of Server
AMPB	ASL State Machine of Publisher
AMSB	ASL State Machine of Subscriber
AMRS	ASL State Machine of Report Source
AMRK	ASL State Machine of Report Sink
AO	Analog Output
ASL	Application Sub-layer
ASN	Absolute timeSlot Number
APDU	Application Protocol Data Unit
ASDU	Application Service Data Unit
C/S	Client/Server
CCM*	Extension of counter with cipher block chaining message authentication code

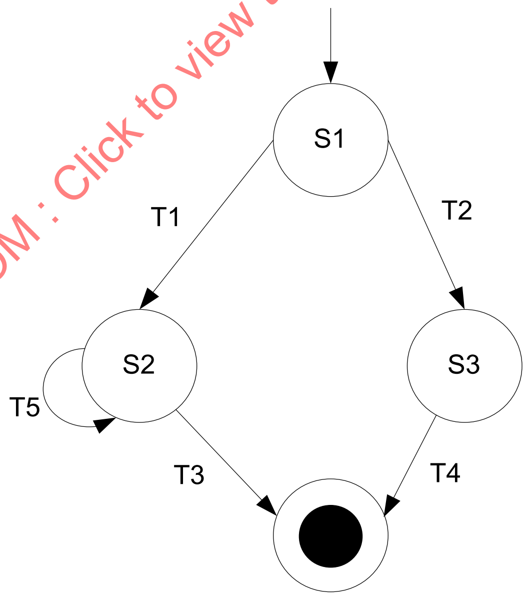
DGO	DisaGgregation Object
DI	Digital Input
DLDE	Data Link layer Data Entity
DLL	Data Link Layer
DLME	Data Link layer Management Entity
DLPDU	Data link Layer Protocol Data Unit
DMAP	Device Management Application Process
DO	Digital Output
DoS	Deny of Service
DSSS	Direct Sequence Spread Spectrum
EIRP	Equivalent Isotropic Radiated Power
ENC	ENCryption
EUI-64	Extended Unique Identifier-64 bits
FCS	Frame Check Sequence
FD	Field Device
FDMA	Frequency Division Multiple Access
FHSS	Frequency-Hopping Spread Spectrum
GACK	Group ACK
GW	Gateway Device
HC	Host Computer
HD	Handheld Device
HMAC	keyed-Hash Message Authentication Code
ID	Identifier
KED	Data Encryption Key
KEDB	Broadcast Data Encryption Key
KEDU	Unicast Data Encryption Key
KEK	Key Encryption Key
KJ	Join Key
KS	Shared Key
LSB	Least Significant Bit
LQI	Link Quality Indication
MAC	Medium Access Control
MIB	Management Information Base
MIC	Message Integrity Code
MSB	Most Significant Bit
NACK	Negative Acknowledgement
NM	Network Manager
NONCE	Number used once, a value that has (at most) a negligible chance of repeating
NRT	Non-Real-Time
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open System Interconnection
PAGO	Packet Aggregation Object

PDU	Protocol Data Unit
PHY	PHYsical layer
PLCP	Physical Layer Convergence Protocol
P/S	Publisher/Subscriber
R/S	Report/Sink
SAP	Service Access Point
SM	Security Manager
SN	Sign Notation
TDMA	Time Division Multiple Access
UAO	User Application Object
UAP	User Application Process
VCR	Virtual Communication Relationship
WIA-FA	Wireless Network for Industrial Automation – Factory Automation

3.3 Conventions

For the purposes of the state machines and state transitions in this document, the following conventions apply.

This document uses a diagram to represent a state machine. The conventions used in the state machines are shown in Figure 1.



IEC

Figure 1 – Conventions used for state machines

The conventions used in the state machines are as follows:

- The labeled circles (S1, S2, …) represent states that a device can be in; the solid cycle (◉) represents the end state in which a device terminates its operations.
- State transitions are directed lines. They show which state a component leaves and which state it enters.
- A transition is labeled by (T1, T2, …) standing for the events that trigger it and the corresponding actions (if present).

The conventions used in the state transitions are shown in Table 1.

Table 1 – Conventions used for state transitions

#	Current state	Event/conditions =>action	Next state
Name of transition	The current state to which this state transition applies	Events or conditions that trigger this state transaction. => The actions that are taken when the above events or conditions are met. The actions are always indented below events or conditions	The next state after the actions in this transition is taken

The conventions used in the state transitions are as follows:

- “:=”represents that value of an item on the left is replaced by value of an item on the right. If an item on the right is a parameter, it comes from the primitive shown as an input event.
- “==" represents a logical condition indicating that an item on the left is equal to an item on the right.
- “&&”represents logical "AND"
- “||”represents logical "OR"
- “!=”represents a logical condition indicating that an item on the left is not equal to an item on the right.

This construct allows the execution of a sequence of actions in a loop within one transition. The loop is executed for all values from start_value to end_value.

EXAMPLE 1

```
for (Identifier = start_value to end_value)
    actions
end
```

This construct allows the execution of alternative actions depending on some condition (which may be the value of some identifier or the outcome of a previous action) within one transition.

EXAMPLE 2

```
If (condition)
    actions
else if (condition)
    actions
else
    actions
```

4 Data coding

4.1 Overview

WIA-FA data coding specifies the machine independent syntax for the data conveyed by each layer services. WIA-FA supports the definition and transfer of both basic and structured data types.

Basic types are atomic types that cannot be decomposed into more elemental types. Structured types are types composed of basic types and other structured types. Their complexity and depth of nesting is not constrained by this document.

4.2 Basic data type coding

4.2.1 Integer coding

The integer is a signed value, as shown in Figure 2. An example of Integer16 coding is shown in Table 2. For this data type, the MSB of the most significant octet is transmitted first.

<table>
<tr><th colspan="3">Notations: Integer8, Integer16, Integer24, Integer32</th></tr>
<tr><th>Data type</th><th>Value range</th><th>Length</th></tr>
<tr><td>Integer8</td><td>$-128 \le i \le 127$</td><td>One octet</td></tr>
<tr><td>Integer16</td><td>$-32\ 768 \le i \le 32\ 767$</td><td>Two octets</td></tr>
<tr><td>Integer24</td><td>$-2^{23} \le i \le 2^{23} - 1$</td><td>Three octets</td></tr>
<tr><td>Integer32</td><td>$-2^{31} \le i \le 2^{31} - 1$</td><td>Four octets</td></tr>
<tr><td colspan="3">Two’s complement binary notation
MSB is the bit followed by the first octet (SN)
SN = 0: positive and 0
SN = 1: negative</td></tr>
</table>

IEC

Figure 2 – Integer coding

Table 2 – Integer16 coding

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	SN	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
2	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

4.2.2 Unsigned coding

The unsigned value is coded as shown in Figure 3. An example of Unsigned16 coding is shown in Table 3. For this data type, the MSB of the most significant octet is sent first.

<table>
<tr><th colspan="3">Notations: Unsigned8, Unsigned16, Unsigned24, Unsigned32, Unsigned40, Unsigned48, Unsigned64, Unsigned80</th></tr>
<tr><th>Data type</th><th>Value range</th><th>Length</th></tr>
<tr><td>Unsigned8</td><td>$0 \le i \le 255$</td><td>One octet</td></tr>
<tr><td>Unsigned16</td><td>$0 \le i \le 65\ 535$</td><td>Two octets</td></tr>
<tr><td>Unsigned24</td><td>$0 \le i \le 2^{24} - 1$</td><td>Three octets</td></tr>
<tr><td>Unsigned32</td><td>$0 \le i \le 2^{32} - 1$</td><td>Four octets</td></tr>
<tr><td>Unsigned40</td><td>$0 \le i \le 2^{40} - 1$</td><td>Five octets</td></tr>
<tr><td>Unsigned48</td><td>$0 \le i \le 2^{48} - 1$</td><td>Six octets</td></tr>
<tr><td>Unsigned64</td><td>$0 \le i \le 2^{64} - 1$</td><td>Eight octets</td></tr>
<tr><td>Unsigned80</td><td>$0 \le i \le 2^{80} - 1$</td><td>Ten octets</td></tr>
</table>

IEC

Figure 3 – Unsigned coding

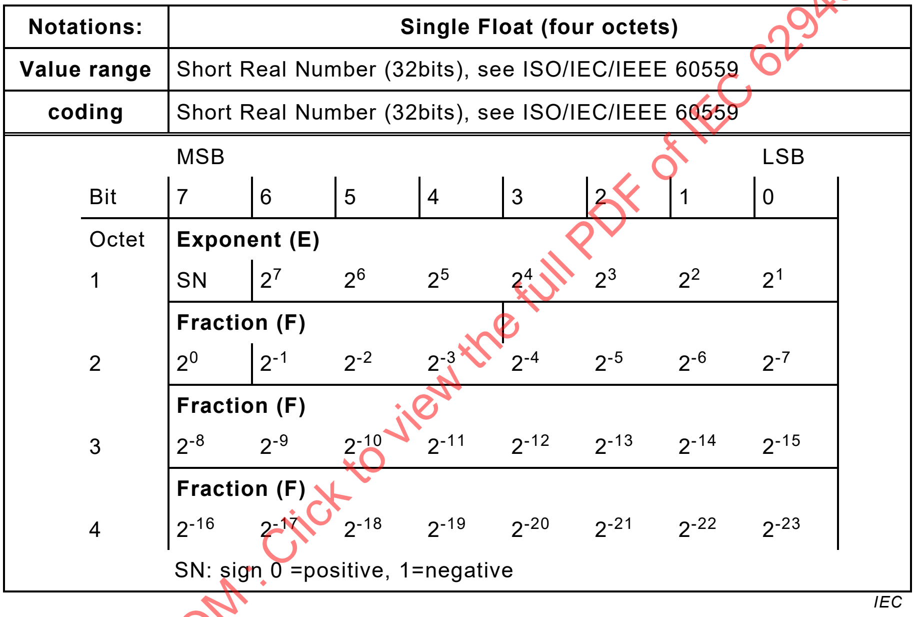
Table 3 – Unsigned16 coding

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}
2	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

4.2.3 Float coding

The value of floating point is coded as shown in Figure 4 and Figure 5.

The MSB of the sign and the exponent is sent first, and then the remained bits of the exponent and the bits from MSB to LSB of the fraction. If the value of the floating point data is unknown, 0x7F and 0xA0 are firstly sent, followed by 0x00, which means “Not-a-number”.



Notations:	**Single Float (four octets)**
Value range	Short Real Number (32bits), see ISO/IEC/IEEE 60559
coding	Short Real Number (32bits), see ISO/IEC/IEEE 60559

	MSB							LSB
Bit	7	6	5	4	3	2	1	0
Octet	**Exponent (E)**							
1	SN	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}
	Fraction (F)							
2	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
	Fraction (F)							
3	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}
	Fraction (F)							
4	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}	2^{-23}

SN: sign 0 =positive, 1=negative

IEC

Figure 4 – Single float coding

Notations:	Double Float (eight octets)
Value range	Short Real Number, 64 bits, see ISO/IEC/IEEE 60559
coding	Short Real Number, 64 bits, see ISO/IEC/IEEE 60559

	MSB							LSB
bit	7	6	5	4	3	2	1	0
octet	**Exponent (E)**							
1	SN	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}
	Exponent (E)				**Fraction (F)**			
2	2^{3}	2^{2}	2^{1}	2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	Fraction (F)							
3	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}
	Fraction (F)							
4	2^{-13}	2^{-14}	2^{-15}	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}
	Fraction (F)							
5	2^{-21}	2^{-22}	2^{-23}	2^{-24}	2^{-25}	2^{-26}	2^{-27}	2^{-28}
	Fraction (F)							
6	2^{-29}	2^{-30}	2^{-31}	2^{-32}	2^{-33}	2^{-34}	2^{-35}	2^{-36}
	Fraction (F)							
7	2^{-37}	2^{-38}	2^{-39}	2^{-40}	2^{-41}	2^{-42}	2^{-43}	2^{-44}
	Fraction (F)							
8	2^{-45}	2^{-46}	2^{-47}	2^{-48}	2^{-49}	2^{-50}	2^{-51}	2^{-52}

SN: sign 0 =positive, 1=negative

IEC

Figure 5 – Double float coding

4.2.4 Octetstring coding

The Octetstring coding method is shown in Table 4. For the data with N octets, the MSB of the most significant octet is first sent.

Table 4 – Octetstring coding

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	2^{8N-1}	2^{8N-2}	2^{8N-3}	2^{8N-4}	2^{8N-5}	2^{8N-6}	2^{8N-7}	2^{8N-8}
2	2^{8N-9}	2^{8N-10}	2^{8N-11}	2^{8N-12}	2^{8N-13}	2^{8N-14}	2^{8N-15}	2^{8N-16}
…	…	…	…	…	…	…	…	…
N	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

4.2.5 BitField coding

BitField data are used to encode single-bit data with fixed length. BitFieldn indicates BitField with n-bit length. The bit coding method is shown in Table 5, Table 6 and Table 7. This data type is defined as a series of eight bits. For the data with more than one octet, the most significant octet is first sent. There is a corresponding definition table for each bit of the data with BitField data type.

Table 5 – Coding of BitField8 data with one octet

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	7	6	5	4	3	2	1	0

Table 6 – Coding of BitField16 data with two octets

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	15	14	13	12	11	10	9	8
2	7	6	5	4	3	2	1	0

Table 7 – Coding of BitField24 data with three octets

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	23	22	21	20	19	18	17	16
2	15	14	13	12	11	10	9	8
3	7	6	5	4	3	2	1	0

4.2.6 Bitstring coding

Bitstring data are used to encode single-bit data with flexible length. The bit coding method is shown in Table 8. This data type is defined as a sequence of bits, and its bit length shall be specified concomitantly. Bitstring data are packed into octets and transferred as an octet sequence over network. The amount of octets is the maximum number of octets that can contain all bit values. For the data with more than one octet, the most significant octet of data is first sent.

Table 8 – Bitstring coding

octet	bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1	X[a]	X	X	X	X	X	X	X
2	X	X	X	X	X	X	X	X
3	X	X	X	X	X	X	X	X
…[b]	…[c]							

[a] MSB of the Bitstring data

[b] the amount of octets is the maximum number of octets that contain all bit values.

[c] the position of last bit (LSB) = 8 – bit length % 8

4.2.7 TimeData coding

This data type is an unsigned integer with a length of 64 bits, indicating time incremented by 1 µs.

4.2.8 KeyData coding

This data type is an unsigned integer with a length of 128 bits.

4.3 Structured data type coding

4.3.1 Structure type coding

A structure is made of an ordered set of heterogeneously different typed data called members, which may be basic data type or structured data type. The member of a structure is identified by MemberID. The structure data may be accessed as a whole, or just one of its members may be accessed by specifying the MemberID.

4.3.2 List type coding

A list is composed of an ordered set of homogeneously same typed elements called records, which may be basic data type or structured data type. The record of a list is identified by index. The list data may be accessed as a whole, or one or several of its records may be accessed by specifying the starting FirstStoreIndex and Count.

5 WIA-FA overview

5.1 Device types

5.1.1 General

This document specifies five types of WIA-FA devices:

a) host computer (HC);

b) gateway device (GW);

c) access device (AD);

d) field device (FD);

e) handheld device (HD).

To improve availability and reliability, WIA-FA network allows the existence of redundant gateway device as a hot backup, and multiple access devices work in parallel.

NOTE Redundant gateway device is out of the scope of this document.

5.1.2 Host computer

Host computer is used to configure and monitor WIA-FA network for the operation, maintenance and management personnel. In addition, host computer may carry out controlling functions. Its implementation is out of the scope of this document.

5.1.3 Gateway device

The main functions of the gateway device shall be as follows:

- Providing interconnection interface between WIA-FA network and other external networks, such as fieldbus, using data mapping and protocol conversion means;
- Responsible for network management and security management;
- Communicating with field devices through access devices, and exchanging information between field devices;
- Acting as the unique clock source in the WIA-FA network for network time synchronization;
- There is only one gateway device in one WIA-FA network with the unique NetworkID (see Table 16 for NetworkID).

5.1.4 Access device

The main functions of the access device shall be as follows:

- Receiving data from field devices and forwarding them to the gateway device;
- Forwarding the control commands of gateway device to field devices;
- Forwarding the management information and configuration information of the gateway device to field devices;
- Receiving alarms and network management information of field devices and forwarding them to the gateway device.

NOTE Access devices and the gateway device are connected by wire (see Clause 9). Their synchronization method is out of the scope of this document .

5.1.5 Field device

Field devices shall be installed in industrial field and connect sensors and actuators. Field devices shall send field application data and alarms to the gateway device, as well as receive configuration information, management information and control commands from the gateway device. The power supply modes of field devices include wired power supply, battery power supply, etc., which are out of the scope of this document.

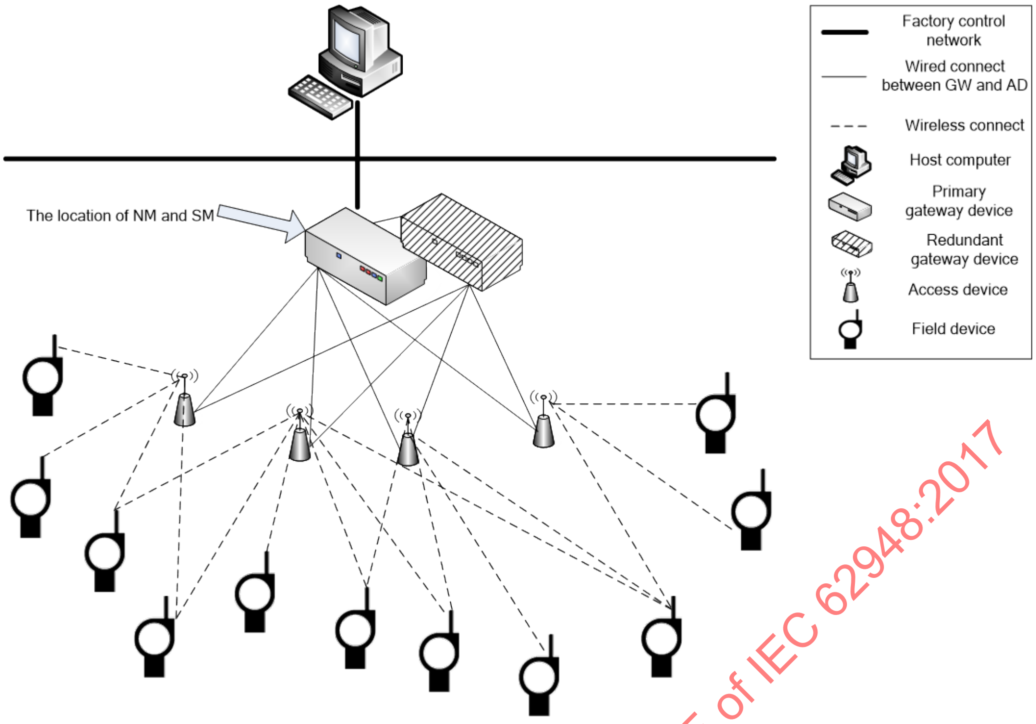
5.1.6 Handheld device

Handheld devices are portable devices that shall be used for provisioning field devices, access devices and the gateway device. A handheld device shall communicate only with its direct connected device using wired maintenance port. Provision parameters may include security level, join keys, shared keys (except security level 0), and network Identifier (ID).

NOTE Wired maintenance ports used by a handheld device could be RS232, RS485, USB, etc.

5.2 Network topology

As shown in Figure 6, WIA-FA supports redundant star topology. The center is comprised of one gateway device (redundant gateway device may exist) and one or several access devices, which means multiple access devices may communicate with the field devices in parallel and form multiple star topologies. These access devices shall have the same address (see 6.3) and are transparent for the field devices. Each access device forms a star topology with the field devices. See 8.1.4 for communication details of redundant star topology.



IEC

Figure 6 – WIA-FA redundant star topology

5.3 Protocol architecture

The WIA-FA network protocol is based on the ISO/IEC 7498-1 OSI reference model. The WIA-FA network protocol defines the Physical Layer (PHY), Data Link Layer (DLL) and Application Layer (AL). Figure 7 shows the mapping between WIA-FA and OSI basic reference model.

OSI layer	Function	WIA-FA
Application	Provides the user with network capable application	Distributed application services
Presentation	Converts between application layer data and the lower layer data formats	↑
Session	Connection management services	↑
Transport	Provides network independent, transparent message transfer	↓ and ↑
Network	Resolving network addresses, End- to- end routing of packets.	↓ and ↑
Data link	Establishes data packet structure, framing, error detection, bus arbitration	DLL Communication based on multiple ADs, TDMA, FDMA, retransmission, aggregation...
Physical	Mechanical / electrical connection. Transmits raw bit stream	PHY (IEEE STD 802.11-2012 PHY)

NOTE ↓ and ↑ indicate that the functionality of this layer, when present, can be included in the protocol layer that is nearest in the direction of the arrow. Thus network and transport functionality can be included in data link layer or application layer, while session and presentation functionality can be included in application layer, not in data link layer.

IEC

Figure 7 – OSI basic reference model mapped to WIA-FA

Figure 8 shows the protocol architecture of WIA-FA. The protocol architecture of WIA-FA shall include the following components:

- Protocol layers: including PHY, DLL, AL; AL is comprised of ASL, UAP and DMAP;
- Entities of protocol layers: including data entities (DLDE and ASLDE) and management entities (DLME and ASLME);
- Protocol layer interfaces: including data entity SAPs (DLDE-SAP and ASLDE-SAP) and management entity SAPs (DLME-SAP and ASLME-SAP).

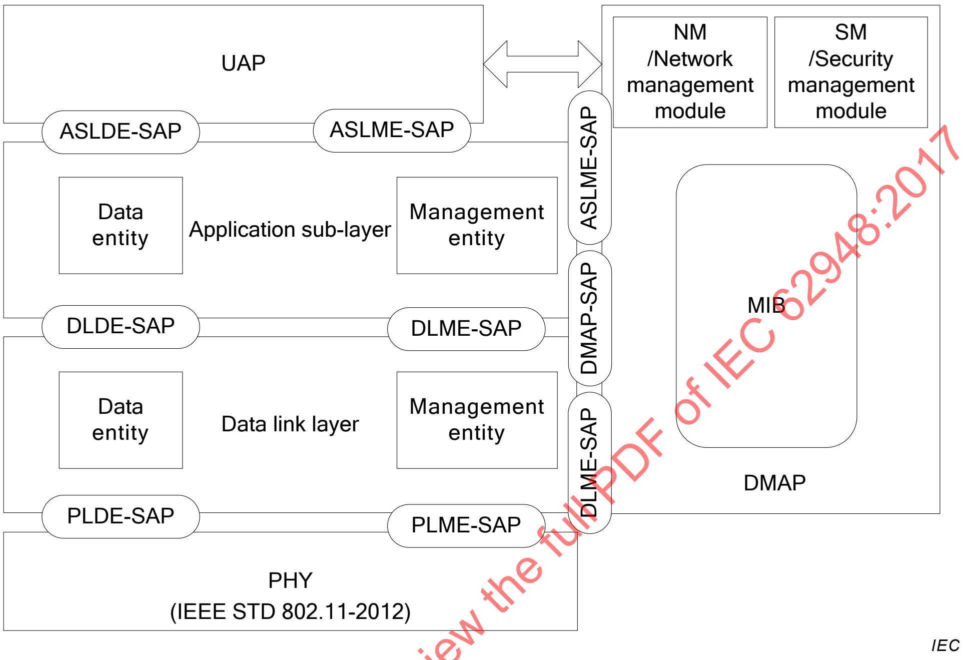


Figure 8 – Protocol architecture of WIA-FA

The data flow over WIA-FA network is shown in Figure 9.

- A field device shall have PHY, DLL, and AL;
- An access device shall have PHY and DLL; an access device shall connect with the gateway device by wires;
- The gateway device shall only have AL and part of DLL. The AL of the gateway device shall communicate with AL of WIA-FA field devices. The AL of the gateway device may also communicate with AL of the plant network and translate ALs between the WIA-FA network and the plant network.

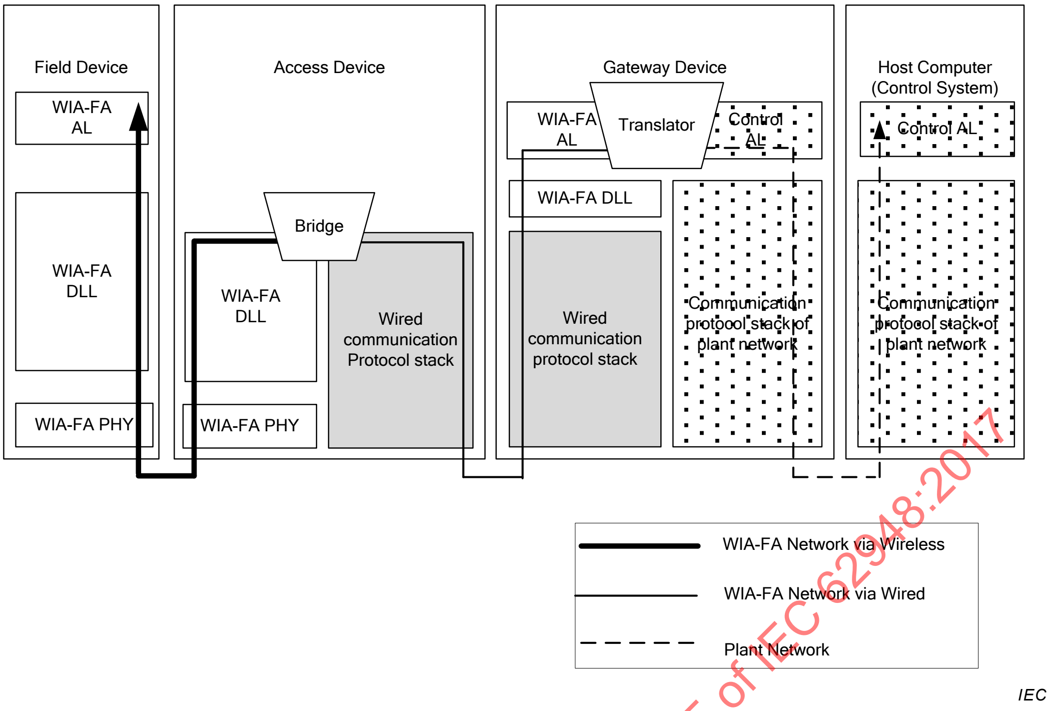

Field Device
WIA-FA AL
WIA-FA DLL
WIA-FA PHY
Access Device
Bridge
WIA-FA DLL
Wired communication Protocol stack
WIA-FA PHY
Gateway Device
WIA-FA AL
Translator
Control AL
WIA-FA DLL
Wired communication protocol stack
Communication protocol stack of plant network
Host Computer (Control System)
Control AL
Communication protocol stack of plant network
WIA-FA Network via Wireless
WIA-FA Network via Wired
Plant Network
IEC

Figure 9 – Data flow over WIA-FA network

6 System management

6.1 Overview

The WIA-FA network adopts centralized management framework, as shown in Figure 10. The system management shall be implemented by the network manager and security manager in the gateway device, and the network management modules and security management modules in field devices and access devices. The network manager and security manager are responsible for managing the access devices and field devices. Network management modules and security management modules shall be implemented in field devices and access devices, which perform management functions together with the gateway device.

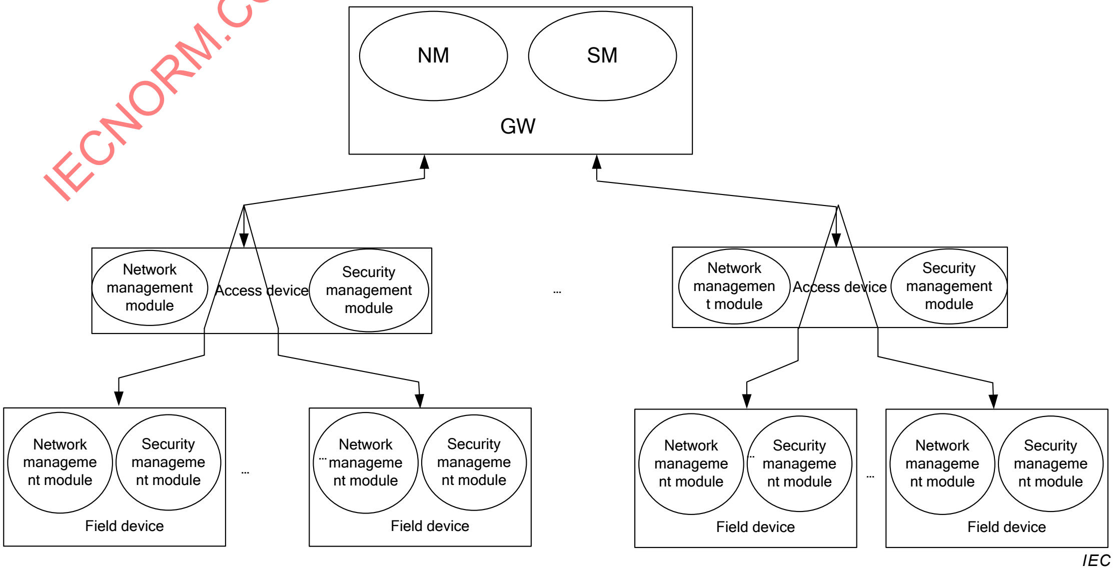

NM
SM
GW
Network management module
Access device
Security management module
...
Network managemen t module
Access device
Security management module
Network manageme nt module
Security manageme nt module
Field device
...
Network manageme nt module
Security manageme nt module
Field device
Network manageme nt module
Security manageme nt module
Field device
...
Network manageme nt module
Security manageme nt module
Field device
IEC

Figure 10 – System management scheme

6.2 Device Management Application Process

6.2.1 General

The functions of system management in WIA-FA network shall be implemented by the Device Management Application Process (DMAP) in each device. The DMAP is a particular User Application Process (UAP) managing devices and providing MIB access services. The position and component of the DMAP in the protocol architecture are shown in Figure 11. The gray part is the DMAP, and the white parts are function modules in DMAP. DMAP shall include:

- Network Manager (NM) of GW or the network management modules of ADs and FDs;
- Security Manager (SM) of GW or the security management modules of ADs and FDs;
- Management Information Base (MIB), storing all the attributes for network management and security management in WIA-FA network.

DMAP shall use ASLME-SAP and DLME-SAP to interact with ASL and DLL respectively, as shown in Figure 11.

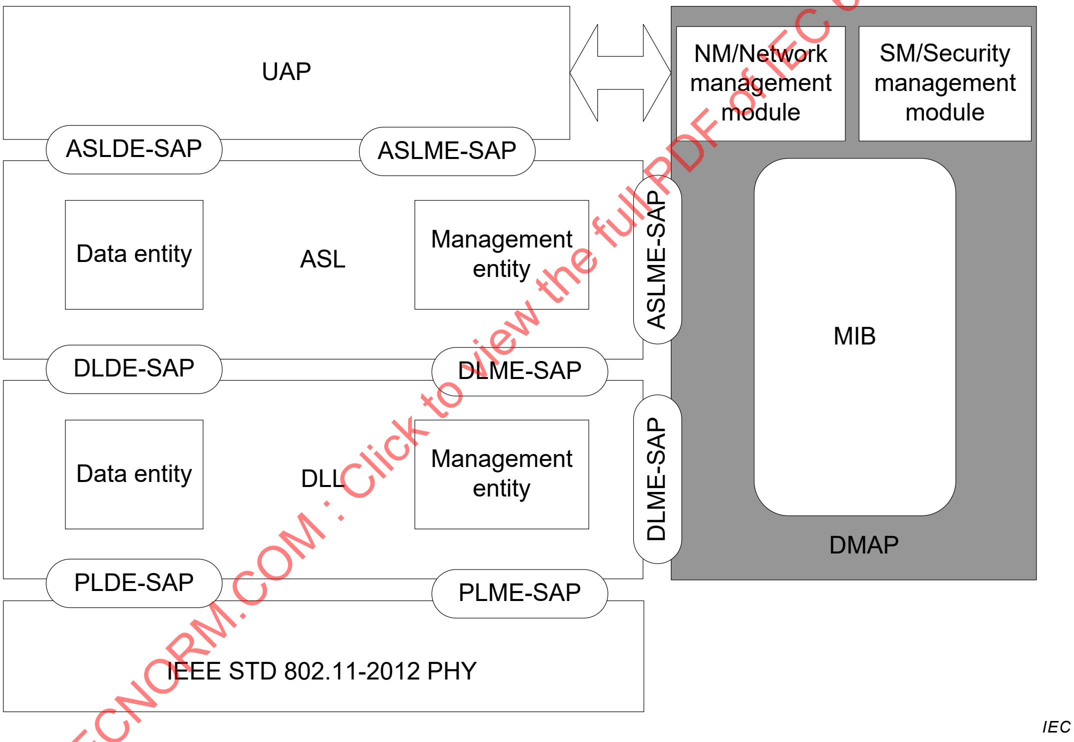


Figure 11 – DMAP of management system

The network management and security management functions implemented in WIA-FA network are shown in Table 9 and Table 10.

Table 9 – Network management functions

Network management functions	Descriptions
Network establishment	Initialization: initializing the NM/network management modules and starting the network
	Clock source configuration and time synchronization service; the WIA-FA network shall set up one reference clock source, which is performed by the gateway device. Devices in the network synchronize with the gateway device.
	Device join process management: the devices need NetworkID (see Table 16) before joining. A new device invokes the join process by sending a join request; after the authentication of SM, the NM returns the join response.
	Network address allocation: Each device in the WIA-FA network shall have a global unique 64-bit address which is called long address and an 8-bit or 16-bit network address which is called short address. The long address of each device is assigned by vendors according to the 64-bit Extended Unique Identifier (EUI-64). The short address of each network device is assigned by the NM.
	Topology management: forming and maintaining the redundant star topology as shown in Figure 6.
	Network configuration management: configuring the information of all network devices by the NM, including the communication resources, network address and other network attributes.
Network scheduling and communication resource allocation	Superframe establishment: establishing the superframes for communication according to the application configuration.
	Communication resource allocation: allocating the communication resources to the links.
	Activation/deactivation: activating and deactivating the superframes and links according to application process.
Network diagnosis and performance monitoring	Channel management: monitoring and maintaining the channel list and condition.
	Device health status management: monitoring and maintaining the health status of each device.
Leaving	Device leave process management: the leave process of field devices and access devices includes passive leaving and abnormal leaving. The passive leaving is invoked by the gateway device; the field devices and access devices leave the network after receiving the leave request; the gateway device releases the communication resources of field devices. The abnormal leaving is detected and processed by the gateway device.

Table 10 – Security management functions

Security management functions	Descriptions
Secure network establishment and configuration	Initially key configuration:
	Secure join process: when a new field device invokes the secure join process, SM shall authenticate the field device and return the result to NM; NM shall return a join response carrying the authentication result.
	Key establishment: after FD secure joining, SM shall generate and distribute keys including KEK, KEDB, and KEDU to make secure operation during normal operating process.
Key updating	Key update: SM shall update the keys in use before the ends of their lifetime, including KEK, KEDB, and KEDU.
Security performance monitoring	Security alarm: monitoring the update status of keys and attacked counts of keys.

6.2.2 Network manager

Network Manager (NM) shall implement network management function in the gateway device. One WIA-FA network has only one NM.

NM shall mainly perform the following functions:

- allocating the unique 8-bit or 16-bit short address for all devices in the network (see 6.3);
- constructing and maintaining the redundant star topology;
- allocating communication resources for communications of WIA-FA devices;
- monitoring the status of the WIA-FA network, including device status and channel condition.

6.2.3 Security manager

Security Manager (SM) shall implement the function of security management in the gateway device, and one WIA-FA network has only one SM. SM communicates directly with NM.

SM shall mainly perform the following functions:

- authenticating the field devices attempting to join the WIA-FA network;
- managing keys in the WIA-FA network, including key establishment and update (see 11.4);
- handling security alarm.

6.2.4 Network management module

Network management modules in field devices and access devices shall maintain the information needed for communication.

The main functions of the network management module shall be as follows:

- coordinating with NM to construct and maintain the redundant star topology;
- coordinating with NM to allocate communication resources for devices;
- coordinating with NM to monitor the status of the WIA-FA network, including device status and channel condition.

6.2.5 Security management module

Security management modules in field devices and access devices shall maintain the information for security management functions, and implement the following security management functions:

- coordinating with SM to handle secure joining (field devices only);
- coordinating with SM to manage keys;
- coordinating with SM to report security alarm.

6.2.6 DMAP state machines

6.2.6.1 DMAP state machine of gateway device

The DMAP state machine of the gateway device is shown in Figure 12, which includes Init and Active states. The gateway device shall enter into Active state if it finishes initializations.

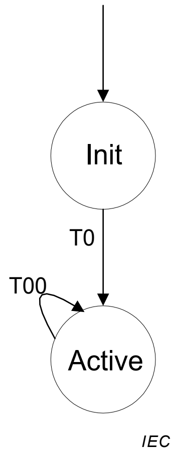


Figure 12 – DMAP state machine of gateway device

The DMAP state transitions of gateway device are shown in Table 11.

Table 11 – DMAP state transition of gateway device

#	Current State	Event or condition => action	Next state
T0	Init	IsDMAPInitializationDone() == TRUE => ;	Active
T00	Active		Active

When the DMAP of gateway device is in Active state and receives a join request from a new field device, it shall create and maintain a sub-state machine for this field device, as shown in Figure 13, so as to process multiple packets from multiple field devices concurrently.

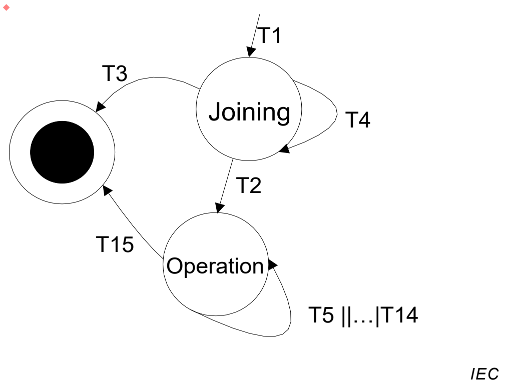


Figure 13 – DMAP sub-state machine of gateway device for each field device

The DMAP sub-state transition of gateway device for each field device is listed in Table 12.

Table 12 – DMAP sub-state transition of gateway device for each field device

#	Current State	Event or condition => action	Next state

T1		PrimitiveType == DLME-JOIN.indication
		=>
		JoinResult:= HandleFieldJoinRequest();
		If (JoinResult == SUCCESS)
		{
		ShortAddrResult:= AllocateShortAddr(Addr);
		DLME-JOIN.response(Status:= SUCCESS,
		ShortAddr);
		Delay(DefaultSuperFrameTime);
		DLME-INFO-GET.request(DeviceStatus);
		Start DeviceStatusConfirmTimer;
		}
			Joining

T2	Joining	PrimitiveType == DLME-INFO-GET.confirm
		&&Status == SUCCESS
		&&DeviceStatus == Operation
		=>
		If （SECLEVEL != 0 ）
		{
		KEY-ESTABLISH.request(DstAddr, KeyMaterial);
		}
			Operation

T3	Joining	JoinResult != SUCCESS
		|| (DeviceStatusTimer expires)&&(No DLME-INFO-GET.confirm)
		|| ((PrimitiveType ==DLME-INFO-GET.confirm)
		&&((Status != SUCCESS)||(DeviceStatus != Operation)))
		=>
		If (JoinResult != SUCCESS)
		{
		DLME-JOIN.response(JoinResult, ShortAddr);
		}
			End

T4	Joining	PrimitiveType != DLME-INFO-GET.confirm => Ignore;	Joining

T5	Operation	IsHostComputerConfigureDone() == TRUE
		=>
		ResAllocAgrithm (SuperframeList, LinkList);
		If (AllocResult == SUCCESS) {
		DLME-INFO-SET.request(VCR_ID);
		DLME-INFO-SET.request(SuperframeList);
		DLME-INFO-SET.request(LinkList);
		}
			Operation

#	Current State	Event or condition => action	Next state
T6	Operation	IsHostComputerSet MIB() == TRUE => DLME-INFO-SET.request(Handle, DstAddr, AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, AttributeValue);	Operation
T7	Operation	PrimitiveType ==DLME-INFO-SET.confirm => IndicateSetMIBResult(Handle, Status);	Operation
T8	Operation	IsHostComputerGet MIB() == TRUE => DLME-INFO-GET.request(Handle, DstAddr, AttributeID, MemberID, FirstStoreIndex, Count);	Operation
T9	Operation	PrimitiveType ==DLME-INFO-GET.confirm => IndicateGetMIBResult(Handle, Status);	Operation
T10	Operation	PrimitiveType == DLME-CHANNEL-STATUS.indication => HandleChannelStatusReport(Addr, ChannelConditonInfo);	Operation
T11	Operation	PrimitiveType == DLME-DEVICE-STATUS.indication => HandleDeviceStatusReport(PowerSupplyStatus);	Operation

T12	Operation	IsHostComputerRequestDeviceLeave() == TRUE
		=>
		DLME-LEAVE.request(DeviceAddr);
		IndicateHostComputerLeaveResult(Addr);
			Operation

T13	Operation	PrimitiveType ==DLME-LEAVE.confirm && Status == FAILURE =>	Operation
T14	Operation	PrimitiveType != DLME-INFO-SET.confirm && PrimitiveType != DLME-INFO-SET.confirm && PrimitiveType != DLME-CHANNEL-STATUS.indication && PrimitiveType !=DLME-LEAVE.confirm && PrimitiveType != DLME-DEVICE-STATUS.indication =>	Operation
T15	Operation	((PrimitiveType ==DLME-LEAVE.confirm)&&(Status == SUCCESS)) || ((LossConnectDuration time out) &&(receive no frame from this field device)) => ReleaseResources(Addr); IndicateHostComputerLeaveResult(Addr);	End

– Joining state

In Joining state, the DMAP of the gateway device shall handle join request received from the field device, which includes authentication and assignment of short address for the device. If authentication or short address assignment fails, DMAP shall invoke DLME-JOIN.response to notify the field device of the failure result, then the DMAP sub-state machine shall enter to End state; if authentication and short address assignment succeeds, the DMAP of the gateway device shall invoke DLME-JOIN.response to notify the field device that the join process is successful; then remotely get DeviceStatus in the device if DeviceStatus is equal to Operation, the DMAP sub-state machine shall enter to Operation state.

– Operation state

In Operation state, the following events and responding actions may occur:

- If host Computer finishes Configuration, DMAP of the gateway device shall allocate communication resources to field devices; DMAP shall invoke DLME-INFO-SET.request to remotely write VCR_ID, superframes and links to field devices.
- If it is necessary to remotely set the field device's MIB attributes, DMAP shall invoke DLME-INFO-SET.request to request DLL generating a Remote attribute set request frame (see 8.4.17).
- If DLL invokes DLME-INFO-SET.confirm, DMAP shall handle the result of remote attribute set operation.
- If it is necessary to remotely get the field device's MIB attributes, DMAP shall invoke DLME-INFO-GET.request to request DLL generating a Remote attribute get request frame (see 8.4.15).
- If DLL invokes DLME-INFO-GET.confirm, DMAP shall get and handle the returned MIB attributes of a field device.
- If DLL invokes DLME-DEVICE-STATUS.indication, DMAP shall handle the device status report from a field device.
- If DLL invokes DLME-CHANNEL-STATUS.indication, DMAP shall handle the channel condition report from a field device.
- If it is necessary to request a field device to leave WIA-FA network. DMAP shall invoke DLME-LEAVE.request to indicate DLL to generate a Leave command frame (see 8.4.9).

If DLL invokes DLME-LEAVE.response with Status, DMAP shall handle the result of requesting a field leaving WIA-FA network. If Status is equal to SUCCESS, or LossConnectDuration times out and the abnormal leaving of a field is detected, DMAP shall release all MIB attributes and communication resources that are occupied by a leaving field device, then the DMAP sub-state machine enters to End state.

6.2.6.2 DMAP state machine of field device

The DMAP state machine of field device is shown in Figure 14.

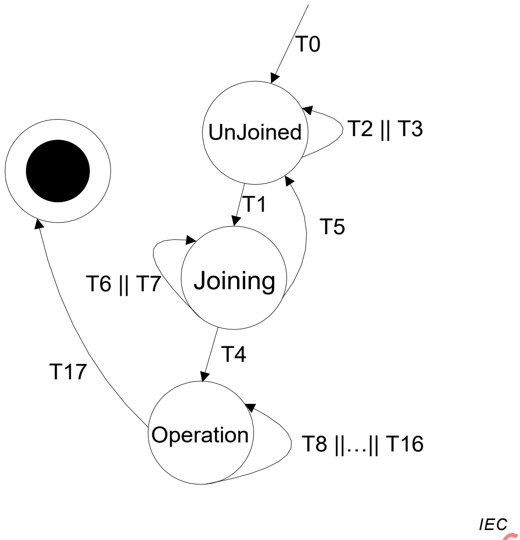


Figure 14 – DMAP state machine of a field device

The DMAP state transition of a field device is shown in Table 13.

Table 13 – DMAP state transition of a field device

#	Current State	Event or condition => action	Next state
T0	-	IsDMAPInitializationDone() == TRUE => DLME-DISCOVERY.request(ScanChannels, ProbeTime);	UnJoined
T1	UnJoined	PrimitiveType == DLME-DISCOVERY.confirm && Status == SUCCESS => DeviceStatus:= Joining; DLME-JOIN.request(NetworkID, Channel, PhyAddr, SecMaterial);	Joining
T2	UnJoined	PrimitiveType == DLME-DISCOVERY.confirm &&Status == NO_BEACON => DLME-DISCOVERY.request(ScanChannels, ProbeTime);	UnJoined
T3	UnJoined	PrimitiveType != DLME-DISCOVERY.confirm => Ignore;	UnJoined

T4	Joining	PrimitiveType == DLME-JOIN.confirm
		&& Status == SUCCESS
		=>
		DeviceStruct.ShortAddr:= ShortAddr;
		DeviceStatus:= Operation;
		If (TwoWayTimeSyn == TRUE)
		{
		DLME-TIME-SYN.request();
		}
			Operation

#	Current State	Event or condition => action	Next state

T5	Joining	PrimitiveType == DLME-JOIN.confirm && Status != SUCCESS
		=>
		DeviceStatus:= UnJoined;
		DLME-DISCOVERY.request(ScanChannels, ProbeTime);
			UnJoined

T6	Joining	PrimitiveType ==DLME-INFO-GET.indication
		=>
		DMAP-MIB-GET.request (Handle, AttributeID, MemberID, FirstStroeIndex, Count, AttributeValue);
		DMAP-MIB-GET.confim(Handle, Status, Count, AttributeValue);
		DLME-INFO-GET.response(DstAddr, Status, Count, FirstStoreIndex, Count, AttributeValue);
			Joining

T7	Joining	PrimitiveType != DLME-JOIN.confirm &&PrimitiveType !=DLME-INFO-GET.indication => Ignore;	Joining
T8	Operation	PrimitiveType == DLME-TIME-SYN.confirm && Status == SUCCESS => HandleTwoWaySyn();	Operation
T9	Operation	PrimitiveType == DLME-TIME-SYN.confirm && Status != SUCCESS => DLME-TIME-SYN.request();	Operation

T10	Operation	PrimitiveType == DLME-INFO-SET.indication
		=>
		DMAP-MIB-SET.request (Handle, AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, AttributeValue);
		DMAP-MIB-SET.confim(Handle, Status);
		DLME-INFO-SET.response(SrcAddr , AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, Status);
			Operation

T11	Operation	PrimitiveType ==DLME-INFO-GET.indication
		=>
		DMAP-MIB-GET.request (Handle, AttributeID, MemberID, FirstStroeIndex, Count, AttributeValue);
		DMAP-MIB-GET.confim(Handle, Status, Count, AttributeValue);
		DLME-INFO-GET.response(DstAddr, Status, Count, FirstStoreIndex, Count, AttributeValue);
			Operation

T12	Operation	DevStaRptCycle timeout => DLME-DEVICE-STATUS.request(PowerSupplyStatus);	Operation
T13	Operation	PrimitiveType == DLME-DEVICE-STATUS.confirm =>	Operation
T14	Operation	(ChaStaRptCycle timeout) => DLME-CHANNEL-STATUS.request(ChannelConditionInfo);	Operation

#	Current State	Event or condition => action	Next state
T15	Operation	PrimitiveType == DLME-CHANNEL-STATUS.confirm =>	Operation
T16	Operation	PrimitiveType != DLME-INFO-SET.indication && PrimitiveType !=DLME-INFO-GET.indication && PrimitiveType != DLME-CHANNEL-STATUS.confirm &&PrimitiveType != DLME-DEVICE-STATUS.confirm &&PrimitiveType != DLME-LEAVE.indication && PrimitiveType != DLME-TIME-SYN.confirm => Ignore;	Operation

T17	Operation	PrimitiveType == DLME-LEAVE.indication
		=>
		DLME-LEAVE.response();
		ReleaseResources(Addr);
		DeviceStatus:= End;
			End

– UnJoined state

After the DMAP of field device finishes initialization, the state machine shall enter to UnJoinedstate. The DMAP of the field device shall invoke DLME-DISCOVERY.request to scan WIA-FA network. DLL invokes DLME-DISCOVERY.confirm to return the result of network scanning. If the result is successful, the DMAP state machine of a field device enters to Joining state; otherwise, the DMAP state machine of a field device still stays in UnJoined state and restarts the network discovery.

– Joining state

In Joining state, the DMAP of a field device shall invoke DLME-JOIN.request to try to join the network; DLL invokes DLME-JOIN.confirm to return the joining results. If the result indicates the join process is successful, the state machine enters to Operation state; otherwise, the state machine enters to UnJoinedstate.

– Operation state

In Operation state, the following events and responding actions may occur:

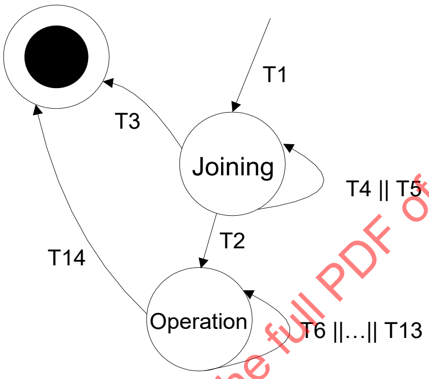
- If the state machine is entering to Operation State, DMAP of the field device shall invoke DLME-TIME-SYN.request to request a two way synchronization.
- If DLL invokes DLME-TIME-SYN.confirm after it receives a two way synchronization response frame, and if the status is SUCCESS, the DMAP of the field device shall handle the two way synchronization, otherwise, the DMAP of the field device invoke DLME-TIME-SYN.request to restart the two way synchronization procedure.
- If DLL invokes DLME-INFO-SET.indication after it receives a Remote attribute set request frame (see 8.4.17). The DMAP of the field device shall invoke DMAP-MIB-SET.request to locally set the MIB attributes properly and invoke DLME-INFO-SET.response with the set result which is indicated by DMAP-MIB-SET.confim.
- If DLL invokes DLME-INFO-GET.indication after it receives a remote attribute get request frame (see 8.4.15). The DMAP of the field device shall invoke DMAP-MIB-GET.request to locally get the MIB attributes properly and invoke DLME-INFO-GET.response with the requested MIB attributes which are indicated by DMAP-MIB-GET.confim.
- If the DevStaRptCycle (see 6.7.1.2.1) timer expires, the DMAP of the field device shall invoke DLME-DEVICE-STATUS.request to send the device status report.

- If DLL invokes DLME-DEVICE-STATUS.confirm, DMAP shall handle the result of sending device status report.
- If the ChaStaRptCycle (see 6.7.1.2.1) timer expires, the DMAP of the field device shall invoke DLME-CHANNEL-STATUS.request to send channel condition report.
- If DLL invokes DLME-CHANNEL-STATUS.confirm, DMAP shall handle the result of sending channel condition report.

  If DLL invokes DLME-LEAVE.indication after receiving a Leave request frame from the gateway device; the DMAP of the field device shall invoke DLME-LEAVE.response and release all MIB attributes and communication resources, then enters to End state.

6.2.6.3 DMAP state machine of access device

The DMAP state machine of access device is shown in Figure 15.



IEC

Figure 15 – DMAP state machine of an access device

The DMAP state transition of an access device is shown in Table 14.

Table 14 – DMAP state transition of an access device

#	Current State	Event or condition => action	Next state
T1	-	IsDMAPInitializationDone() == TRUE => DLME-JOIN.request(NetworkID, Channel, PhyAddr, SecMaterial);	Joining
T2	Joining	PrimitiveType == DLME-JOIN.confirm && Status == SUCCESS => DeviceStruct.ApId:= ApId; DeviceStatus:= Operation;	Operation
T3	Joining	PrimitiveType == DLME-JOIN.confirm && Status != SUCCESS => DeviceStatus:= End;	End

T4	Joining	PrimitiveType ==DLME-INFO-GET.indication
		=>
		DMAP-MIB-GET.request (Handle, AttributeID, MemberID, FirstStroeIndex, Count, AttributeValue);
		DMAP-MIB-GET.confim(Handle, Status, Count, AttributeValue);
		DLME-INFO-GET.response(DstAddr, Status, Count, FirstStoreIndex, Count, AttributeValue);
			Joining

#	Current State	Event or condition => action	Next state
T5	Joining	PrimitiveType != DLME-JOIN.confirm && PrimitiveType !=DLME-INFO-GET.indication => Ignore;	Joining
T6	Operation	PrimitiveType == DLME-TIME-SYN.indication => DLME-TIME-SYN.response(DstAddr, FieldDeviceTimeValue, ReceiveTimeValue);	Operation

T7	Operation	PrimitiveType == DLME-INFO-SET.indication
		=>
		DMAP-MIB-SET.request (Handle, AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, AttributeValue);
		DMAP-MIB-SET.confim(Handle, Status);
		DLME-INFO-SET.response(SrcAddr , AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, Status);
			Operation

T8	Operation	PrimitiveType ==DLME-INFO-GET.indication
		=>
		DMAP-MIB-GET.request (Handle, AttributeID, MemberID, FirstStroeIndex, Count, AttributeValue);
		DMAP-MIB-GET.confim(Handle, Status, Count, AttributeValue);
		DLME-INFO-GET.response(DstAddr, Status, Count, FirstStoreIndex, Count, AttributeValue);
			Operation

T9	Operation	DevStaRptCycle timeout => DLME-DEVICE-STATUS.request(PowerSupplyStatus);	Operation
T10	Operation	PrimitiveType == DLME-DEVICE-STATUS.confirm =>	Operation
T11	Operation	(ChaStaRptCycle timeout) => DLME-CHANNEL-STATUS.request(ChannelConditionInfo);	Operation
T12	Operation	PrimitiveType == DLME-CHANNEL-STATUS.confirm =>	Operation
T13	Operation	PrimitiveType != DLME-TIME-SYN.indication &&PrimitiveType != DLME-INFO-SET.indication && PrimitiveType !=DLME-INFO-GET.indication && PrimitiveType != DLME-CHANNEL-STATUS.confirm && PrimitiveType != DLME-DEVICE-STATUS.confirm && PrimitiveType != DLME-LEAVE.indication => Ignore;	Operation

T14	Operation	PrimitiveType == DLME-LEAVE.indication
		=>
		DLME-LEAVE.response();
		ReleaseResources(Addr);
		DeviceStatus:= End;
			End

– Joining state

After the DMAP of access device finishes initialization, the state machine shall enter to Joining state, the DMAP of an access device shall invoke DLME-JOIN.request to try to join the network; DLL invokes DLME-JOIN.confirm to return the joining results. If the result indicates the join process is successful, the state machine enters to Operation state; otherwise, the state machine enters to End state.

– Operation state

In Operation state, the following events and responding actions can occur:

- If DLL invokes DLME-TIME-SYN.indication after it receives a two way Time synchronization request frame, the DMAP of the access device shall invoke DLME-TIME-SYN.response.
- If DLL invokes DLME-INFO-SET.indication after it receives a Remote attribute set request frame (see 8.4.17). The DMAP of the access device shall invoke DMAP-MIB-SET.request to locally set the MIB attributes properly and invoke DLME-INFO-SET.response with the set result which is indicated by DMAP-MIB-SET.confim.
- If DLL invokes DLME-INFO-GET.indication after it receives a Remote attribute get request frame (see 8.4.15). The DMAP of the access device shall invoke DMAP-MIB-GET.request to locally get the MIB attributes properly and invoke DLME-INFO-GET.response with the requested MIB attributes which are indicated by DMAP-MIB-GET.confim.
- If the DevStaRptCycle (see 6.7.1.2.1) timer expires, the DMAP of the access device shall invoke DLME-DEVICE-STATUS.request to report the device status.
- If DLL invokes DLME-DEVICE-STATUS.confirm, DMAP shall handle the result of sending device status report.
- If the ChaStaRptCycle (see 6.7.1.2.1) timer expires, the DMAP of the access device shall invoke DLME-CHANNEL-STATUS.request to report channel condition.
- If DLL invokes DLME-CHANNEL-STATUS.confirm, DMAP shall handle the result of sending channel condition report.

If DLL invokes DLME-LEAVE.indication after receiving a Leave request frame from the gateway device; the DMAP of the access device shall invoke DLME-LEAVE.response and release all MIB attributes and communication resources, then enters to End state.

6.2.6.4 Functions used in DMAP state machines

The functions used in DMAP state machines of the gateway device are listed in Table 15.

Table 15 – Functions used in DMAP state machines

Function	Input	Output	Description
AllocateShortAddr()	Addr	AllocateShortAddrResult	Allocating short address for a field device The values of AllocateShortAddrResult include: SUCCESS ; FAILURE

HandleFieldJoinRequest()		JoinResult	Handle the join request from device.
			The value of JoinResult includes:
			SUCCESS
			NetworkID mismatched;
			Authentication failure;
			Network overload;

HandleTwoWaySyn();			Handle two way synchronization.
IsHostComputerConfigureDone()		HCCDResult	Judging whether the application configuration of the host computer is finished The values of HCCDResultinclude: TRUE ; FALSE
ResAllocAgrithm()	SuperframeList LinkList	ResAllocResult	Allocating communication resources The values of ResAllocResult include: SUCCESS ; NO_RESOURCE
IsHostComputerRequestDeviceLeave()		DeviceLeaveResult	Judging whether the host computer requests a field device leave WIA-FA network The values of DeviceLeaveResult include: TRUE ; FALSE
IsHostComputerSet MIB()		HostComputerSetMIBResult	Judging whether the host computer remotely request setting the attributes of a field device The values of HostComputerSetMIBResult include: TRUE ; FALSE

Function	Input	Output	Description
IndicateSetMIBResult()	Handle Status		Indicating the results of the remote attribute set operation to the host computer
HandleChannelStatusReport()	Addr ChannelConditonInfo		Processing the channel condition report
HandleDeviceStatusReport()	PowerSupplyStatus		Process the device status report
IsHostComputerGet MIB()		HostComputerGetMIBResult	Judging whether the host computer remotely request getting the MIB attributes The values of HostComputerGetMIBResult include: TRUE ; FALSE
IndicateGetMIBResult()	Handle Status		Indicating the results of the remote attribute get operation to the host computer
ReleaseResources()	Addr		Releasing communication resources occupied by a field device
IndicateHostComputerLeaveResult()	Addr		Indicating the leave of a field device to the host computer
IsDMAPInitializationDone()		DMAPInitializationResult	Judging whether the DMAP initialization is finished The values of DMAPInitializationResult include: TRUE ; FALSE
SendWiredFrameToGW();	FrameName		Send wired frames between AD and GW, detail format see 9.3 for frame formats between GW and AD.

6.3 Addressing and address assignment

Each WIA-FA network device shall have a global unique 64-bit long address and an 8-bit or 16-bit short address (indicated by AddressTypeFlag in Table 16). When the number of field devices in the network is less than 252, 8-bit short address should be adopted; otherwise, the 16-bit short address should be adopted. The long address, shown in Figure 16, shall be assigned by manufacturers according to the EUI-64. The least significant octet of DeviceShortAddress (see Table 21 in 6.7.1.2.2) is valid if the network uses 8-bit short address.

Gateway device shall use AdID (see Table 21 in 6.7.1.2.2) to distinguish different access devices.

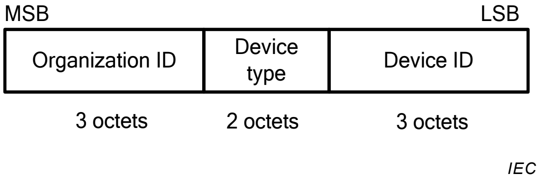


Figure 16 – Long address structure of device

In the WIA-FA network, the 8-bit short address of a WIA-FA device shall be set as follows:

- The short address of the gateway device is 0x01;
- The short address of each access device is 0x02;
- The short address range of a field device is 0x03 to 0xFE;
- The broadcast address in the WIA-FA network is 0xFF.

In the WIA-FA network, the 16-bit short address of a WIA-FA device shall be set as follows:

- The short address of the gateway device is 0x0001;
- The short address of each access device is 0x0002;
- The short address range of a field device is 0x0003 to 0xFFFE;
- The broadcast address in the WIA-FA network is 0xFFFF.

The default short address of a WIA-FA device shall be 0x00 or 0x0000, which means the device hasn’t been assigned a short address.

6.4 Communication resource allocation

6.4.1 General

Communication resources consist of channels (see 3.1.8) and timeslots (see 3.1.30). Allocation of communication resources means allocating channels and timeslots in superframe to access devices and field devices for creating links according to data priorities and resource occupation methods. LinkList attributes shall include LinkID, LinkType, PeerAddr, RelativeSlotNumber, ChannelIndex, and SuperframeID (see Table 19 in 6.7.1.2.2).

6.4.2 Communication resource allocation

6.4.2.1 Data priorities

According to functions and requirements of data in industrial fields, the data shall be set to different priorities. WIA-FA defines the following five data priorities.

- Urgent data (RT0)

RT0 data shall have the highest priority. It refers to the data that plays a key role to application behaviours and requires being timely transferred. RT0 data includes: command from the host computer to brake the actuator; urgent alarm of failure/error notification; critical network management services from the host computer, such as start/stop command. The transmission of RT0 data shall use R/S communication model (see 10.5.5.4.4).

- Periodic process data (RT1)

RT1 data shall have the second highest priority. It refers to the periodically transmitted process data that has strict real-time requirement. RT1 data mainly includes the physical measurement and control data of a control system. The transmission of RT1 data shall use P/S communication model (see 10.5.5.4.3).

- Aperiodic non-urgent data (RT2)

RT2 data shall have the third highest priority. It refers to the aperiodically transmitted data triggered by events, such as the non-urgent alarm data. The transmission of RT2 data shall use R/S communication model (see 10.5.5.4.4).

– Periodic management data (RT3)

RT3 data shall have the fourth highest priority. It refers to the periodically transmitted monitoring data of device and network status, which has certain real-time requirements. RT3 data mainly includes the device status, channel condition, etc. The transmission of RT3 data shall use P/S communication model (see 10.5.5.4.3).

– Non-real-time data (NRT)

NRT data shall have the lowest priority. It refers to data that is generated by network operation and has no real-time requirement. In the industrial field, NRT data usually includes configuration parameters and management data. The NRT data transmission should not interfere with transmissions of real-time data. The transmission of NRT data shall use C/S communication model (see 10.5.5.4.2).

The transmission of data with one of the above five priorities shall use different communication models and different VCRs. The definitions of communication models, VCRs, and the corresponding relationships between VCRs and data are shown in 10.5.5.4.

6.4.2.2 Occupation method of communication resources

For transmitting data with different priorities, the occupation methods of communication resources shall include scheduling, preemption, and competition.

– Scheduling

Scheduling is used for transmitting RT1 data and RT3 data. Network manager shall allocate all communication resources for the whole WIA-FA network. After field devices and access devices joining the network, the network manager shall allocate fixed timeslots to periodically transmit and retransmit RT1 and RT3 data.

– Preemption

Preemption is used to transmit RT0 data. The gateway device and field devices can utilize timeslots original for periodic data (RT1 data and RT3 data) to transmit RT0 data, which defers the transmission of the periodic data.

– Competition

Competition refers to using transmit-shared link (see Table 19) to transmit RT0 data, RT2 data, and NRT data according to data priorities.

6.5 Joining and leave process of field device

6.5.1 Join process of a field device

The handheld device provisions the gateway device through wired maintenance port. The provisioning information includes:

- NetworkID;
- SecLevel;
- KJ (when the security level is not 0);
- KS (when the security level is not 0 or 1);
- the long address list of the joining devices.

The handheld device provisions a field device through wired maintenance port when this field device attempts to join WIA-FA network. The provisioning information includes:

- NetworkID;
- SecLevel;
- KJ (when the security level is not 0);
- KS (when the security level is not 0 or 1).

The join process of a provisioned field device is shown in Figure 17. The join process shall be as follows.

- Access devices periodically broadcast beacon frames;
- A field device with attempt to join the WIA-FA network continually scans available channels to get beacon frames from access devices and synchronizes with the gateway device by using one-way time synchronization method (see 8.1.5);
- The field device chooses a channel on which field device receives Beacon frame and utilizes the shared timeslots for sending the Join request frame; the shared timeslot is determined by the "First shared timeslot number" and the “Shared timeslot count” (see 8.4.6) in the Beacon frame); the field device competes for transmitting Join request frame utilizing timeslot based backoff method (see 8.1.8.5) on the channel used by the Beacon frame;
- An access device transfers the Join request frame received from a field device to the gateway device;
- The NM residing on the gateway device returns the Join response frame; if the joining is approved, the value of Status in the Join response frame is set to SUCCESS; otherwise, the value of the Status should be set according to error reasons (See Status in 8.3.4.3);
- An access device forwards the Join response frame to the corresponding field device;
- The field device receives the Join response frame from an access device; if the value of Status in the Join response frame is not SUCCESS, the field device should restart or terminate the join process; if the value of Status in the Join response frame is SUCCESS, the join process is completed.

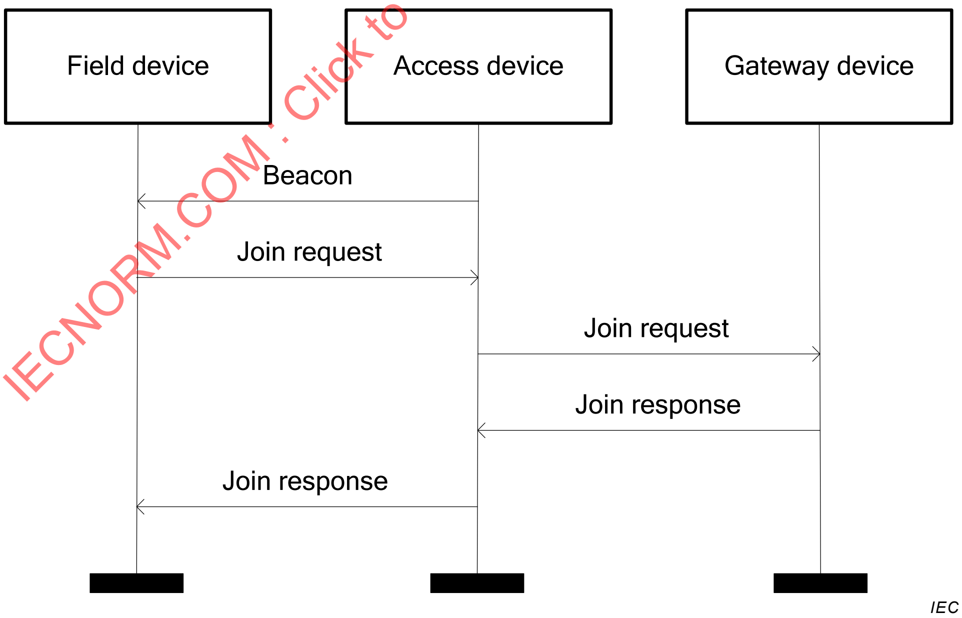


Figure 17 – Join process of field device

NOTE See Clause 11 for security.

6.5.2 Communication resource allocation to field device

After a new field device joins WIA-FA network, the host computer requests the gateway device to read UAOs of the field device by using the remote attribute get services (see 8.3.7), and to write application configuration information/VCRs (see 10.5.5.3 and 10.5.5.4) for the field device by using the remote attribute set services (see 8.3.8).

The NM residing on the gateway shall allocate communication resources for new joined field device by utilizing the remote attribute set services (see 8.3.8). These communication resources are used for communication between field devices and access devices. If the new joined field device influences the superframe structures of access devices (see 8.1.3), the NM shall update SuperframeList and LinkList attributes of both access devices and field devices.

The communication resource allocation process for a field device is illustrated in Figure 18.

- The network manager sends the remote attribute set request;
- Access devices forward the remote attribute set request to the field device;
- After receiving the remote attribute set request, the field device returns the remote attribute set response;
- Access devices forward the remote attribute set response to the gateway device.

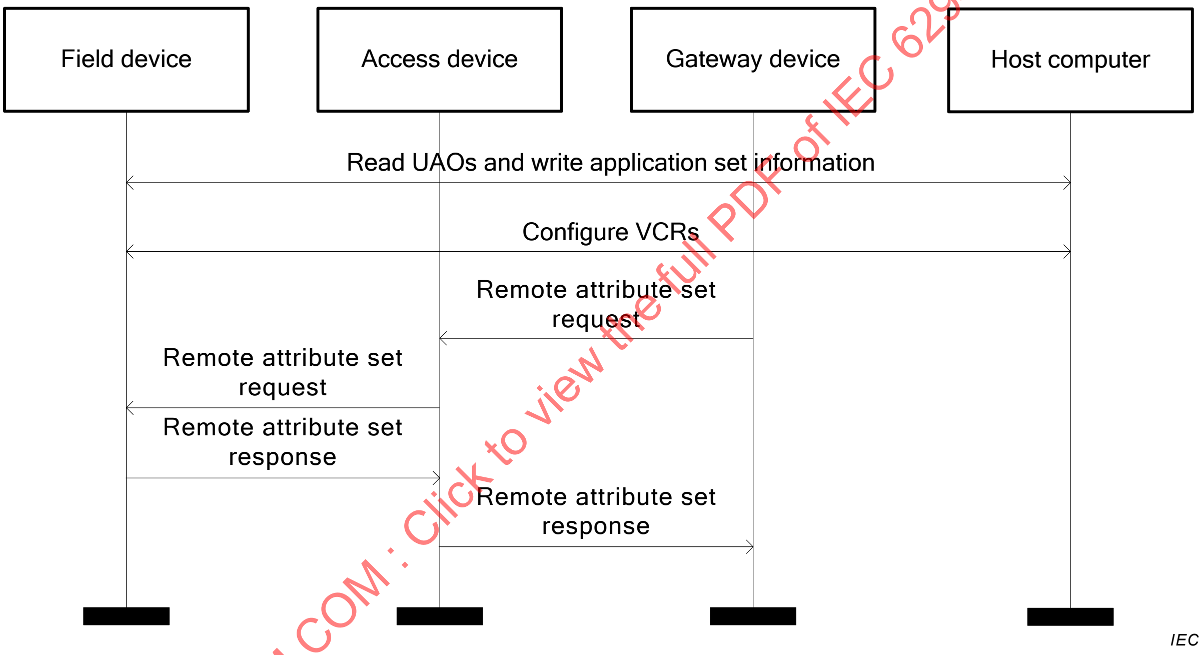

Field device
Access device
Gateway device
Host computer
Read UAOs and write application set information
Configure VCRs
Remote attribute set request
Remote attribute set request
Remote attribute set response
Remote attribute set response
IEC

Figure 18 – Communication resource allocation process for a field device

6.5.3 Leaving process of a field device

The leave process of field devices includes abnormal leaving and passive leaving.

- Abnormal leaving: devices can not communicate with other devices because of failure, invalidation, or energy depletion. If the gateway device does not receive any packet from a field device during LossConnectDuration (see 6.7.1.2.1), it shall judge that the field device has left the WIA-FA network abnormally. The gateway device releases the short address and communication resources of the field device. If the field device does not receive any packet from access devices during LossConnectDuration (see 6.7.1.2.1), it shall judge that its connection with access devices has lost. The field device shall handle it as abnormal leaving, then releases all resources and sets itself as unjoined state.
- Passive leaving: the gateway device requests a field device to leave the WIA-FA network. The passive leave process of a field device is shown in Figure 19, which includes the following procedures.
  - The gateway device sends the Leave request frame to a field device (see 8.4.9) through the access device;

- After receiving the Leave request frame, the field device returns a Leave response frame, then releases all resources and sets itself as unjoined state;
- After the gateway device receiving the Leave response frame forwarded by access devices from a field device, the network manager on the gateway device handles the Leave response frame, updates the DeviceList attribute, and releases the short address and communication resources of the leaving field device; and the access device updates its MIB accordingly.

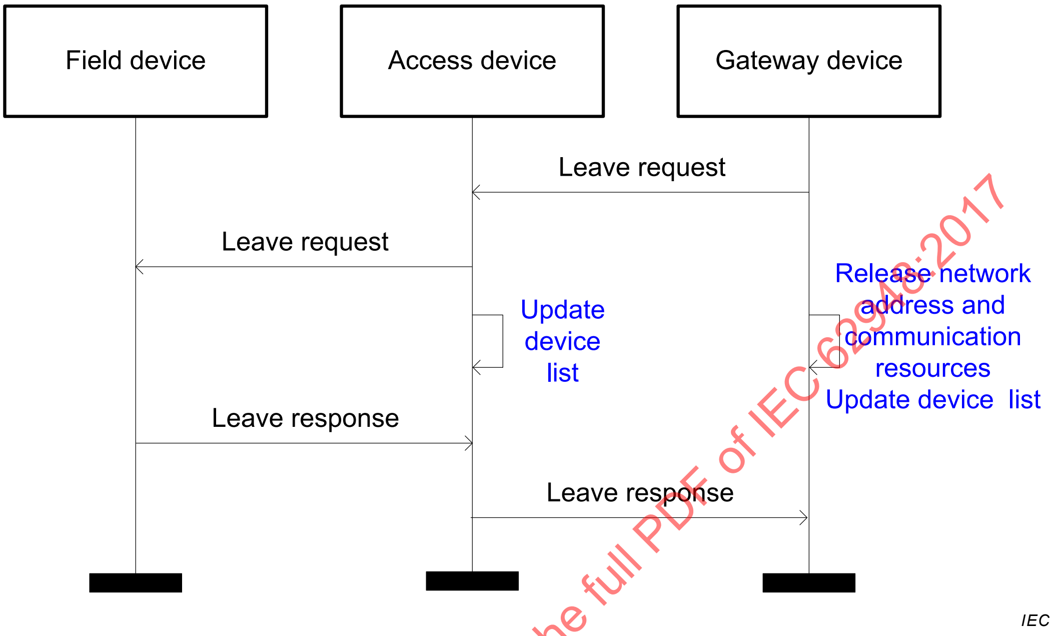


Figure 19 – Passive leave process of a field device

6.6 Network performance monitoring

6.6.1 Device status report

The field device periodically reports its device status to the gateway device (see DeviceList in 6.7.1.2.2 for the detailed device status). After the gateway device receives the device status report of a field device, the NM evaluates and diagnoses the device status. The device status report is used to detect abnormal status of a field device, such as low battery power. The status report process of a field device is shown in Figure 20.

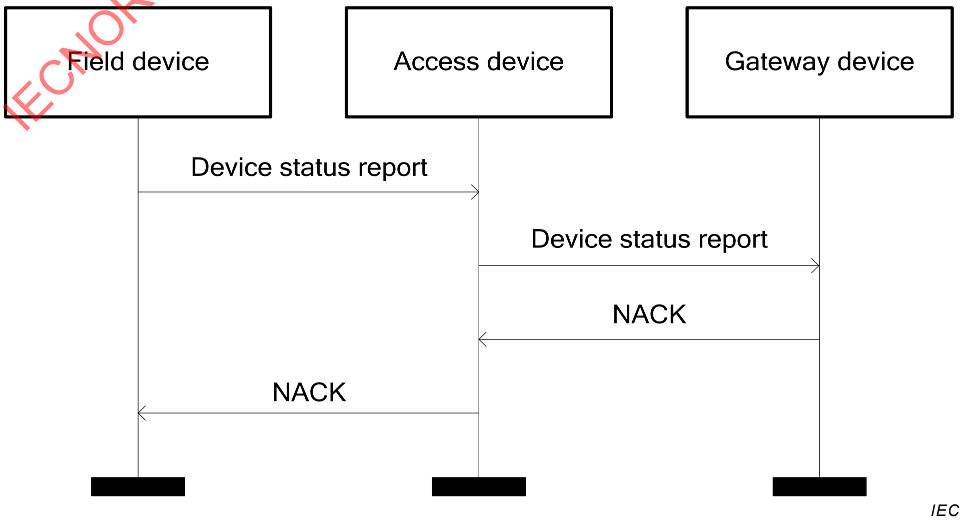


Figure 20 – Device status report process of field device

6.6.2 Channel condition report

The channel condition report is used for a field device to remotely report the channel condition to the gateway device (see ChannelConditionList in 6.7.1.2.2 for the detailed channel condition). The process of the channel condition report is shown in Figure 21.

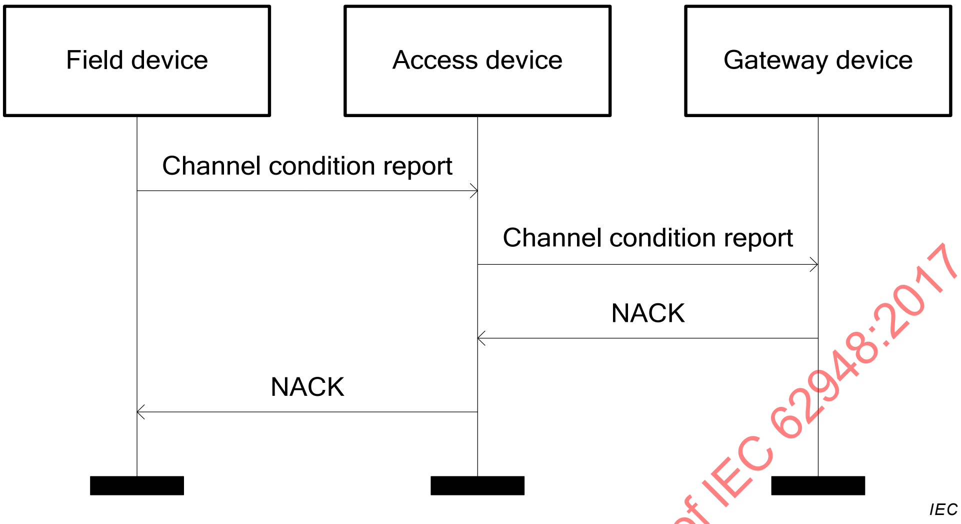


Figure 21 – Channel condition report process of field device

6.7 Management information base and services

6.7.1 Management information base

6.7.1.1 General

The attributes of WIA-FA device are stored in the MIB. These attributes can be configured, accessed and updated by the NM.

According to the storage types, the attributes in the MIB are classified into three categories.

- C (Constant): its value is unchangeable with time, such as the global unique 64-bit address. The constant attribute is set by manufacturer and shall not be modified.
- S (Static): its value can be changed infrequently during the process of device operation, and can only be modified by NM. The static attribute shall hold the previous setting value after the powerup or restart.
- D (Dynamic): its value can be changed frequently without any external request. The dynamic attribute shall recover to the default value set by manufacturer after the powerup or restart.

According to the attribute data types, the attributes in the MIB are divided into unstructured attributes and structured attributes.

There are two access types for attributes in the MIB:

- R (Read): the value of the attribute can be read by other devices in WIA-FA network;
- W (Write): the value of attribute can be modified by other devices in WIA-FA network.

According to the implementation requirements, the attributes in the MIB are divided into mandatory attributes and optional attributes. Device type indicates whether the attribute is valid for the specified devices.

Attributes in MIB are identified by AttributeID (see Table 16).

6.7.1.2 MIB attributes

6.7.1.2.1 Unstructured attributes

Unstructured attributes are listed in Table 16. The values of unstructured attributes shall be unique in the whole WIA-FA network.

Table 16 – Unstructured attributes

Attribute ID	Attribute name	Data type	Valid range	Ac-cess type	Sto-rage type	De-fault value	Device type	Description
0	PriGwFailureTime	TimeData	0 to $(2^{64} – 1)$	R/W	S	10	Gate-way device	The maximal duration without heartbeat signal (in µs)
1	AddressTypeFlag	Unsigned8	0 to 255	R/W	S	0	All devices	0= 8-bit short address; 1= 16-bit short address
2	MaxPayloadLength	Unsigned16	0 to $(2^{16} – 1)$	R/W	S	1 000	All devices	Maximum length of the DLL payload (in octet)
3	ADRetransNum	Unsigned8	0 to 255	R/W	S	1	Gate-way devices and access devices	Retransmition number of each AD
4	NetworkID	Unsigned8	0 to 255	R/W	S	0	All devices	Network identifier, used for identifying multiple coexisting networks

5	BitMap	BitField24		R/W	S	0x3fff3	All devices	Channel bitmap, indicating used modulation mechanism and channel states:
								bit 0 to 3: modulation mode;
								0 = FHSS;
								1 = DSSS;
								2 = HR/DSSS;
								3 = OFDM;
								4 = ERP-DSSS;
								5 = ERP-CCK;
								6 = ERP-OFDM;
								7 = ERP-PBCC;
								8 = DSSS-OFDM;
								9 = No-HT mode;
								10 = Mix mode;
								11 = HT mode;
								others are reserved.
								Each bit of bits 4 to 17 represents channel state of channels 1 to 14 respectively.
								0 = unavailable channel;
								1 = available channel;
								bit 18 to bit 23 is reserved.
								See 7.3.3 for details of channel BitMap.

Attribute ID	Attribute name	Data type	Valid range	Access type	Storage type	Default value	Device type	Description
6	DevStaRptCycle	Unsigned16	0 to 65 535	R/W	S	10	Gateway devices and field devices	Cycle of device status report (in length of default superframe)
7	ChaStaRptCycle	Unsigned16	0 to 65 535	R/W	S	10	Gateway devices and field devices	Cycle of channel status report (in length of default superframe)
8	LossConnectDuration	Unsigned24	0 to $(2^{24} – 1)$	R/W	S	30	Gateway device and field devices	If a WIA-FA device does not receive any packet from a neighbour device during LossConnectDuration, it shall judge that the neighbour device has abnormally left the WIA-FA network (in length of default superframe)
9	KeyupdataDuration	Unsigned8	0 to 255	R/W	S	24	All devices	Cycle of key update (in hour)
10	TimeSlotDuration	Unsigned16	0 to 65 535	R/W	S	256	All devices	Timeslot length (in μs)
11	TwoWayTimeSyn	Unsigned8	0 to 255	R/W	S	0	All devices	Indicate the utilization of two-way time synchronization method: 0= one-way time synchronization; 1= two-way time synchronization.
12	TwoWayOverTime	Unsigned8	0 to 255	R/W	S	1	All devices	The dead time of the two-way time synchronization (in length of default superframe)
13	ADTeamNum	Unsigned8	0 to 255	R/W	S	1	Gateway device	Count of the AD teams
14	TargetLossRate	Single Float	0 to 255	R/W	S	0	Gateway device	Expected packet loss rate in WIA-FA network, which is between 0 and 1
15	LossRate	Single Float	0 to 255	R/W	S	0	Gateway device	Current packet loss rate of factory environment, which is between 0 and 1.
16	MaxRetry	Unsigned8	0 to 255	R/W	S	3	All devices	Maximum number of retransmissions

Attribute ID	Attribute name	Data type	Valid range	Access type	Storage type	Default value	Device type	Description

17	SecLevel	Unsigned8	0 to 255	R/W	S	1	All devices	Security level of DLL:
								0=None;
								1=Authentication;
								2=Authentication&MIC-32;
								3=Authentication&MIC-64;
								4=Authentication&MIC-128;
								5=Authentication&ENC;
								6=Authentication&ENC&MIC-32;
								7=Authentication&ENC&MIC-64;
								8=Authentication&ENC&MIC-128;
								others are reserved

18	AttackStatisDur	Unsigned16	0 to 65 535	R/W	S	60	All devices	The period of attack statistics (in min)
19	MaxKeyAttackedNum	Unsigned8	0 to 255	R/W	S	5	All devices	Maximum count of key attacked
20	AlarmRptDur	Unsigned8	0 to 255	R/W	S	1	All devices	Interval of repeating alarm (in length of default superframe)
21	ChannelNum	Unsigned8	0 to 255	R/W	S	1	All devices	Available channel count for a link
22	AGGEnableFlag	Unsigned8	0, 1	R/W	S	0	All devices	Aggregation and disaggregation enable flag: 0 = Disable; 1 = Enable
23	UTC	Unsigned32	0 to (2^{32} – 1)	R/W	D	0	All devices	Coordinated universal time

6.7.1.2.2 Structured attributes

Structured attributes are listed in Table 17. Access device and field device shall only maintain structured attributes related to them. Besides its own structured attributes, gateway device should maintain the structured attributes for each field device and access device.

Table 17 – Structured attributes

Attribute ID	Attribute name	Data type	Access type	Storage type	Device type	Description
128	SuperframeList	Superframe_Struct List, see Table 18	R/W	S	All devices	Describing the superframe information
129	LinkList	Link_Struct List, see Table 19	R/W	S	All devices	Describing the link information
130	ChannelConditionList	ChanCon_Struct List, see Table 20	R/W	D	Gateway device and field devices	Recording the statistic information of channel condition
131	DeviceList	Device_Struct List, see Table 21	R/W		All devices	Describing the device-related attributes of WIA-FA devices
132	KeyList	Key_Struct List, see Table 22	R/W		All devices	Describing the information of keys
133	VCRList	VcrEP_Struct List, see Table 23	R/W	S	Gateway device and field devices	Recording the VCR information
134	SupUAOList	UAOClassDesc _Struct List, see Table 24	R	S	Gateway device and field devices	Describing the information of UAO classes that the field device supports
135	CfgUAOList	UAPInstDesc _Struct List, Table 25	R/W	S	Gateway device and field devices	Describing the information of UAO instants that are configured by the gateway device

Table 18 – Superframe_StructStructure

MemberID	Member name	Data type	Valid range	Description
0	SuperframeID	Unsigned8	0 to 255	Unique identifier of a superframe, assigned by the NM
1	NumberSlots	Unsigned16	0 to 65 535	Superframe size (counts of timeslots)
2	ActiveFlag	Unsigned8	0 to 255	Superframe active flag: 0 = Inactive; 1 = Active.
3	ActiveSlot	Unsigned48	0 to (2^{48} – 1)	Absolute timeslot number (ASN) indicates when a superframe being active, which is calculated as: ⌊TimeValue/TimeSlotDuration⌋. "⌊TimeValue/TimeSlotDuration⌋" stands for the greatest integer less than or equal to the result of "TimeValue/TimeSlotDuration"

Table 19 – Link_Struct Structure

MemberID	Member name	Data type	Valid range	Description
0	LinkID	Unsigned16	0 to 65 535	Unique identifier of a link

1	LinkType	Unsigned8	0 to 255	Bit 0 represents the link type:
				0 = Unicast;
				1 = Broadcast;
				Bit 1 and bit 2 represent the characteristics of a link:
				00 = Transmitting;
				01 = Transmit-shared;
				10 = Retransmitting;
				11 = Receiving;
				Bit 3 to bit 5represents the type of a timeslot:
				000=Beacon;
				001=NACK;
				010= GACK;
				011= Management timeslot;
				100= Data timeslot;
				101 = Management/Data timeslot;
				others are reserved.
				Bit 6 to bit 7 is reserved.

2	ActiveSlot	Unsigned48	0 to $(2^{48} - 1)$	Absolute timeslot number (ASN) indicates when a link being active, which is calculated as: ⌊TimeValue/TimeSlotDuration⌋. "⌊TimeValue/TimeSlotDuration⌋" stands for the greatest integer less than or equal to the result of "TimeValue/TimeSlotDuration".
3	PeerAddr	Unsigned16	0 to 65 535	The short address of peer device
4	RelativeSlotNumber	Unsigned16	0 to 65 535	Relative timeslotnumber
5	ChannelIndex	Unsigned8List		The ChannelID list of used channels to support channel hopping. This document uses 0 to 13 to identify ChannelID; others are reserved. The channel count in ChannelIndex is restricted by ChannelNum (see 6.7.1.2.1). ChannelIndex is allocated by NM. Each WIA-FA field device sequently changes a channel according to ChannelIndex in each timeslot.
6	SuperframeID	Unsigned8	0 to 255	Reference to a superframe in the SuperframeList.

Table 20 – ChanCon_Struct Structure

MemberID	Member name	Data type	Valid range	Description
0	ChannelID	Unsigned8	0 to 255	Unique identifier of a channel. This document uses 0 to 13 to identify ChannelID; others are reserved.
1	LinkQuality	Unsigned8	0 to 255	Link Quality Indication (LQI) value of a channel
2	PacketLossRate	Single Float	0 to 255	Packet loss rate of a channel, the value of which is between 0 and 1
3	RetryNum	Unsigned16	0 to 65 535	The count of retransmission of every channel

Table 21 – Device_Struct Structure

Member ID	Member name	Data type	Valid range	Access type	Storage type	Default value	Description
0	Version	Unsigned16	0 to 65535	R	C		Standard version

1	LongAddress	Unsigned64	0 to ($2^{64}-1$)	R	C		EUI-64 (see 6.3), which defines the octet 3 and octet 4 as follows:
							0 = Gateway device;
							1 = Access device;
							2 = Field device;
							3 = Handheld device;
							others are reserved.

2	AGGSupportFlag	Unsigned8	0 to 255	R	S	1	Indicating whether a WIA-FA device supports aggregation/ disaggregation mechanism:
							0 = Not support;
							1 = Support;
							others are reserved.

3	NumOfSupUAO	Unsigned16	0 to 65 535	R	S	1	Number of UAO classes supported by a field device
4	NumOfCfgUAO	Unsigned16	0 to 65 535	R/W	S	1	Number of configured UAO instances in a field device
6	ProbeTime	Unsigned8	0 to 255	R/W	S	2	Time for scanning a channel (in length of default superframe)
7	TimeValue	TimeData	0 to (2^{64}–1)	R/W	D	0	Absolute time (in μs), counted from zero time

8	RedundantDevFlag	Unsigned8	0 to 255	R/W	S	0	Indicating whether this device is a redundant device:
							0 = Primary device;
							1 = Redundant device;
							others are reserved.

9	AdID	Unsigned8	0 to 255	R/W	S	0	Identifier of an access device, invalid for a field device
10	DeviceShortAddress	Unsigned16	0 to 65 535	R/W	S	0	Short address of a device (see 6.3)
11	PowerSupplyStatus	Unsigned8	0 to 255	R/W	S	10	Types of power supply and energy levels: 0 = Fixed power supply; 1 to 10 = energy level of battery power supply (from low to high); others are reserved.

Member ID	Member name	Data type	Valid range	Access type	Storage type	Default value	Description

12	DeviceState	Unsigned8	0 to 3	R/W	D	0	Status of device:
							0 = Not joined;
							1 = Joining;
							2 = Joined;
							3 = Application configuring;
							4 = Resource allocating;
							5 = Operation;
							others are reserved.

Table 22 – Key_Struct Structure

MemberID	Member name	Data type	Valid range	Access type	Storage type	Description
0	KeyID	Unsigned16	0 to 65 535	R/W	S	Unique identifier of a key
1	PeerAddr	Unsigned16	0 to 65 535	R/W	S	The short address of peer device

2	KeyType	Unsigned8	0 to 255	R/W	S	The type of a key:
						0 = KJ;
						1 = KS;
						2 = KEK;
						3 = KEDU;
						4 = KEDB;
						others are reserved.

3	KeyDataValue	KeyData		R/W	S	The value of a key
4	KeyActiveSlot	Unsigned48	0 to $(2^{48}-1)$	R/W	S	Absolute timeslot number (ASN), indicating when the key being active (in ms)
5	KeyAttackCnt	Unsigned16	0 to 65 535	R	D	The total attack number of a key
6	AlarmFlag	BitField8		R	D	Each bit of AlarmFlag indicates a security alarm event. If a security alarm event related to the key is detected, the corresponding bit is set to 1. Bit 0: key attacked alarm; Bit 1: key update timeout alarm; others are reserved.

7	KeyState	Unsigned8	0 to 255	R/W	D	The state of a key:
						0 = BACKUP;
						1 = USING;
						2 = EXPIRED;
						3 = INVALID;
						others are reserved.

Table 23 – VcrEP_StructStructure

MemberID	Member name	Data type	Valid range	Description
0	VCR_ID	Unsigned16	0 to 65 535	Unique identifier of the VCR on the device. VCR_ID = 0 is used for default C/S VCR, and others are assigned by NM.

1	VcrEP_Type	Unsigned8	0 to 255	Types of the VCR endpoint:
				0=CLIENT;
				1=SERVER;
				2=PUBLISHER;
				3=SUBSCRIBER;
				4=REPORT SOURCE;
				5=REPORT SINK;
				others are reserved.

2	UAP_ID	Unsigned8	0 to 255	Unique identifier of the UAP on the device. UAP_ID = 0 is used for DMAP, and others are assigned by NM.
3	PeerAddr	Unsigned16	0 to 65 535	Short address of the peer field device or the gateway device
4	VCRActiveTime	TimeData	0 to $(2^{64}-1)$	Only valid for P/S VCR, indicating when the VCR endpoint should be activated. The DataUpdateRate of the UAP shall start at this time. The default value is 0, which means the VCR endpoint shall be activated immediately. For a C/S VCR or a R/S VCR, the value should be set 0.
5	DataUpdateRate	Unsigned32	0 to $(2^{32}-1)$	Only valid for P/S VCR, indicating the publishing cycle of UAP (in ms). For a C/S VCR or a R/S VCR, the value should be set 0.
6	Deadline	Unsigned8	0 to 255	Only valid for P/S VCR, indicating the multiples of DataUpdateRate for the VCR endpoint. The permitted maximum time interval not receiving new data is DataUpdateRate × Deadline. If a device has not received new data within DataUpdateRate × Deadlinetime interval, it shall generate the "PROCESS DATA NOT UPDATED" alarm event. The default value is 1.
7	WatchdogTime	Unsigned32	0 to $(2^{32}-1)$	Only valid for C/S VCR, indicating the maximum time interval that the VCR endpoint shall wait for the service response (in ms). The default value is 10 s. If the VCR endpoint has not received the service response within WatchdogTime interval, it shall return a negative response with " SERVICE_EXPIRATION"

Table 24 – UAOClassDesc_Struct Structure

MemberID	Member name	Data type	Valid range	Description
0	Class_ID	Unsigned8	0 to 255	Unique identifier of the UAO class on the field device, indicating the class template for instantiating UAOs.

1	UAOType	Unsigned8	0 to 255	The type of a UAO class:
				0 = AI;
				1 = AO;
				2 = DI;
				3 = DO;
				others are reserved.

2	MaxInputDataLen	Unsigned16	0 to 65 535	Maximum input data size supported by the UAO class
3	MaxOutputDataLen	Unsigned16	0 to 65 535	Maximum output data size supported by the UAO class
4	MinDataUpdateRate	Unsigned32	0 to $(2^{32}-1)$	Minimum publishing cycle (in ms) supported by the UAO class
5	SuppInputType	ProDataDesc_Struct Structure, see Table 25		Input data description of the UAO class, indicating all data types of the input data supported by the UAO class. That is, the bits corresponding to the input data types supported by the UAO class shall be set to 1. If the UAO class has no input data, each bit shall be set to 0.
6	SuppOutputType	ProDataDesc_Struct Structure, see Table 25		Output data description of the UAO, indicating all data types of the output data supported by the UAO class. That is, the bits corresponding to the output data types supported by the UAO class shall be set to 1. If the UAO class has no output data, each bit shall be set to 0.

Table 25 – ProDataDesc_Struct Structure

MemberID	Member name	Data type	Data Length (in octet)	Valid range	Description
0	ParamDesc	BitField16	2		For SupUAOList, it shall describe the process data types supported by the UAO class. For CfgUAOList, it shall describe one of the input or output data types configured for the UAO instance and the count of data with corresponding data type. Bit 10 to Bit 15, each bit stands for one of the following data types: Bit 15 indicating Unsigned8 data type; Bit 14 indicating Unsigned16 data type; Bit 13 indicating Unsigned32 data type; Bit 12 indicating Single Float data type; Bit 11 indicating Double Float data type; Bit 10 indicating Bitstring data type. The value of each bit from Bit 10 to Bit 15 is: 0 = not supported; 1 = supported When this structure is used to describe SuppInputType or SuppOutputType of the UAO classes in SupUAOList, more than one bits of Bit 10 to Bit 15 may be set to 1, indicating all data types supported by the UAO class. In this case, Bit 0 to Bit 9 is invalid and shall be reserved as 0. When this structure is used to describe CfgInputDataList or CfgOutputDataList of the UAO instances in CfgUAOList, only one bit of Bit 10 to Bit 15 shall be set to 1, indicating one input or output data type configured for the UAO instance. In this case, the value from Bit 0 to Bit 9 shall indicate the count of data with the corresponding data type, and the values are: Bit 0 to Bit 9 = 0, one data with the corresponding data type; Bit 0 to Bit 9 = 1, two data with the corresponding data type; … Bit 0 to Bit 9 = 1 023, 1 024 data with the corresponding data type.

Table 26 – UAOInstDesc_Struct Structure

MemberID	Member name	Data type	Valid range	Description
0	UAO_ID	Unsigned8	0 to 255	Unique identifier of the UAO. UAO_ID = 0 is used for MIB.
1	Class_ID	Unsigned8	0 to 255	Class identifier of the UAO class, indicating the UAO is an instance of the UAO class with class_ID in the SuppUAOList
2	UAP_ID	Unsigned8	0 to 255	Identifier of the UAP that the UAO belongs to on the device. If a UAP is allocated with more than one UAO, these UAOs shall have the same UAP_ID
3	AckFlag	BitField16		This value shall be set to the AckFlag of the UAO EventData. See Table 77 for the coding of each bit.
4	NumInputData	Unsigned8	0 to 255	Count of UAO input data
5	NumOutputData	Unsigned8	0 to 255	Count of UAO output data
6	CfgInputDataList	ProDataDesc_StructList, see Table 25		Data description list of the UAO input data. Each record of the list indicates the data type of one input data and its count. The record order specifies the input data order to be transferred periodically of the UAO instance.
7	CfgOutputDataList	ProDataDesc_StructList, see Table 25		Data description list of the UAO output data. Each record of the list indicates the data type of one output data and its count. The record order specifies the output data order to be transferred periodically of the UAO instance.
8	DataUpdateRate	Unsigned32	0 to $(2^{32}-1)$	Specifying the period for repeatedly transmitting the input and output data of the UAO (in ms)
9	PrioFlag	BitField16		This value shall be set to the PrioFlag of the UAO's EventData. See Table 77 for the coding of each bit.

6.7.2 MIB services

6.7.2.1 General

The attributes in the MIB can be read and written locally through the DMAP attribute get and DMAP attribute set services provided by local DMAP.

6.7.2.2 DMAP attribute get service

DMAP-MIB-GET.request is used by all layers to request attributes in the MIB.

The semantics of DMAP-MIB-GET.request are as follows:

```
DMAP-MIB-GET.request(
                        Handle,
                        AttributeID,
                        MemberID,
                        FirstStoreIndex,
                        Count
                         )
```

Table 27 specifies the parameters for DMAP-MIB-GET.request.

Table 27 – DMAP-MIB-GET.request parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DMAP-MIB-GET.request
AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of an attribute member. The value 255 means that all attribute members shall be read. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The index of the first one of records to be read. FirstStoreIndex shall be valid only for attributes of List type.
Count	Unsigned16	0 to 65 535	Number of records to be read. Count shall be valid only for attributes of List type. All records from FirstStoreIndex shall be read if Count = 0

DMAP-MIB-GET.confirm is used to return the result of DMAP-MIB-GET.request.

The semantics of DMAP-MIB-GET.confirm are as follows:

```
DMAP-MIB-GET.confirm(
                    Handle,
                    Status,
                    Count,
                    AttributeValue
                    )
```

Table 28 specifies the parameters for DMAP-MIB-GET.confirm.

Table 28 – DMAP-MIB-GET.confirm parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DMAP-MIB-GET.request

Status	Unsigned8	0 to 255	Attribute getting results:
			0=SUCCESS;
			1=INVALIDATTRIBUTE;
			2=INVALIDATTRIBUTEMEMBER;
			3=INVALIDRANGE;
			others are reserved.

Count	Unsigned16	0 to 65 535	Number of records which has been read. Count shall be valid only for attributes of List type and Status = SUCCESS.
AttributeValue	Octetstring		Returned attribute values or attribute member values. AttributeValue is valid only if Status = SUCCESS.

If the operation of getting attributes is successful, the Status shall be SUCCESS and the AttributeValue is valid; if the MIB does not have the requested attributes, the Status shall be INVALID ATTRIBUTE; if the MIB does not have the requested attribute members, the Status shall be INVALID ATTRIBUTE MEMBER; if the MIB does not have the requested records indexed with FirstStoreIndex, the Status shall be INVALID RANGE.

6.7.2.3 DMAP attribute set service

DMAP-MIB-SET.request is used by all layers to write attributes to the MIB.

The semantics of DMAP-MIB-SET.request are as follows:

```
DMAP-MIB-SET.request(
                    Handle,
                    ShortAddr,
                    AttributeID,
                    MemberID,
                    FirstStoreIndex,
                    Count,
                    AttributeValue
                    )
```

Table 29 specifies the parameters for DMAP-MIB-SET.request.

Table 29 – DMAP-MIB-SET.request parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DMAP-MIB-SET.request
ShortAddr	Unsigned16	0 to 65 535	The 8- or 16-bit short address of a field device, or the AdID (see 6.7.1.2.2) of an access device. ShortAddr shall be valid only for GW to write its local MIB attributes of a field device or an access device.
AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of an attribute member. The value 255 means that all attribute members shall be written. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The index of the first one of records to be written. FirstStoreIndex shall be valid only for attributes of List type.
Count	Unsigned8	0 to 255	Number of records to be written. Count shall be valid only for attributes of List type. All records from FirstStoreIndex shall be written if Count = 0
AttributeValue	Octetstring		Attribute values or attribute member values to be written.

DMAP-MIB-SET.confirm is used to return the result of DMAP-MIB-SET.request.

The semantics of DMAP-MIB-SET.confirm are as follows:

```
DMAP-MIB-SET.confirm(
                    Handle,
                    Status
                     )
```

Table 30 specifies the parameters for DMAP-MIB-SET.confirm.

Table 30 – DMAP-MIB-SET.confirm parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DMAP-MIB-SET.request

Status	Unsigned8	0 to 255	Attribute setting results:
			0=SUCCESS;
			1=INVALIDATTRIBUTE;
			2=INVALIDATTRIBUTEMEMBER;
			3 = INVALID RANGE;
			4 =INVALID VALUE;
			5=READ ONLY
			others are reserved.

If the operation of setting attributes is successful, the Status shall be SUCCESS; if the MIB does not have the requested attributes, the Status shall be INVALID ATTRIBUTE; if the MIB does not have the requested attribute members, the Status shall be INVALID ATTRIBUTE MEMBER; if AttributeValue is out of valid range, the Status shall be INVALID VALUE. If the MemberID is not 255 and MIB does not have the requested records indexed with FirstStoreIndex, the Status shall be INVALID RANGE.

7 Physical layer

7.1 General

The physical layer of WIA-FA is based on the IEEE STD 802.11-2012 PHY. WIA-FA supports different modulation modes (FHSS, DSSS, OFDM, etc.) in IEEE STD 802.11-2012.

7.2 General requirements based on IEEE STD 802.11-2012

Table 31 specifies the IEEE STD 802.11-2012 PHY selection for a WIA-FA device.

Table 31 – PHY protocol selection

Clause/ Subclause	Header	Presence	Constraints
7	PHY service specification		
7.1	Scope	YES	
7.2	PHY functions	YES	
7.3			
7.3.1	Scope and field of application	YES	
7.3.2	Overview of the service	YES	
7.3.3	Overview of interactions	YES	
7.3.4			
7.3.4.1	General	YES	
7.3.4.2	PHY-SAP peer-to-peer service primitives	YES	
7.3.4.3	PHY-SAP sublayer-to-sublayer service primitives	YES	
7.3.4.4	PHY-SAP service primitives parameters	YES	
7.3.4.5	Vector descriptions	YES	
7.3.5			
7.3.5.1	Introduction	YES	

Clause/ Subclause	Header	Presence	Constraints
7.3.5.2	PHY-DATA.request	YES	
7.3.5.3	PHY-DATA.indication	YES	
7.3.5.4	PHY-DATA.confirm	YES	
7.3.5.5	PHY-TXSTART.request	YES	
7.3.5.6	PHY-TXSTART.confirm	YES	
7.3.5.7	PHY-TXEND.request	YES	
7.3.5.8	PHY-TXEND.confirm	YES	
7.3.5.9	PHY-CCARESET.request	NO	
7.3.5.10	PHY-CCARESET.confirm	NO	
7.3.5.11	PHY-CCA.indication	NO	
7.3.5.12	PHY-RXSTART.indication	YES	
7.3.5.13	PHY-RXEND.indication	YES	
7.3.5.14	PHY-CONFIG.request	YES	
7.3.5.15	PHY-CONFIG.confirm	YES	
7.4	PHY management	YES	
14	Frequency-Hopping spread spectrum (FHSS) PHY specification for the 2,4 GHz industrial, scientific, and medical (ISM) band	YES	
15	Infrared (IR) PHY specification	NO	
16	DSSS PHY specification for the 2,4 GHz band designated for ISM applications	YES	
17	High Rate direct sequence spread spectrum (HR/DSSS) PHY specification	YES	
18	Orthogonal frequency division multiplex (OFDM) PHY specification	Partial	2,4 GHz
19	Extended rate PHY (ERP) specification	Partial	2,4 GHz
20	High Throughput (HT) PHY specification	Partial	2,4 GHz

7.3 Additional requirements

7.3.1 General

A device compliant with this document shall operate in the license-free 2,4 GHz band using modulations of IEEE STD 802.11-2012. The frequency band, channels, transmission power, and the data rate are defined specifically.

7.3.2 Frequency band

WIA-FA devices operate in the license-free 2,4 GHz band. Different countries define different 2,4 GHz frequency band (see IEEE STD 802.11-2012 PHY). For example, the 2,4 GHz frequency band in China/USA/Europe is defined from 2,4 GHz to 2,483 5 GHz, and the 2,4 GHz frequency band in Japan is defined from 2,471 GHz to 2,497 GHz.

7.3.3 Channel bitmap

WIA-FA devices use BitMap (see 6.7.1.2.1) to describe the usable channels. The BitMap is defined by 3 octets to indicate modulation modes and channel states. The format of the BitMap is shown in Figure 22.

Bits: 0 to 3	Bits: 4 to 17	Bits: 18 to 23
Modulation modes	Channel states	Reserved

IEC

Figure 22 – BitMap format

BitMap shall be composed of the following fields.

- Modulation modes: length is 4 bits. See Table 32 for detailed coding of modulation modes.
- Channel states: Bits 4 to 17 indicate channels in different modulation modes (see Table 33). Bit 4 indicates the available state of Channel 1; the value of 1 indicates the channel is available and the value of 0 indicates the channel is not available. Bits 5 to 17 indicated the available states of Channel 2 to Channel 14 are the same as Bit 4.

The coding method of Modulation modes in Figure 22 according to IEEE STD 802.11-2012 is shown in Table 32.

Table 32 – Coding of Modulation modes

Bit 3 to Bit 0	Modulation
0000	FHSS
0001	DSSS
0010	HR/DSSS
0011	OFDM
0100	ERP-DSSS
0101	ERP-CCK
0110	ERP-OFDM
0111	ERP-PBCC
1000	DSSS-OFDM
1001	No-HT mode
1010	Mix mode
1011	HT mode

The supported channels are shown in Table 33.

Table 33 – Channel indices

Modulation		Supported channels
FHSS		
DSSS		Channels indexed from 1 to 14
HR/DSSS		Channels indexed from 1 to 14
OFDM		Not defined, maximum 14 channels
ERP	ERP-DSSS	Channels indexed from 1 to 14
	ERP-CCK	
	ERP-OFDM	
	ERP-PBCC	
	DSSS-OFDM	
HT	No-HT model	Channels indexed from 1 to 14 for 20 MHz channel spacing; Channels indexed from 1 to 9 or 5 to 13 for 40 MHz channel spacing
	Mix mode	
	HT mode	

7.3.4 Transmit power

The maximum transmit power level shall be according to the regional regulations.

NOTE This feature is intended to support the regulatory requirements that apply where the device is deployed.

7.3.5 Data rate

A WIA-FA device shall support the data rates defined on the 2,4 GHz band.

The data rates supported by WIA-FA devices shall be as shown in Table 34.

Table 34 – Data rate

Modulation		Data rate (in Mbps)
FHSS		1/1,5/2/2,5/3/3,5/4/4,5
DSSS		1/2
HR/DSSS		1/2/5,5/11
OFDM		1,5/2,25/3/4,5/6/9/12/13,5 for 5 MHz channel spacing (support of 1,5/3//6 Mb/s data rates is mandatory)
ERP	ERP-DSSS	1/2
	ERP-CCK	5,5/11
	ERP-OFDM	6/9/12/18/24/36/48/54
	ERP-PBCC	5,5/11/22/33
	DSSS-OFDM	6/9/12/18/24/36/48/54
HT	No-HT model	Support rates of ERP PHY
	Mix mode	Not defined
	HT mode	Unsupported

8 Data Link Layer

8.1 General

8.1.1 DLL functions

The WIA-FA Data Link Layer (DLL) is designed to guarantee real-time, reliable and secure communication between WIA-FA field devices and access devices. WIA-FA DLL shall include:

- DLL data transport functions: adopting TDMA mechanism based on superframe, to avoid transmission collisions between frames, and ensure the reliability and real-time performance of transmission; supporting frame aggregation/disaggregation, etc.
- DLL management functions: defining device joining, leaving, time synchronization, remote attribute get/set, etc.

8.1.2 Protocol architecture

WIA-FA DLL protocol architecture is shown in Figure 23. WIA-FA DLL shall provide service interfaces for AL. DLL shall include DLL data entity (DLDE) and DLL management entity (DLME). DLDE shall provide data service interface DLDE-SAP; DLME shall provide management interface DLME-SAP for joining, leaving, time synchronization, configuring parameters, and monitoring running status of the network, etc.

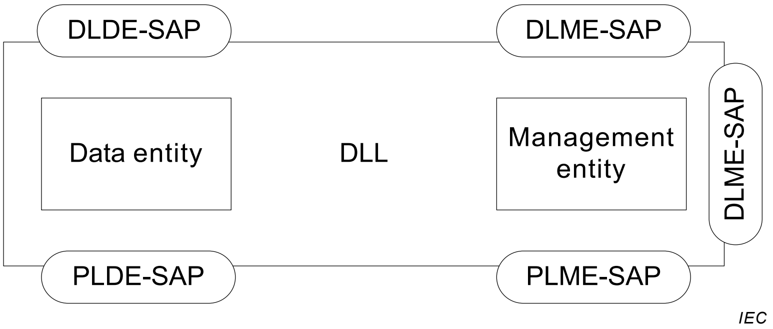


Figure 23 – WIA-FA DLL protocol architecture

8.1.3 WIA-FA superframe

WIA-FA superframe shall adopt TDMA access mechanism to realize reliability and real-time performance of data transmission. WIA-FA superframe is a collection of timeslots cyclically repeating at a constant rate. The template of a timeslot is shown in Figure 24. The length of a timeslot shall be configurable, and each timeslot shall be only used for the transmission of one frame. A timeslot of WIA-FA superframe together with channels is assigned to a link and each link is specified by a timeslot and a channel. Table 35 gives parameters of the timeslot template.

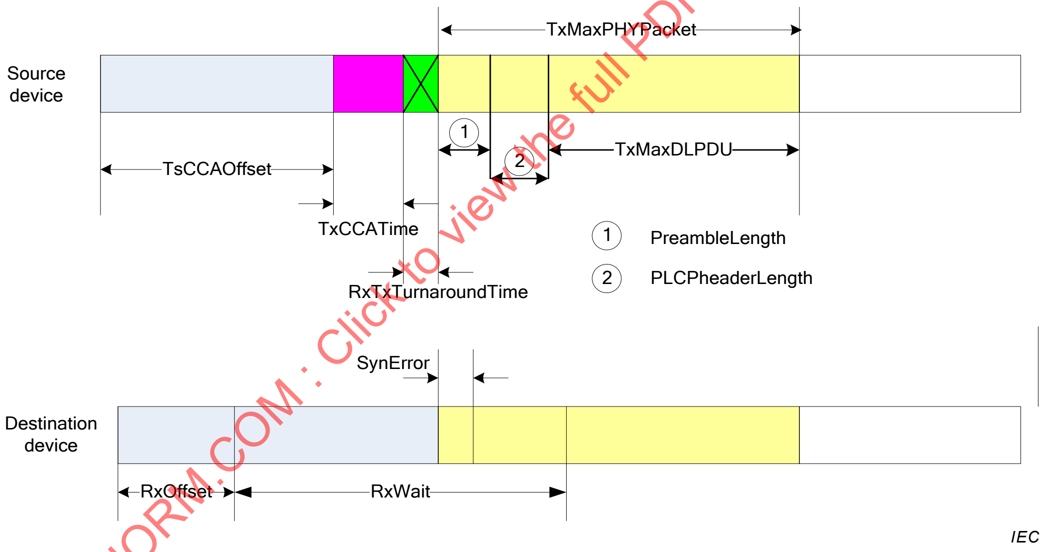


Figure 24 – The template of timeslot

The parameters of timeslot template are defined in Table 35.

Table 35 – Parameters of timeslot template

Parameter name	Description
TsCCAOffset	Time from the beginning of a timeslot to the start time of CCA (in µs)
TsCCATime	Time of performing CCA (8 symbols)
RxTxTurnaroundTime	Maximum switch time of Rx to Tx transition
PreambleLength	Transmission time of physical layer preamble
PLCPheaderLength	Transmission time of PLCP head
TxMaxDPDU	Transmission time of the longest DLPDU
TxMaxPHYPacket	Transmission time of the longest physical layer frame; the value is PreambleLength+PLCPheaderLength+TxMaxDLPDU (in µs).
RxOffset	Time from the beginning of a timeslot to the beginning of transceiver sensing
RxWait	Minimum time of destination device waiting to start receiving a frame, which is influenced by time shift.
SynError	Time difference from the actual start time of a frame to the ideal time. In other words, this is how much the receiving device perceives the transmitting device to be out of sync.(in µs).

After WIA-FA gateway device initializes, it shall firstly generate and maintain a default superframe. As shown in Figure 25, the default superframe consists of beacon timeslots, management timeslots, and data timeslots. The record of default superframe in SuperframeList attribute (see Table 18 in 6.7.1.2.2) shall be set as follows:

- SuperframeID is set to 0;
- NumberSlots is set to 512;
- ActiveFlag and ActiveSlot are set to 0; ActiveFlag and ActiveSlot can be modified by users.

The structure of the default superframe shall be broadcast by Beacon frame, and Beacon frame format is shown in 8.4.6.

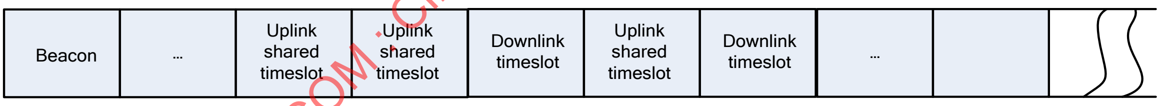


Figure 25 – WIA-FA default superframe

The functions of different kinds of timeslots in default superframe are shown as follows:

- Beacon timeslots shall be used by access devices to broadcast beacon frames for field device joining the network (see 8.3.3.5).
- Uplink shared timeslots shall be used by field devices for sending frames to access devices, including join request frames, leave response frames, time synchronization request frames, remote attribute get/set response frames for application configuration, VCR configuration, communication resource allocation, etc. (see 8.4 for frame formats).
- Downlink timeslots shall be used by access devices to send frames to field devices, including join response frames, leave request frames, time synchronization response frames, remote attribute get/set request frames for application configuration, VCR configuration, communication resource allocation, etc. (see 8.4 for frame formats).

After a field device joins the network, NM shall configure multiple superframes for it. The number of superframes and the length of these superframes shall be determined by report cycles and data update rates of UAOs in field devices. UAOs with the same data update rate or report cycle shall share the same superframe. That is, if device UAO has N data update rates and report cycles, N superframes should be configured. For example, if the data update rates of UAOs are 1 ms, 2 ms, and 16 ms, the device status report cycle and channel condition report cycle are 2 ms and 16 ms, the field device should be configured with 4 superframes, which include the default superframe (131 ms). Gateway device maintains all devices' superframes in the network. The number of superframe and the superframe structure in an access device are the same as those of managed field devices. The structure of a configured superframe is shown in Figure 26.

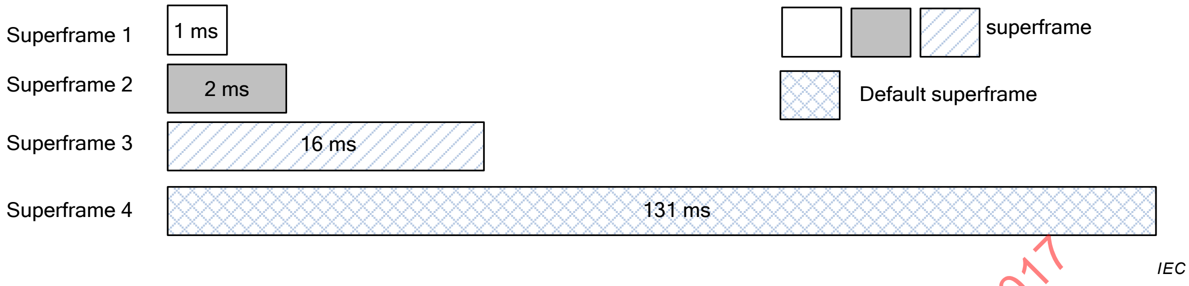


Figure 26 – WIA-FA superframe

WIA-FA supports multi-channel access. WIA-FA network devices adopt TDMA access mechanism to communicate at different channels. As shown in Figure 27, WIA-FA network devices use multiple channels to communicate.

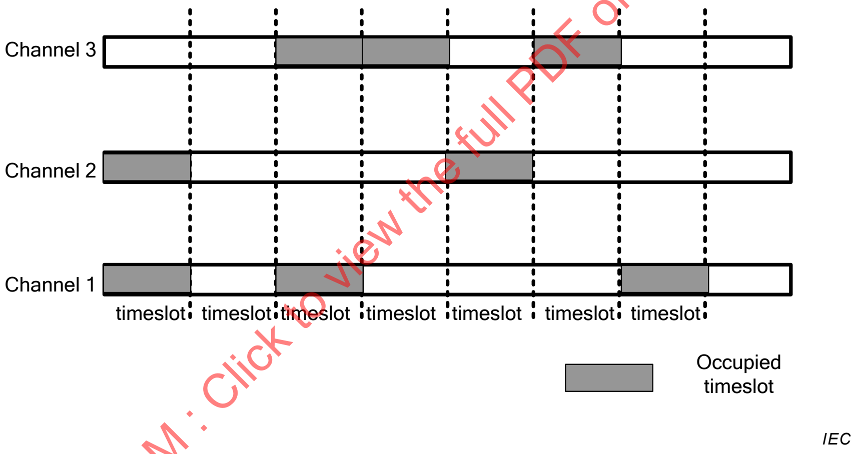


Figure 27 – The example of WIA-FA devices multi-channel communication

8.1.4 Communication based on multiple access devices

8.1.4.1 General

WIA-FA network allows having multiple access devices. Multiple access devices connect with the gateway device by wires (see Clause 9). NM in the gateway device shall allocate a unique AdID for each access device (see 9.2). Beacon frames in the WIA-FA network shall be broadcast by access devices.

NM may divide the access devices into multiple sets. Each set is allocated multiple available channels specified by BitMap (see 6.7.1.2.1). Access devices in different sets work in parallel in different channels. When a field device chooses a channel to send a join request via listening to beacon frames, it joins the set to which this channel belongs. All access devices in the same set are transparent to a field device. In other words, when a field device sends a frame, all access devices in the same set should receive this frame. The algorithm of dividing sets is out of the scope of this document.

8.1.4.2 Beacon communication based on multiple access devices

WIA-FA supports a grouping strategy for broadcasting beacons. Access devices in each set could be divided into several teams to broadcast beacon frames, and the number of teams is indicated by ADTeamNum (see 6.7.1.2.1).

The grouping strategy for broadcasting beacons includes the following steps:

- The default superframe is divided into several segments. The number of segments equals to ADTeamNum;
- Allocating beacon timeslots at the beginning of each segment for each team. The number of allocated beacon timeslots in one segment equals to the number of access devices in one team.
- Calculating the beacon timeslot number for each access device in a team as

$$[SuperframeLength/ADTeamNum] \times TeamID + InTeamID$$

where

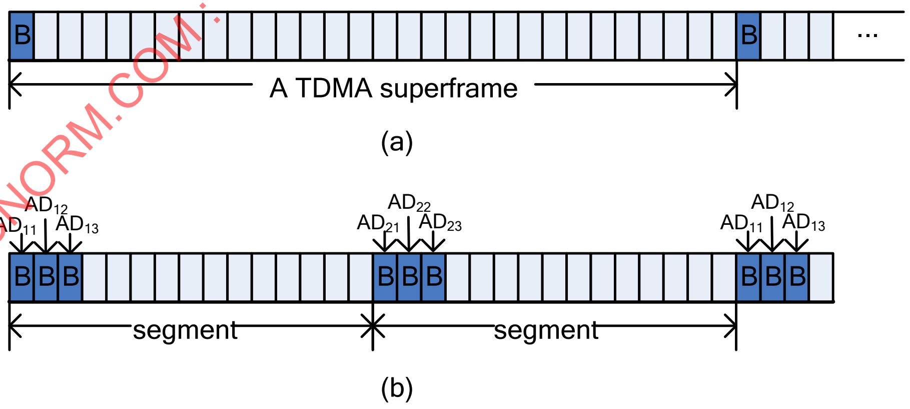
SuperframeLength is the length of the default superframe;

TeamID is the team identifier, indexed from 0;

InTeamID is the AD order within a team, indexed from 0.

Each field device synchronizes with the access device if it receives a Beacon frame from any access device during SuperframeLength. The time precision is improved by multiple time synchronizations within multiple segments.

An example of a TDMA superframe for an access device is shown in Figure 28(a). In Figure 28(a), the superframe length is 30 timeslots and the timeslot 0 is used for broadcast beacon; Figure 28(b) gives an example of a superframe for multiple access devices. Six access devices are divided into two teams. AD11, AD12, and AD13 in one team, and AD21, AD22, and AD23 are in another team. The superframe in Figure 28(b) is divided into two segments and the first three timeslots in each segment are used for broadcasting beacon frames.



IEC

Figure 28 – An example of beacon communication based on multiple ADs

8.1.4.3 Other frame communication based on multiple access devices

When a field device sends a frame according to the LinkList attributes (see 6.7.1.2.2) distributed by NM, the gateway device may receive the same frames from multiple access devices. The gateway device shall filter duplicate frames through the sequence number (see 8.4.1).

NM should choose some access devices and allocates communication resources to them for sending frames to field devices. Specific selection algorithm in NM is out of the scope of this document.

8.1.5 Time synchronization

The local clock of the gateway device should be clock source in WIA-FA network. All access devices shall keep strict time synchronization with the gateway device by wires based on IEC 61588. According to IEC 61588, WIA-FA gateway device and access devices act as ordinary clock nodes, they adopt Request_Response mechanism to get time difference.

WIA-FA supports both one-way time synchronization and two-way synchronization between access devices and field devices.

- If TwoWayTimeSyn (see 6.7.1.2.1) is set to 0, field devices shall implement the one-way time synchronization with access device.
- If TwoWayTimeSyn (see 6.7.1.2.1) is set to 1, field devices shall implement the one-way time synchronization with access device before joining WIA-FA network, implement two-way time synchronization for calculating TxDelay (see 6.7.1.2.2) after joining the network, and continue one-way time synchronization compensated by TxDelay after two-way time synchronization. The TxDelay is recorded during the two-way time synchronization.

For the one-way time synchronization, a field device shall calibrate its local clock according to the Absolute time value in a Beacon frame after receiving a Beacon frame, in order to keep the time synchronization. The process of the one-way time synchronization is shown in Figure 29. See 8.4.6 for the format of a Beacon frame.

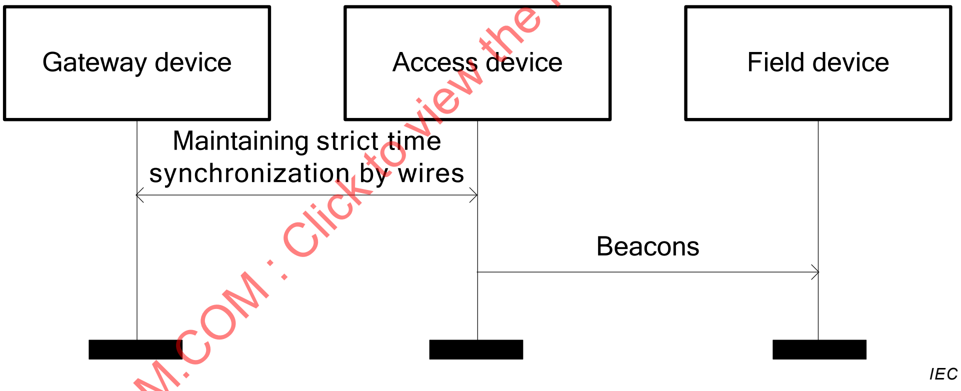


Figure 29 – Process of one-way time synchronization

For the two-way time synchronization, shown in Figure 30, a field device shall send a Time synchronization request frame (see 8.4.13) to an access device in Uplink shared timeslot after receiving a Beacon frame, and record the receiving time of this Beacon frame. An access device shall send the Time synchronization response frame (see 8.4.14) to the field device in Downlink timeslot. The payload of the Time synchronization response frame contains FieldDeviceTimeValue and ReceiveTimeValue. The field device shall calculate TxDelay according to Absolute time value in Beacon frame, receiving time of this beacon, FieldDeviceTimeValue, and ReceiveTimeValue. TxDelay shall be calculated as:

$$\text{TxDelay} = ((\text{receiving time of this beacon} - \text{Absolute time value in Beacon frame}) + (\text{ReceiveTimeValue- FieldDeviceTimeValue}))/2$$

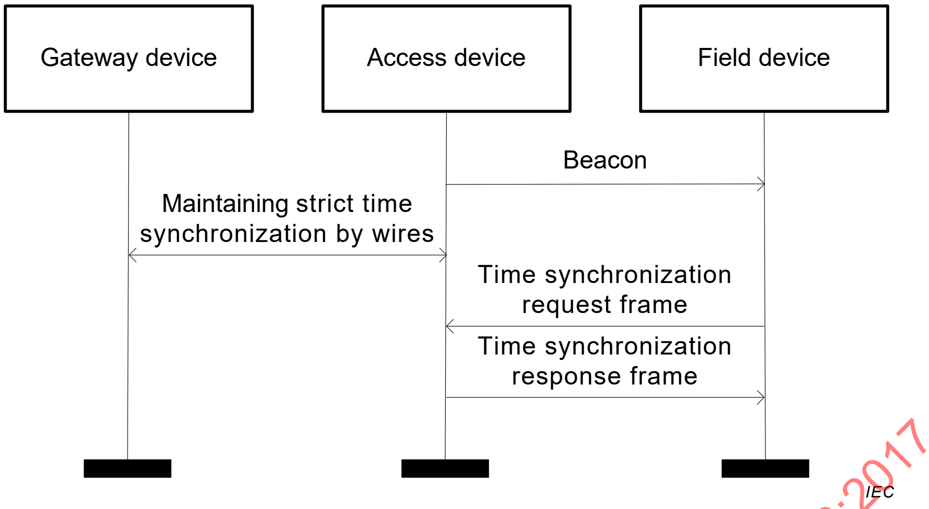

Gateway device
Access device
Field device
Beacon
Maintaining strict time synchronization by wires
Time synchronization request frame
Time synchronization response frame
IEC

Figure 30 – Process of two-way time synchronization

8.1.6 Frame aggregation/disaggregation

The DLL of WIA-FA supports frame aggregation/disaggregation mechanism to reduce the number of transmitted frames. This mechanism is optional and indicated by AGGSupportFlag (see 6.7.1.2.2).

NOTE The device specific aggregation/disaggregation is not supported.

The frame aggregation shall apply to RT1 and RT3 data. The aggregated frames should be in same priority, i.e., RT1 or RT3.

If AGGSupportFlag is 1, access devices shall implement frame aggregation and field device shall implement disaggregation. When the DLL of an access device intends to send N data frames to field devices, the DLL of access device aggregates those data frames. If the length of Aggregation Frame Payload exceeds MaxPayloadLength, the payload shall be fragmented. After receiving aggregated data frame, each field device shall disaggregate the aggregated frame and get its own frame. The aggregation/disaggregation mechanism can reduce the number of frames from access devices to field devices and can improve the network capacity.

The configuration process of aggregation function is as follows:

- Host computer reads the AGGSupportFlag (see 6.7.1.2.2) from field devices and access devices to determine whether they support frame aggregation/disaggregation function. If the AGGSupportFlag of both field devices and access devices are 1, host computer continues the following configuration process. Otherwise, AGGEnableFlag (see 6.7.1.2.1) in all the WIA-FA devices are set to 0.
- If the aggregation/disaggregation function is enabled, NM shall set AGGEnableFlag in all WIA-FA devices to 1.

The frame aggregation/disaggregation function of an access device is performed as follows:

- If the value of AGGEnableFlag is 0, the access device shall not enable the frame aggregation/disaggregation function.
- If the value of AGGEnableFlag is 1, the access device shall enable the frame aggregation/disaggregation function. The access devices aggregate multiple frames for multiple field devices according to the format of the aggregated frame shown in Figure 31. DLL of the access device shall set Frame Type (see 8.4.1) to Aggregation frame, and use broadcast timeslots (receiving timeslot in the corresponding field devices) which NM pre-allocated for access devices to send an aggregation frame.

	The first frame			…	The nth frame		
1 octet	1/2 octets	2 octets	Variable	…	1/2 octets	2 octets	Variable
Aggregated number	Field device address	Data length	Data	…	Field device address	Data length	Data

IEC

Figure 31 – Aggregation frame payload format

Each field in Figure 31 is defined as follows:

- Aggregation number: 1 octet, indicating the number of frames that is aggregated to send to field devices.
- Field device address: 1 or 2 octets, indicating the destination address of following aggregated data.
- Data length: 2 octets, representing the length of data that is sent to a field device.
- Data: variable length, representing the data that is sent to a field device.

The disaggregation function is set and used as follows:

- If AGGSupportFlag of field devices and access devices is 1, and NM set AGGEnableFlag of field devices and access devices to 1, then aggregation/disaggregation function is enabled at the same time, i.e. access devices enable frame aggregation function, meanwhile, field devices enable disaggregation function.
- If Frame Type (see 8.4.1) of a received frame indicates an Aggregation frame, the field device shall disaggregate the Aggregation Frame Payload according to the format of aggregation frame (see Figure 31).

8.1.7 Fragmentation and reassembly

Fragmentation and reassembly are handled at the DLL. If the length of DLL data frame payload is longer than MaxPayloadLength, the DLPDU should be fragmented at the DLL of the sender. When the fragmented DLPDUs reach the receiver, they are reassembled at the DLL. See packet format in 8.4.1 for detailed implementation of fragmentation and reassembly.

8.1.8 Retransmission

8.1.8.1 Retransmission modes

WIA-FA supports the following retransmission modes:

- NACK-based retransmission mode: when a field device sends data to the gateway device periodically, it shall adopt the retransmission mode based on NACK.
- Multi-unicast retransmission mode: when the gateway device sends non-aggregated data to a field device, it shall unicast the same frame multiple times to this field device.
- Multi-broadcast retransmission mode: when the gateway device sends non-aggregated broadcast frame or periodic aggregation frame, it shall broadcast the same Aggregation frame multiple times to field devices.
- GACK-based timeslot backoff mode: when a field device sends a non-periodic data frame or management frame (e.g., remote attribute get, remote attribute set, two-way time synchronization) to the gateway device, it shall retry by backoff the retransmission timeslot according to GACK.

8.1.8.2 NACK-based retransmission mode

The NACK retransmission mode is realized according to the following steps.

- The NM shall allocate several groups of retransmission timeslots for periodic data exchange between field devices and the gateway device. These periodic data can be data or management frames. The count of groups and the count of timeslots in each group are determined by Lossrate and TargetLossRate (see 6.7.1.2.1).

A method of reserving retransmission timeslots is designed as follows.

The minimum times of retransmission minRetryTime is calculated as

TargetLossRate = Lossrate to the power of minRetryTime

Then, the retransmission times MaxRetry should satisfy

MaxRetry >= minRetryTime (see 6.7.1.2.1 for MaxRetry).

The count of the n-th group retransmission timeslots minRetrySlotNum[n] should satisfy

$$\text{minRetrySlotNum}[n] >= \text{FrameCount} \times \text{LossRate}^{n},$$

where

FrameCount is the count of periodic frames sent to the gateway device during one superframe.

n is the retry times.

– A field device firstly uses an allocated timeslot to send a periodic data/ management frame to the gateway device.

– The gateway device generates NACK frame (see 8.4.4) in a certain order after receiving multiple periodic data/management frames and broadcasts the NACK frame multiple times. The NACK frame includes the short addresses and relative timeslot number of the failure field devices from which the gateway device does not receive periodic data/management frames in scheduled timeslots. NACK frames should be sent multiple times to ensure the network reliability. The repeating count of NACK frames is MaxRetry.

– Field devices parse the received NACK frames. If the payload of the NACK frame has the address of a field device, this field device shall retransmit its periodic frame by using the retransmission timeslot in the order indicated by the NACK payload. If the scheduled retransmission timeslots are not enough for field device retransmissions, the field device shall postpone its retransmissions according to next NACK.

Figure 32 is an example of NACK retransmission mode. Supposing that Lossrate is 0,1 and TargetLossRate is 0,01 %, then minRetryTime = 4. Let MaxRetry be 4. Each field device firstly uses scheduled timeslots to send periodic data/management frames to the gateway device. The gateway device generates NACK frame and broadcasts 4 times to field devices indicating whether it receives frames from field devices. Field devices retry periodic data/management frames in scheduled retransmission timeslots according to the sequence indicated in NACK frame payload.

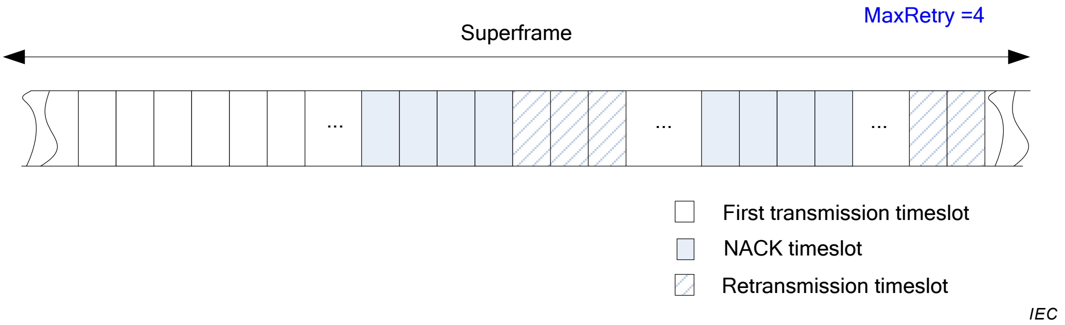


Figure 32 – Example of NACK-based retransmission mode

8.1.8.3 Multi-unicast retransmission mode

The multi-unicast retransmission mode is used by the gateway device to send non-aggregated and periodic data/management frames. NM should choose some access devices and allocates communication resources to them for multi-unicast retransmission to field devices. The total retransmission times of a non-aggregated or a periodic data/management frame sent to a field device via these selected access devices should be determined by MaxRetry (see 6.7.1.2.1).

An example of multi-unicast retransmission mode is shown in Figure 33.

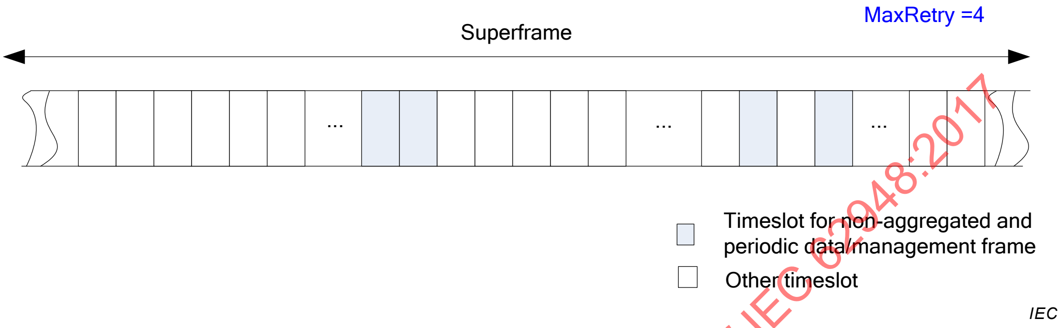

MaxRetry =4
Superframe
...
...
...
Timeslot for non-aggregated and periodic data/management frame
Other timeslot
IEC

Figure 33 – Example of multi-unicast retransmission mode

8.1.8.4 Multi-broadcast retransmission mode

The gateway device shall send a non-aggregated broadcast frame or a periodic aggregation frame to field devices by using multi-broadcast retransmission mode. NM should choose some access devices and allocates communication resources to them for multi-broadcast retransmission to field devices. The total retransmission times of a non-aggregated broadcast frame or a periodic aggregation frame sent to a field device via these selected access devices should be determined by MaxRetry (see 6.7.1.2.1).

An example of multi-broadcast retransmission mode is shown in Figure 34.

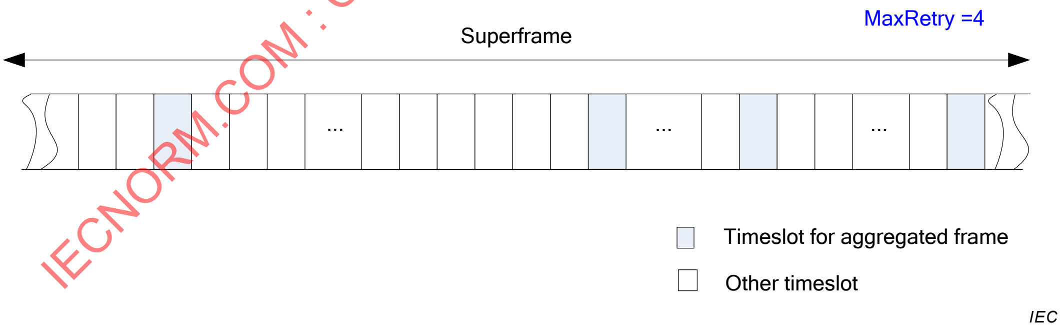


Figure 34 – Example of multi-broadcast retransmission mode

8.1.8.5 GACK-based timeslot backoff mode

GACK frames should be sent multiple times to ensure the network reliability. Field devices utilize the timeslot backoff retransmission mode to retry aperiodic data/management frames (Remote attribute get frame, Remote attribute set frame, Time synchronization frame, etc.) to the gateway device based on GACKs (see 8.4.5).

After the gateway device receives aperiodic data/management frames from multiple field devices, it generates a GACK frame (see 8.4.5) according to the addresses of these field devices. The generated GACK is broadcast by multiple times, as in the NACK-based retransmission mode (see 8.1.8.2). If a field device does not receive a GACK frame or the received GACK frame does not include its address, the field device retries the related aperiodic data/management frame by using the timeslot backoff method to compete retransmission timeslots.

The transmit-shared timeslots (see LinkList in 6.7.1.2.2) in each superframe shall be used by field devices to retry aperiodic data/management frames. If a field device does not receive a GACK frame or the received GACK frame does not include its address or sequence number, the field device competes the retransmitting timeslots for sending aperiodic data/management frames. If the competition is failure, the field device shall delay its retransmission to next retransmitting timeslot until MaxRetry (see 6.7.1.2.1).

GACK information for frames with different GACK address mode shall not be packed into one frame. If GACKs are used for acknowledging join request frames, the Successful Device address in GACK information shall only be the long address (8 octets) of the field device attempting to join the network. Otherwise, the Devices address in GACK information shall only be the short address (1/2 octets) of the field device.

An example of GACK-based timeslot backoff mode is shown in Figure 35.

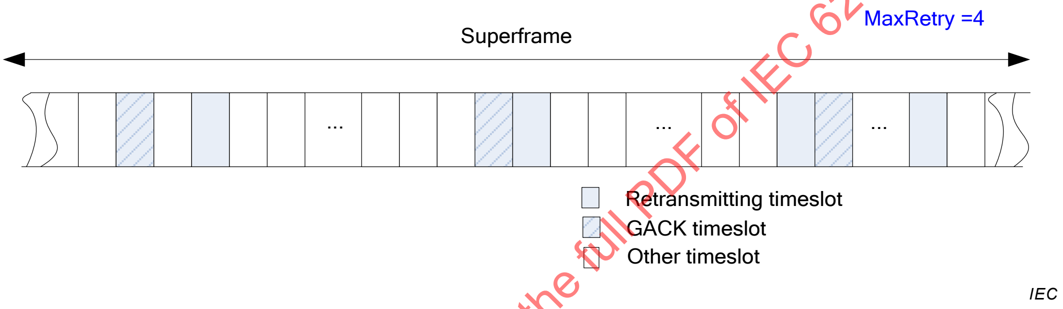


Figure 35 – Example of GACK-based timeslot backoff mode

8.2 Data link sub-layer data services

8.2.1 General

DLDE-SAP supports the point-to-point transmission between an access device and a field device. The DLL data services primitive includes DLDE-DATA.request, DLDE-DATA.confirm, and DLDE-DATA.indication.

8.2.2 DLDE-DATA.request primitive

ASL invokes DLDE-DATA.request primitive to send data.

The semantics of DLDE-DATA.request primitive are as follows:

```
DLDE-DATA.request(
                    DstAddr,
                    VCR_ID,
                    DataType,
                    Priority,
                    PayloadLength,
                    Payload
                    )
```

Table 36 specifies the parameters of DLDE-DATA.request primitive.

Table 36 – DLDE-DATA.request primitive parameters

Name	Data type	Valid range	Description
DstAddr	Unsigned16	0 to 65 535	The short address of destination device
VCR_ID	Unsigned16	0 to 65535	VCR identifier for data. VCR_ID is invalid only if DataType =0 VCR _ID is used for searching DateUpdateRate and UAP_ID DateUpdateRate is used to search communication resources (in ms) DataUpdateRate = 0 and Priority = 2 or 4, use default superframe; DataUpdateRate = 0 and Priority =0, preempt the next timeslot allocated; DataUpdateRate > 0, use the allocated superframe.

DataType	Unsigned8	0 to 255	Data type:
			0=DATA;
			1= NACK;
			2=GACK;
			others are reserved.

Priority	Unsigned8	0 to 255	The priority of the payload,
			0 = RT0;
			1 = RT1;
			2 = RT2;
			3= RT3;
			4= NRT;
			others are reserved.

PayloadLength	Unsigned16	0 to 65 535	Denoting the length of payload (in octet)
Payload	Octetstring		APDU

8.2.3 DLDE-DATA.indication primitive

DLDE-DATA.indication primitive is used to report received data to ASL.

The semantics of DLDE-DATA.indication primitive are as follows:

```
DLDE-DATA.indication(
                     SrcAddr,
                     DataType,
                     PayloadLength,
                     Payload
                     )
```

Table 37 specifies the parameters of DLDE-DATA.indication.

Table 37 – DLDE-DATA.indication primitive parameters

Name	Data type	Valid range	Description
SrcAddr	Unsigned16	0 to 65 535	The short address of source device

DataType	Unsigned8	0 to 255	Data type:
			0=DATA;
			1= NACK;
			2= GACK;
			others are reserved.

PayloadLength	Unsigned16	0 to 65 535	Denoting the length of Payload (in octet)
Payload	Octetstring		DLL payload

8.2.4 Time sequence of DLL data service

Figure 36, Figure 37, and Figure 38 give the basic process of data frame transmission, reception, and acknowledgement.

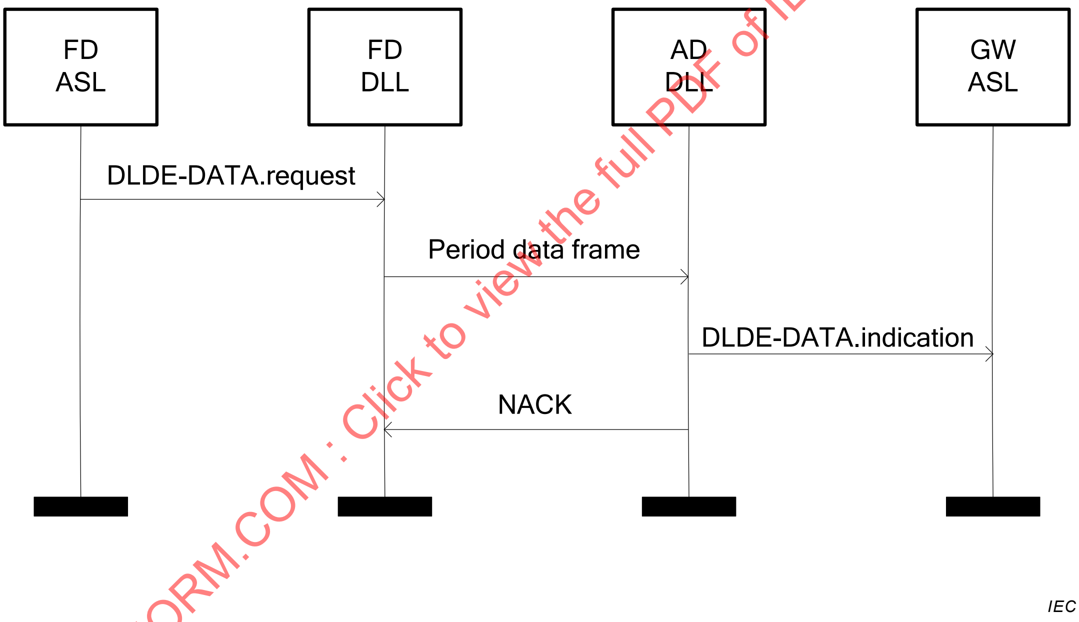


Figure 36 – Time sequence of period data service from FD to GW

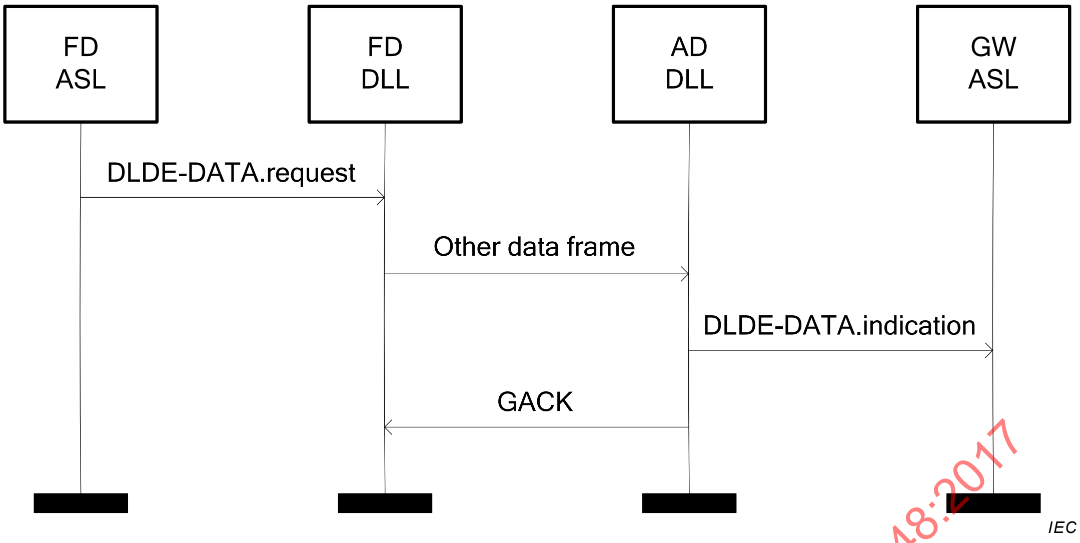


Figure 37 – Time sequence of other data service from FD to GW

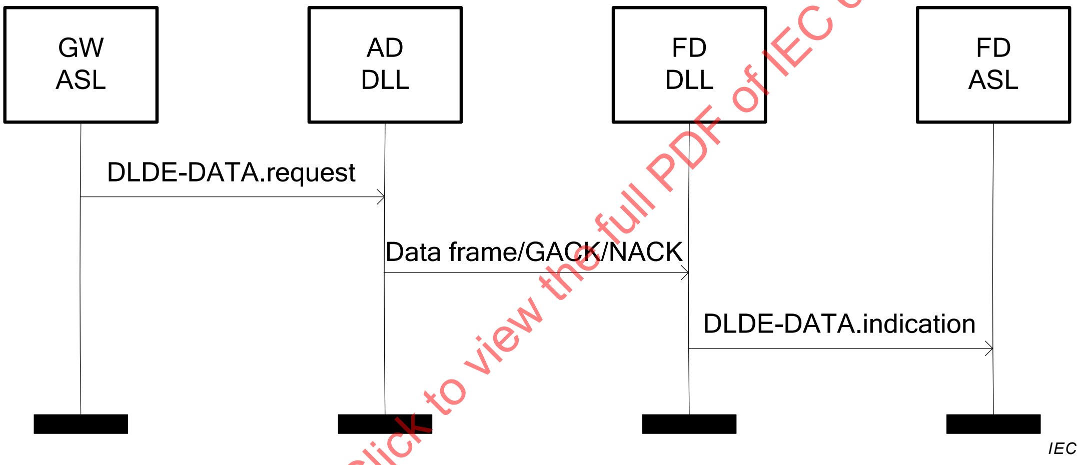


Figure 38 – Time sequence of data service from GW to FD

8.3 Data link sub-layer management services

8.3.1 General

DLME-SAP defines the way that application layer passes management commands to the data link layer. Table 38 summarises all the management services.

All request services shall be invoked by DMAP to DLL to generate a request command frame; Indication services shall be used by DLL to DMAP to report receiving a command frame; Response services shall be invoked by DMAP to DLL to generate a response frame; and confirm services shall be used by DLL to DMAP to return the transmission status of a request command frame.

Table 38 – Management services

Service name	Request	Indication	Response	Confirm
DLME-DISCOVERY	8.3.2.1			8.3.2.2
DLME-TIME-SYN	8.3.3.1	8.3.3.2	8.3.3.3	8.3.3.4
DLME-JOIN	8.3.4.1	8.3.4.2	8.3.4.3	8.3.4.4
DLME-DEVICE-STATUS	8.3.5.1	8.3.5.2		8.3.5.3
DLME-CHANNEL-CONDITION	8.3.6.1	8.3.6.2		8.3.6.3
DLME-INFO-GET	8.3.7.1	8.3.7.2	8.3.7.3	8.3.7.4
DLME-INFO-SET	8.3.8.1	8.3.8.2	8.3.8.3	8.3.8.4
DLME-LEAVE	8.3.9.1	8.3.9.2	8.3.9.3	8.3.9.4

8.3.2 Network discovery services

8.3.2.1 DLME-DISCOVERY.request

DLME-DISCOVERY.request is used to request a device to scan channels.

```
DLME-DISCOVERY.request(
                    ScanChannels
                    )
```

Table 39 specifies the parameters for DLME-DISCOVERY.request.

Table 39 – DLME-DISCOVERY.request parameters

Parameter	Data type	Valid range	Description
ScanChannels	BitField24	BitMap (see 6.7.1.2.1)	IEEE STD 802.11-2012 14 physical channels available

8.3.2.2 DLME-DISCOVERY.confirm

DLME-DISCOVERY.confirm is used to respond to DLME-DISCOVERY.request.

```
DLME-DISCOVERY.confirm(
                    Status,
                    BeaconCount,
                    SuperframeLength,
                    TimeslotDuration,
                    FirstSharedTimeslotNum,
                    SharedTimeslotCount,
                    AbsoluteTimeValue,
                    BeaconDescription
                    )
```

Table 40 specifies the parameters for DLME-DISCOVERY.confirm.

Table 40 – DLME-DISCOVERY.confirm parameters

Parameter	Data type	Valid range	Description

Status	Unsigned8	0 to 255	Scan results:
			0 = SUCCESS;
			1 = NO_BEACON;
			others are reserved.

BeaconCount	Unsigned8	0 to 255	The number of beacon frames discovered
SuperframeLength	Unsigned16	0 to 65 535	Specifies the length of default superframe in timeslot (see 8.1.3)
TimeslotDuration	Unsigned16	0 to 65 535	The timeslot length set by network (see TimeSlotDuration in 6.7.1.2.1)
FirstSharedTimeslotNumber	Unsigned16	0 to 65 535	The first shared timeslot in superframe
ShareTimeslotNumber	Unsigned8	0 to 255	The total number of shared timeslots (see 8.4.6)
AbsoluteTimeValue	TimeData	0 to ($2^{64}-1$)	The absolute time value sending the Beacon frame (see TimeValue in 6.7.1.2.2)
BeaconDescription	BeaconDescription_ Struct List		See Table 41

Table 41 – BeaconDescription_Struct parameters

Name	Data type	Valid range	Description
ChannelIndex	BitField24		The channel of receiving Beacon frame (see BitMap in 6.7.1.2.1)
BeaconRelativeTimeslot-Num	Unsigned16	0 to 65 535	The timeslot sending Beacon frame
ED	Unsigned8	0 to 255	The energy level of a received Beacon frame

8.3.2.3 Network discovery process

The process of the network discovery is shown in Figure 39.

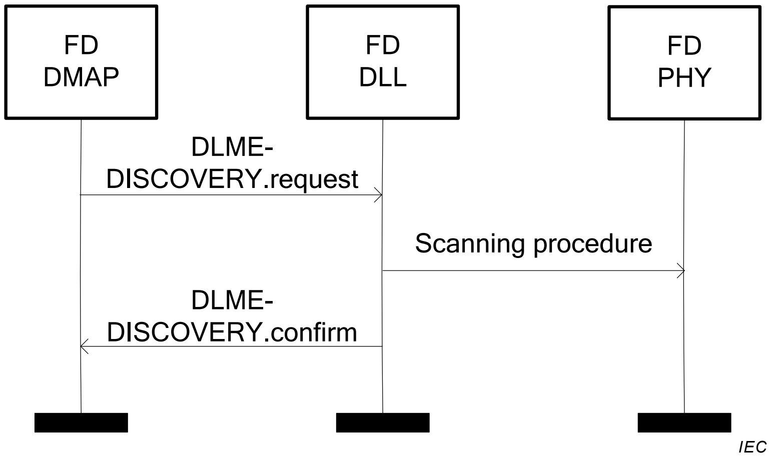


Figure 39 – Network discovery process

The field device DMAP invokes DLME-DISCOVERY.request to request the scanning procedure (see 10.1.4.3.3 in IEEE STD 802.11-2012). The DLL reports scanning results to DMAP by DLDE-DISCOVERY.confirm after scanning.

8.3.3 Time synchronization services

8.3.3.1 DLME-TIME-SYN.request

DLME-TIME-SYN.request is used to send two-way Time synchronization command frame requested by the DMAP.

```
DLME-TIME-SYN.request(
                           )
```

DLME-TIME-SYN.request parameters are null, which request physical layer to write the Absolute time value in the Time synchronization request command frame.

8.3.3.2 DLME-TIME-SYN.indication

DLME-TIME-SYN.indication is used to inform the DMAP that the Time synchronization request command frame has been successfully received.

```
DLME-TIME-SYN.indication(
                           SrcAddr,
                           FieldDeviceTimeValue
                           )
```

Table 42 specifies the parameters for DLME-TIME-SYN.indication.

Table 42 – DLME-TIME-SYN.indication parameters

Parameter	Data type	Valid range	Description
SrcAddr	Unsigned16	0 to 65 535	The short address of the field device (see DeviceShortAddress in 6.7.1.2.2)
FieldDeviceTimeValue	TimeData	0 to $(2^{64}-1)$	The timestamp when the field device sends Time synchronization request frame, in µs, see 8.4.13.

8.3.3.3 DLME-TIME-SYN.response

DLME-TIME-SYN.response is the response of DLME-TIME-SYN.indication.

```
DLME-TIME-SYN.response(
                           DstAddr,
                           FieldDeviceTimeValue,
                           ReceiveTimeValue
                           )
```

Table 43 specifies the parameters for DLME-TIME-SYN.response.

Table 43 – DLME-TIME-SYN.response parameters

Parameter	Data type	Valid range	Description
DstAddr	Unsigned16	0 to 65 535	The short address of the field device (see DeviceShortAddress in 6.7.1.2.2)
FieldDeviceTimeValue	TimeData	0 to $(2^{64}$-1)	The timestamp when the field device sends Time synchronization request command frame (in µs), see 8.4.13
ReceiveTimeValue	TimeData	0 to $(2^{64}$-1)	The time when the access device receives Time synchronization request command frame (in µs), see 8.4.14

8.3.3.4 DLME-TIME-SYN.confirm

DLME-TIME-SYN.confirm is used to respond to DLME-TIME-SYN.request. DLME-TIME-SYN.confirm is used to forward the time when the field device sends the two-way Time synchronization frame and the time when the access device receives the Time synchronization request frame (see 8.4.13).

```
DLME-TIME-SYN.confirm(
                        Status,
                        FieldDeviceTimeValue,
                        ReceiveTimeValue
                        )
```

Table 44 specifies the parameters for DLME-TIME-SYN.confirm.

Table 44 – DLME-TIME-SYN.confirm parameters

Parameter	Data type	Valid range	Description
Status	Unsigned8	0 to 255	Result of the time synchronization request: 0 = SUCCESS; 1 = OVERTIME (When the field device does not receive Time synchronization response frame within TwoWayOverTime, the time synchronization is failed); others are reserved. See 6.7.1.2.1 for TwoWayOverTime.
FieldDeviceTimeValue	TimeData	0 to $(2^{64}$-1)	The timestamp when the field device sends two-way Time synchronization request command frame, in µs, see 8.4.13
ReceiveTimeValue	TimeData	0 to $(2^{64}$-1)	The time when the access device receives two-way Time synchronization request command frame (in µs), see 8.4.13.

8.3.3.5 Time synchronization process

Time synchronization process is shown in Figure 40.

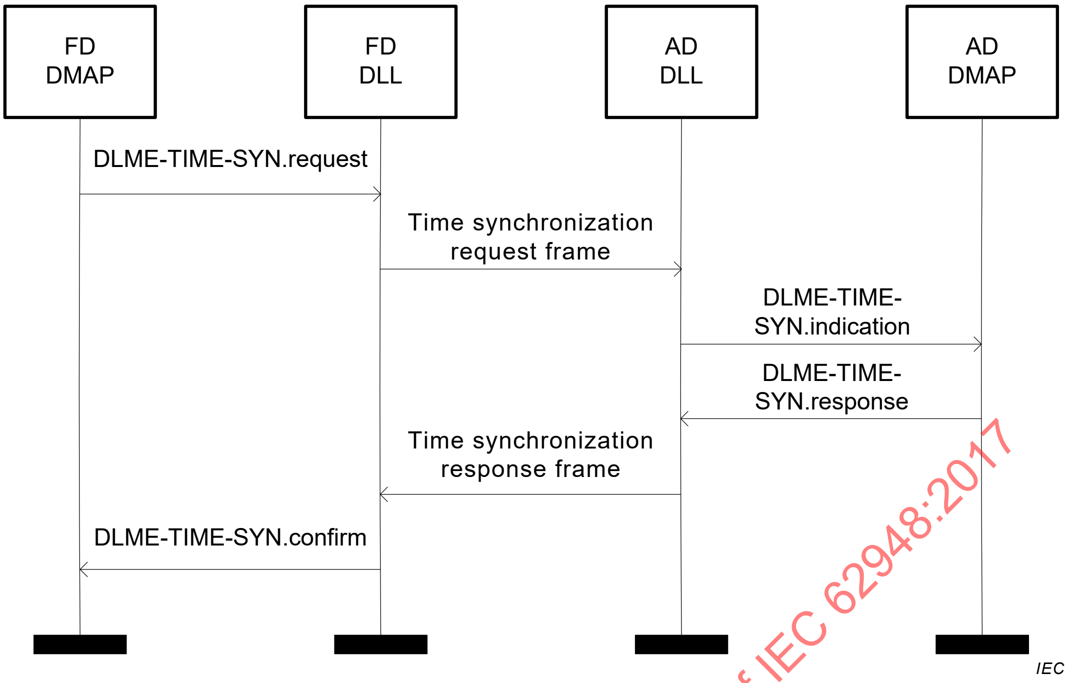


Figure 40 – Time synchronization process

The field device DMAP invokes DLME-TIME-SYN.request in DLL, indicating DLL send Time synchronization request command frame. After receiving the Time synchronization request command frame, the access device returns GACK, and reports to DMAP of the gateway device by DLME-TIME-SYN.indication. DMAP of the gateway device invokes DLME-TIME-SYN.response primitive, indicating the access device DLL send Time synchronization response command frame. After receiving the Time synchronization response command frame, the field device DLL confirms to DMAP by DLME-TIME-SYN.confirm.

8.3.4 Device joining services

8.3.4.1 DLME-JOIN.request

If a field device attempts to join WIA-FA network, DLME-JOIN.request is invoked by DMAP of the field device to request DLL generating a Join request command frame.

```
DLME-JOIN.request(
                    NetworkID,
                    Channel,
                    PhyAddr,
                    SecMaterial
                    )
```

Table 45 specifies the parameters for DLME-JOIN.request.

Table 45 – DLME-JOIN.request parameters

Parameter	Data type	Valid range	Description
NetworkID	Unsigned8	0 to 255	Network identifier, used for multiple networks coexisting (see NetworkID in 6.7.1.2.1)
Channel	BitField24	BitMap (see Table 17, 6.7.1.2.1)	Channel used for joining, chosen from valid channels supported by PHY
PhyAddr	Unsigned64	0 to $(2^{64}-1)$	Long address of the new device waiting to join (see LongAddress in 6.7.1.2.2).
SecMaterial	Unsigned64	0 to $(2^{64}-1)$	Device security material for authentication (see 11.3). If the SecLevel is 0, this parameter is ignored.

8.3.4.2 DLME-JOIN.indication

DLME-JOIN.indication is used to inform DMAP of the gateway device that the Join request frame from one device has been successfully received.

```
DLME-JOIN.indication(
                        PhyAddr,
                        SecMaterial
                        )
```

Table 46 specifies the parameters for DLME-JOIN.indication.

Table 46 – DLME-JOIN.indication parameters

Parameter	Data type	Valid range	Description
PhyAddr	Unsigned64	0 to $(2^{64-}1)$	Long address of the new device waiting to join (see LongAddress in 6.7.1.2.2).
SecMaterial	Unsigned64	0 to $(2^{64}-1)$	Device security material for authentication (see 11.3). If the SecLevel is 0, this parameter is ignored.

8.3.4.3 DLME-JOIN.response

DLME-JOIN.response is the response of DLME-JOIN.indication.

```
DLME-JOIN.response(
                        Status,
                        ShortAddr
                        )
```

Table 47 specifies the parameters for DLME-JOIN.response.

Table 47 – DLME-JOIN.response parameters

Parameter	Data type	Valid range	Description
Status	Unsigned8	0 to 255	Result of join request join request: 0 = SUCCESS 1 = NetworkID mismatched (The NetworkID of the device attempting to join network is different from the NetworkID of the gateway device); 2 = Authentication failure (SecMaterial in the Join request frame is different from the HMAC calculation result of SM, see 11.3.1); 3 = Network overload (no more short address is available); 4 = Duplicated join (a device with the same long address already joined the network); others are reserved.
ShortAddr	Unsigned16	0 to 65 535	Short address allocated by the GW to FD waiting to join (see DeviceShortAddress in 6.7.1.2.2), if Status=SUCCESS, this field is valid.

8.3.4.4 DLME-JOIN.confirm

DLME-JOIN.confirm is a response to DLME-JOIN.request.

```
DLME-JOIN.confirm(
                          Status,
                          ShortAddr
                          )
```

Table 48 specifies the parameters for DLME-JOIN.confirm.

Table 48 – DLME-JOIN.confirm parameters

Parameter	Data type	Valid range	Description
Status	Unsigned8	0 to 255	Result of join request join request: 0 = SUCCESS; 1 = NetworkID mismatched (The NetworkID of the device attempting to join network is different from the NetworkID of the gateway device); 2 = Authentication fail (SecMaterial in the Join request frame is different from the HMAC calculation result of SM, see 11.3.1); 3 = Network overload (no more short address is available); 4 = Duplicated join (a device with the same long address already joined the network); 5 = Join timeout (When the field device does not receive Join response frame within length of 256 default superframes, the join process is failed); others are reserved.

Parameter	Data type	Valid range	Description
ShortAddr	Unsigned16	0 to 65 535	Short address allocated by the GW to FD attempting to join (see DeviceShortAddress in 6.7.1.2.1), if Status=SUCCESS, this field is valid.

8.3.4.5 Device join process

Device join process is shown in Figure 41.

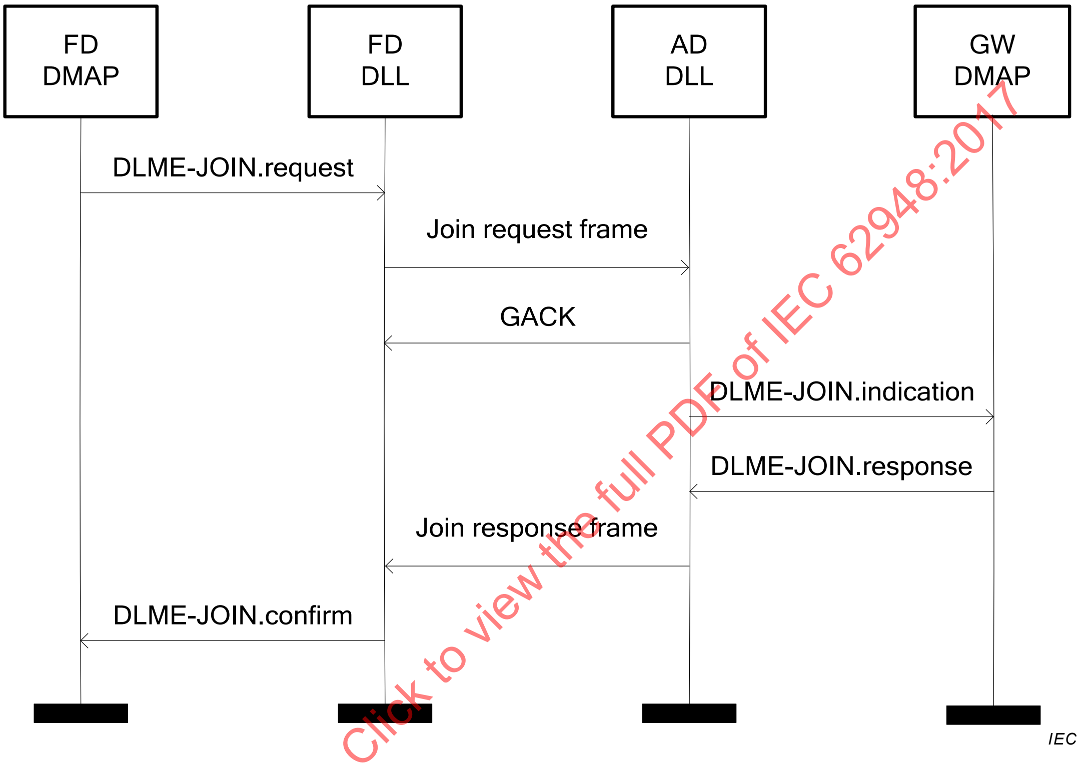


Figure 41 – Device join process

The field device DMAP invokes DLME-JOIN.request primitive in DLL, indicating DLL send Join request command frame; After receiving join request command frame, the access device returns GACK, and reports to the gateway device DMAP by DLME-JOIN.indication; The gateway device DMAP invokes DLME-JOIN.response primitive, indicating the access device DLL send Join response frame; After receiving Join response command frame, the field device DLL confirms to DMAP by DLME-JOIN.confirm.

8.3.5 Device status report services

8.3.5.1 DLME-DEVICE-STATUS.request

DLME-DEVICE-STATUS.request is used to periodically report the device status to the gateway device by device.

```
DLME-DEVICE-STATUS.request(
                                        PowerSupplyStatus
                                        )
```

Table 49 specifies the parameters for DLME-DEVICE-STATUS.request.

Table 49 – DLME-DEVICE-STATUS.request parameters

Parameter	Data type	Valid range	Description
PowerSupplyStatus	Unsigned8	0 to 255	Information of device electric power, (See PowerSupplyStatus in Device_Struct List)

8.3.5.2 DLME-DEVICE-STATUS.indication

DLME-DEVICE-STATUS.indication is used to report the receipt of a Device condition report frame to the DMAP.

```
DLME-DEVICE-STATUS.indication(
                    ShortAddr,
                    PowerSupplyStatus
                    )
```

Table 50 specifies the parameters for DLME-DEVICE -STATUS.indication.

Table 50 – DLME-DEVICE -STATUS.indication parameters

Parameter	Data type	Valid range	Description
ShortAddr	Unsigned16	00 to 65 535	The short address of a field device reporting device status (see DeviceShortAddress in 6.7.1.2.2)
PowerSupplyStatus	Unsigned8	0 to 255	Information of device electric power, (see PowerSupplyStatus in Device_Struct List)

8.3.5.3 DLME-DEVICE-STATUS.confirm

DLME-DEVICE-STATUS.confirm is used to return the results of DLME-DEVICE-STATUS.request.

```
DLME- DEVICE-STATUS.confirm(
                    Status
                    )
```

Table 51 specifies the parameters for DLME-DEVICE-STATUS.confirm.

Table 51 – DLME-DEVICE -STATUS.confirm parameters

Parameter	Data type	Valid range	Description

Status	Unsigned8	0 to 255	Results of the device status report request:
			0 = SUCCESS;
			1 = FAILURE;
			others are reserved.

8.3.5.4 Device status report process

Device status report process is shown in Figure 42.

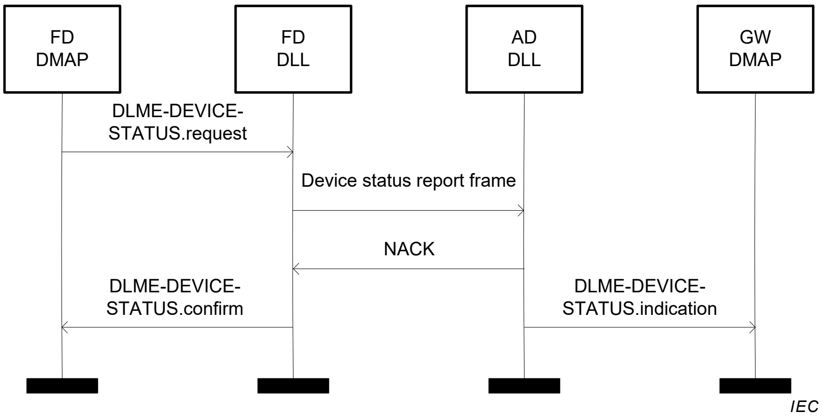


Figure 42 – Device status report process

DMAP of a field device invokes DLME-DEVICE-STATUS.request in DLL, indicating DLL send Device status report frame. After receiving Device status report frame, the access device returns NACK, and reports to the DMAP of the gateway device by DLME-DEVICE-STATUS.indication. The field device DLL confirms to DMAP by DLME-DEVICE-STATUS.confirm after receiving NACK.

8.3.6 Channel condition report services

8.3.6.1 DLME-CHANNEL-CONDITION.request

DLME-CHANNEL-CONDITION.request is used to report the communication channel condition to the gateway device by device.

DLME-CHANNEL-CONDITION.request(
Count,
ChannelConditionInfo
)

Table 52 specifies the parameters for DLME-CHANNEL-CONDITION.request.

Table 52 – DLME-CHANNEL-CONDITION.request parameters

Parameter	Data type	Valid range	Description
Count	Unsigned8	0 to 255	The number of channels requested to report
ChannelConditionInfo	ChanCon_Struct List (see)		Information of channel condition attributes

8.3.6.2 DLME-CHANNEL-CONDITION.indication

DLME-CHANNEL-CONDITION.indication is used to report the receipt of a Channel condition report frame to DMAP of the gateway device.

DLME-CHANNEL-CONDITION.indication(
SrcAddr,
Count,
ChannelConditionInfo
)

Table 53 specifies the parameters for DLME-CHANNEL-CONDITION.indication.

Table 53 – DLME-CHANNEL-CONDITION.indication parameters

Parameter	Data type	Valid range	Description
SrcAddr	Unsigned16	0 to 65 535	The short address of the device reporting channel condition (see DeviceShortAddress in 6.7.1.2.1)
Count	Unsigned8	0 to 255	The length of list of ChanCon_Struct
ChannelConditionInfo	ChanCon_Struct List (See Table 20)		Information of channel condition attributes

8.3.6.3 DLME-CHANNEL-CONDITION.confirm

DLME-CHANNEL-CONDITION.confirm is used to return the results of DLME-CHANNEL-CONDITION.request.

```
DLME-CHANNEL-CONDITION.confirm(
                                Status
                                )
```

Table 54 specifies the parameters for DLME-CHANNEL-CONDITION.confirm.

Table 54 – DLME-CHANNEL-CONDITION.confirm parameters

Parameter	Data type	Valid range	Description

Status	Unsigned8	0 to 255	Result of the channel condition report request:
			0 = SUCCESS;
			1 = FAILURE;
			others are reserved.

8.3.6.4 Channel condition report process

The process of channel condition report is shown in Figure 43.

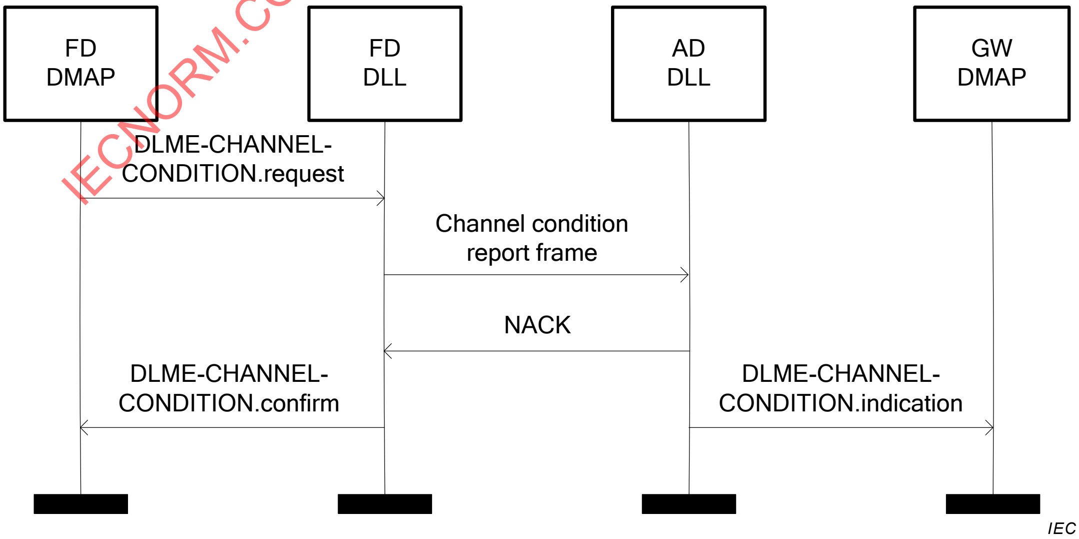

FD DMAP
FD DLL
AD DLL
GW DMAP
DLME-CHANNEL-CONDITION.request
Channel condition report frame
NACK
DLME-CHANNEL-CONDITION.confirm
DLME-CHANNEL-CONDITION.indication
IEC

Figure 43 – Channel condition report process

The DMAP of field device invokes DLME-CHANNEL-CONDITION.request in DLL, indicating DLL to send Channel condition report frame. After receiving Channel condition report frame, the access device returns NACK, and reports to DMAP of the gateway device by

DLME-CHANNEL-CONDITION.indication. If the received NACK frame does not have the field device address, the field device DLL confirms to DMAP by DLME-CHANNEL-CONDITION.confirm with Status being SUCCESS. If the retransmission count exceeds MaxRetry or Channel condition report frame expires ChaStaRptCycle, the field device DLL confirms to DMAP by DLME-CHANNEL-CONDITION.confirm with Status being FAILURE.

8.3.7 Remote attribute get services

8.3.7.1 DLME-INFO-GET.request

DLME-INFO-GET.request is used to remotely request attributes in the MIB.

```
DLME-INFO-GET.request(
                    Handle,
                    DstAddr,
                    AttributeID,
                    MemberID,
                    FirstStoreIndex,
                    Count
                    )
```

Table 55 specifies the parameters for DLME-INFO-GET.request.

Table 55 – DLME-INFO-GET.request parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DLME-INFO-GET.request
DstAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bit address of destination device. The setting attributes using 64-bit long address is supported.
AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of attribute member. The value 255 means that all attribute members should be read. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The index of the first one of records to be read. FirstStoreIndex shall be valid only for attributes of List type.
Count	Unsigned16	0 to 65 535	Number of records to be read. Count shall be valid only for attributes of List type. All records from FirstStoreIndex shall be read if Count = 0

8.3.7.2 DLME-INFO-GET.indication

DLME-INFO-GET.indication is used to inform the DMAP that Attributes get request frame has been successfully received.

```
DLME-INFO-GET.indication(
                    Handle,
                    SrcAddr,
                    AttributeID,
                    MemberID,
                    FirstStoreIndex,
                    Count
                    )
```

Table 56 specifies the parameters for DLME-INFO-GET.indication.

Table 56 – DLME-INFO-GET.indication parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from the Remote attribute get request frame
SrcAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bit address of source device. The setting attributes using 64-bit long address is supported.
AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of attribute member. The value 255 means that all attribute members should be read. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The index of the first one of records to be read. FirstStoreIndex shall be valid only for attributes of List type.
Count	Unsigned16	0 to 65 535	Number of records to be read. Count shall be valid only for attributes of List type. All records from FirstStoreIndex shall be read if Count = 0

8.3.7.3 DLME-INFO-GET.response

DLME-INFO-GET.response is used to respond to DLME-INFO-GET.indication.

```
DLME-INFO-GET.response(
                        Handle,
                        DstAddr,
                        Status,
                        AttributeID,
                        MemberID,
                        FirstStoreIndex,
                        Count,
                        AttributeValue
                        )
```

Table 57 specifies the parameters for DLME-INFO-GET.response.

Table 57 – DLME-INFO-GET.response parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from the DLME-INFO-GET.indication.
DstAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bi address of destination device. The setting attributes using 64-bit long address is supported.

Status	Unsigned8	0 to 255	Attribute getting results:
			0 = SUCCESS;
			1 = INVALID ATTRIBUTE;
			2 = INVALID ATTRIBUTE MEMBER;
			3 = INVALID RANGE;
			others are reserved.

AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of attribute member, which is used to get the structured MIB attributes. The value 255 means that all attribute members should be read. This value is not valid for the unstructured attributes
FirstStoreIndex	Unsigned16	0 to 65 535	The first storage index of multiple records, this value is not valid for the unstructured attributes
Count	Unsigned16	0 to 65 535	Number of records which has been read. Count shall be valid only for attributes of List type and Status = SUCCESS.
AttributeValue	Octectstring		Value of the attribute to be read

If the operation of getting attributes is successful, the Status shall be SUCCESS and the AttributeValue is valid; if the MIB does not have the needed attributes, the Status shall be INVALID ATTRIBUTE; if the MIB does not have the needed attribute members, the Status shall be INVALID ATTRIBUTE MEMBER; if the MIB does not have the needed records indexed with FirstStoreIndex, the Status shall be INVALID RANGE.

8.3.7.4 DLME-INFO-GET.confirm

DLME-INFO-GET.confirm is used to return the results of DLME-INFO-GET.request.

```
DLME-INFO-GET.confirm(
                     Handle,
                     SrcAddr,
                     Status,
                     AttributeID,
                     MemberID,
                     FirstStoreIndex,
                     Count,
                     AttributeValue
                     )
```

Table 58 specifies the parameters for DLME-INFO-GET.confirm.

Table 58 – DLME-INFO-GET.confirm parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from the Remote attribute get response frame.
SrcAddr	Unsigned64	0 to $(2^{64}-1)$	The 8/16/64-bit address of source device. The getting attributes using 64-bit long address is supported.

Status	Unsigned8	0 to 255	Execution result of the request
			0=SUCCESS;
			1= INVALID_ATTRIBUTE;
			2 = INVALID ATTRIBUTE MEMBER;
			3 = INVALID RANGE;
			others are reserved.

AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of attribute member, which is used to get the structured MIB attributes. The value 255 means that all attribute members should be read. This value is not valid for the unstructured attributes
FirstStoreIndex	Unsigned16	0 to 65 535	The first storage index of multiple records, this value is not valid for the unstructured attributes
Count	Unsigned16	0 to 65 535	Number of records which has been read. Count shall be valid only for attributes of List type and Status = SUCCESS.
AttributeValue	Octectstring		Value of the attribute read

8.3.7.5 Remote attribute get process

The process of the remote attribute get process is shown in Figure 44.

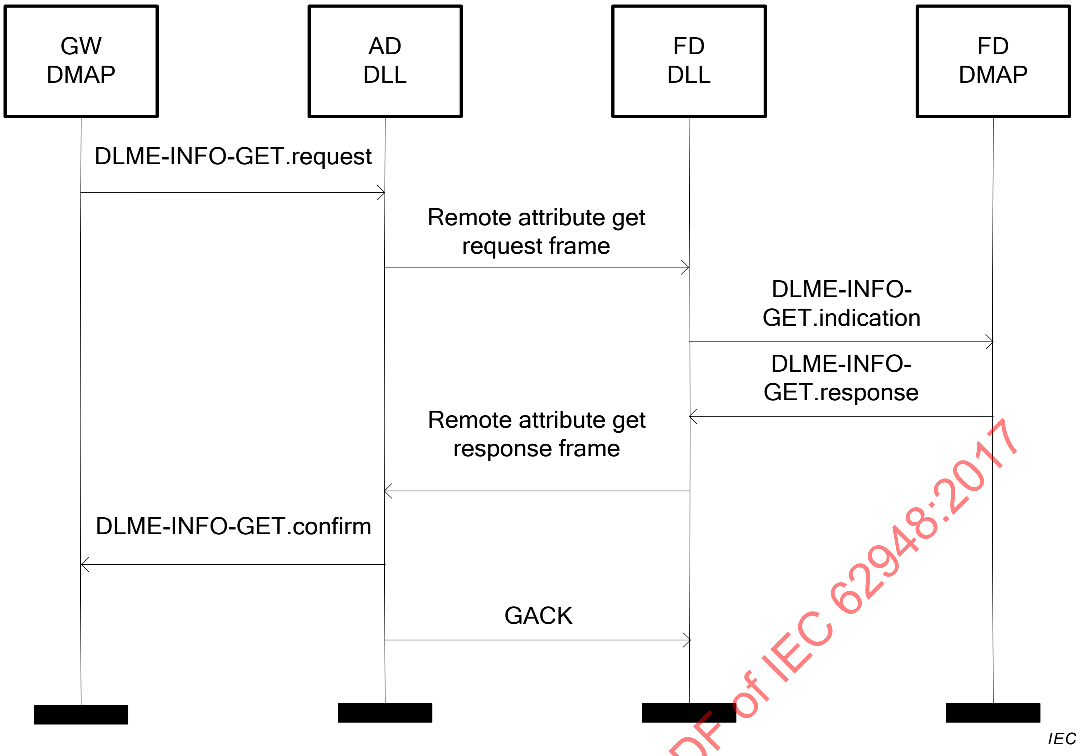


Figure 44 – Remote attribute get process

DMAP of the gateway device invokes DLME-INFO-GET.request primitive in DLL, indicating the access device DLL send Remote attribute get request frame. After receiving Remote attribute get request frame, the field device reports to DMAP by DLME-INFO-GET.indication. The field device DMAP invokes DLME-INFO-GET.response, indicating DLL send Remote attribute get response frame. After receiving Remote attribute get response frame, the access device returns GACK, and reports to DMAP by DLME-INFO-GET.confirm.

8.3.8 Remote attribute set services

8.3.8.1 DLME-INFO-SET.request

DLME-INFO-SET.request is used to remotely modify the MIB attribute values of devices.

```
DLME-INFO-SET.request(
                        Handle,
                        DstAddr,
                        AttributeOption,
                        AttributeID,
                        MemberID,
                        FirstStoreIndex,
                        Count,
                        AttributeValue
                        )
```

Table 59 specifies the parameters for DLME-INFO-SET.request.

Table 59 – DLME-INFO-SET.request parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Assigned handle when invoking the DLME-INFO-SET.request
DstAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bit address of destination device. The setting attributes using 64-bit long address is supported.

AttributeOption	Unsigned8	0 to 255	The operation of remote set attribute:
			0=Add;
			1=Delete;
			2=Update.

AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of an attribute member. The value 255 means that all attribute members shall be written. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The first storage index of multiple records, this value is not valid for the unstructured attributes
Count	Unsigned16	0 to 65 535	Number of records, which is used to set the MIB records, setting all records from FirstStoreIndex if Count = 0
AttributeValue	Octetstring		Value of the attribute to be written. If AttributeOption=1, this value is NULL

8.3.8.2 DLME-INFO-SET.indication

NLME-INFO-SET.indication is used to inform the DMAP of the successful receipt of an attribute set request frame.

```
DLME-INFO-SET.indication(
                        Handle,
                        SrcAddr,
                        AttributeOption,
                        AttributeID,
                        MemberID,
                        FirstStoreIndex,
                        Count,
                        AttributeValue
                        )
```

Table 60 specifies the parameters for DLME-INFO-SET.indication.

Table 60 – DLME-INFO-SET.indication parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from the Remote attribute set request frame.
SrcAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bit address of source device. The setting attributes using 64-bit long address is supported.

AttributeOption	Unsigned8	0 to 255	The operation of remote set attribute:
			0=Add;
			1=Delete;
			2=Update.

AttributeID	Unsigned8	0 to 255	The identifier of an attribute in the MIB
MemberID	Unsigned8	0 to 255	The identifier of an attribute member. The value 255 means that all attribute members shall be written. MemberID is invalid for unstructured attributes.
FirstStoreIndex	Unsigned16	0 to 65 535	The first storage index of multiple records, this value is not valid for the unstructured attributes
Count	Unsigned16	0 to 65 535	Number of records, which is used to set the MIB records, setting all records from FirstStoreIndex if Count = 0
AttributeValue	Octetstring		Value of the attribute to be written. If AttributeOption=1, this value is NULL

8.3.8.3 DLME-INFO-SET.response

DLME-INFO-SET.response is used to respond to DLME-INFO-SET.indication.

```
DLME-INFO-SET.response(
                        Handle,
                        DstAddr,
                        Status
                        )
```

Table 61 specifies the parameters for DLME-INFO-SET.response.

Table 61 – DLME-INFO-SET.response parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from the the DLME-INFO-SET.indication
DstAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bi address of destination device. The setting attributes using 64-bit long address is supported.

Status	Unsigned8	0 to 255	Result of remote set attribute:
			0 = SUCCESS;
			1 = INVALIDATTRIBUTE;
			2 = INVALIDATTRIBUTEMEMBER;
			3 = INVALID RANGE;
			4 = INVALID VALUE;
			5 = READ ONLY;
			others are reserved.

If the operation of setting attributes is successful, the Status shall be SUCCESS; if the MIB does not have the needed attributes, the Status shall be INVALID ATTRIBUTE; if the MIB does not have the needed attribute members, the Status shall be INVALID ATTRIBUTE MEMBER; if AttributeValue is out of valid range, the Status shall be INVALID VALUE. If the MemberID is not 255 and MIB does not have the needed records indexed with FirstStoreIndex, the Status shall be INVALID RANGE. If the access type of the attribute or its members is R, the Status shall be READ_ONLY.

8.3.8.4 DLME-INFO-SET.confirm

DLME-INFO-SET.confirm is used to return the results of DLME-INFO-SET.request.

```
DLME-INFO-SET.confirm(
                        Handle,
                        SrcAddr,
                        Status
                        )
```

Table 62 specifies the parameters for DLME-INFO-SET.confirm.

Table 62 – DLME-INFO-SET.confirm parameters

Parameter	Data type	Valid range	Description
Handle	Unsigned8	0 to 255	Received handle from Remote attribute set response frame
SrcAddr	Unsigned64	0 to (2^{64}-1)	The 8/16/64-bit address of source device. The setting attributes using 64-bit long address is supported.

Status	Unsigned8	0 to 255	Attribute setting results:
			0 = SUCCESS;
			1 = INVALID ATTRIBUTE;
			2 = INVALID ATTRIBUTE MEMBER;
			3 = INVALID RANGE;
			4 = INVALID VALUE;
			5 = READ ONLY
			others are reserved.

8.3.8.5 Remote attribute set process

The process of the remote attribute set is shown in Figure 45.

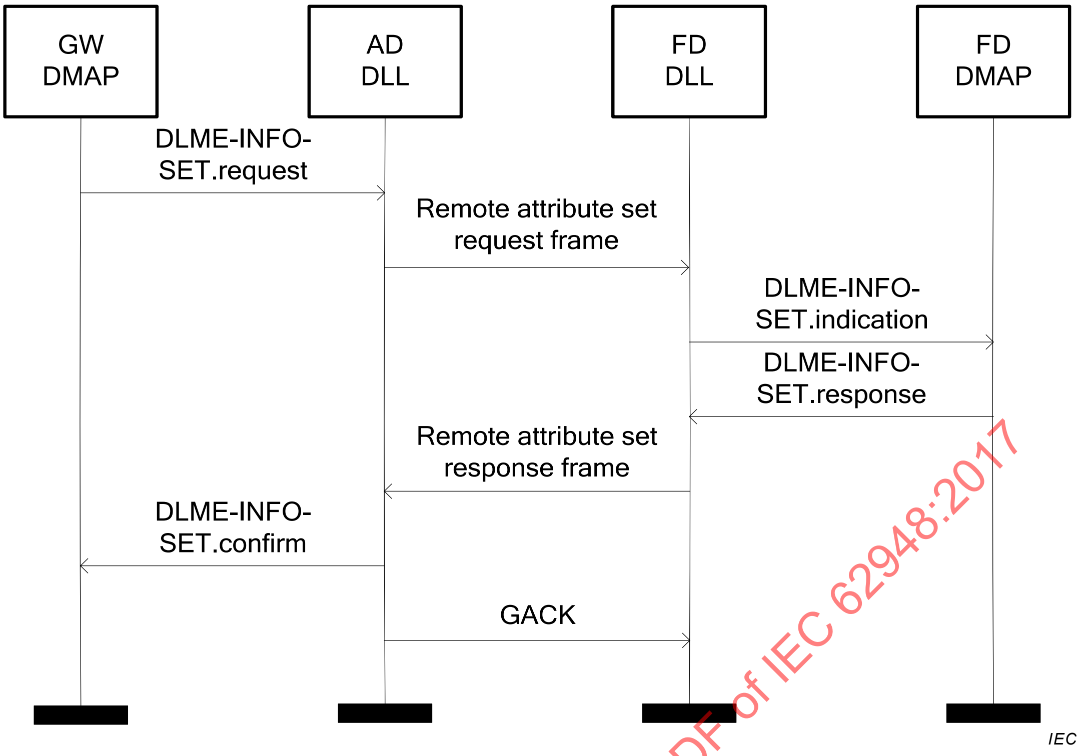


Figure 45 – Remote attribute set process

The DMAP of the gateway device invokes DLME-INFO-SET.request primitive in DLL, indicating the access device DLL send Remote attribute set request frame. After receiving Remote attribute set request frame, the field device reports to DMAP by DLME-INFO-SET.indication. DMAP of a field device invokes DLME-INFO-SET.response, indicating DLL send Remote attribute set response frame. After receiving Remote attribute set response frame, the access device reports to DMAP by DLME-INFO-SET.confirm and returns GACK frame.

8.3.9 Device leaving services

8.3.9.1 DLME-LEAVE.request

DLME-LEAVE.request is used for the gateway device to request a field device leaving WIA-FA network.

```
DLME-LEAVE.request(
                    ShortAddr
                    )
```

Table 63 specifies the parameters for DLME-LEAVE.request.

Table 63 – DLME-LEAVE.request parameters

Parameter	Data type	Valid range	Description
ShortAddr	Unsigned16	0 to 255/0 to 65 535	The short address of the device asked to leave. See DeviceShortAddress in 6.7.1.2.1

8.3.9.2 DLME-LEAVE.indication

DLME-LEAVE.indication is used to notify the field device that it has received a Device Leave request frame.

DLME-LEAVE.indication(
)

8.3.9.3 DLME-LEAVE.response

DLME-LEAVE.response is used to respond to DLME-LEAVE.indication. DMAP invokes DLME-LEAVE.response to generate a Leave response frame.

DLME-LEAVE.response(
)

8.3.9.4 DLME-LEAVE.confirm

DLME-LEAVE.confirm is used to report the result of DLME-LEAVE.request.

DLME-LEAVE.confirm(
Status
)

Table 64 specifies the parameters for DLME-LEAVE.confirm.

Table 64 – DLME-LEAVE.confirm parameters

Parameter	Data type	Valid range	Description

Status	Unsigned8	0 to 255	Result of leave request:
			0 = SUCCESS;
			1 = FAILURE;
			others are reserved.

8.3.9.5 Device leave process

Device leave process is shown in Figure 46.

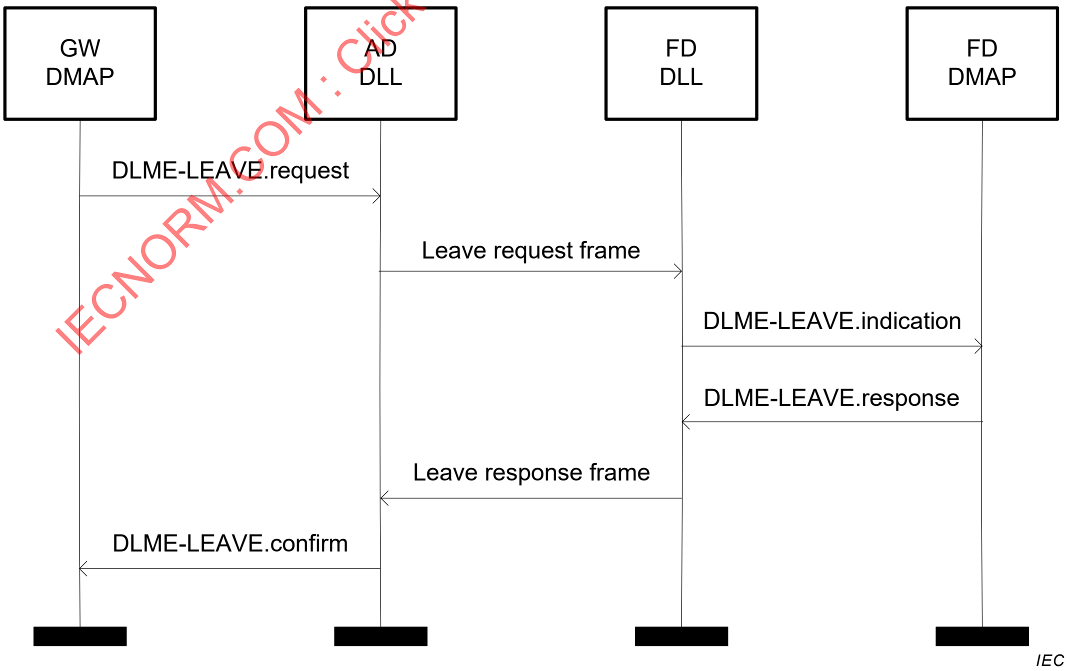


Figure 46 – Device leave process

DMAP of the gateway device invokes DLME-LEAVE.request primitive in DLL, indicating the access device DLL to send Leave request frame. After receiving Leave request frame, the field device reports to DMAP by DLME-LEAVE.indication. DLL of the access device reports sending Leave request frame successfully to the gateway device by DLME-LEAVE.confirm.

8.4 DLL frame formats

8.4.1 General frame format

The DLL general frame format is illustrated in Figure 47.

DLPDU		FCS
DLL Frame Header	DLL Payload	

IEC

Figure 47 – General frame format

The DLL frame is composed of:

- WIA-FA DLL Frame Header, see Figure 48;
- DLL Payload;
- Frame Check Sequence (FCS).

NOTE See Clause 11 for Security.

1 octet	1 octet	1/2/8 octets	2 octets	0/1	0/1	2 octets
Frame Control	Network ID	PeerAddress	Sequence Number	Fragment Count	Fragment Sequence Number	Frame Length

IEC

Figure 48 – DLL frame header

The DLL frame header field has the following subfields:

- Frame Control, see Figure 49;
- Network ID: this field is 1-octet in length, distinguish multiple networks, there shall be only one gateway respond to an unique Network ID;
- Peer Address: this field is 1 or 2 or 8-octet in length;1- or 2- octet length indicates short address, 8-octet length indicates long address;
- Sequence Number: this field is 2-octet in length, indicating frame sequence, increase from 1 to maximum and reset, if two consecutive frames with the same sequence number is received from a device, the second one is identified as duplicate frame;
- Fragment Count: this field is 0 or 1-octet in length, indicating the count of fragments;
- Fragment Sequence Number: this field is 0 or 1-octet in length, indicating the sequence number of fragments;
- Frame length: this field is 2-octet in length, indicating the DLL payload length.

WIA-FA DLL frame control format is shown in Figure 49.

Bit: 0 to 4	Bit: 5	Bit: 6	Bit: 7
Frame Type	Fragment Flag	Preemption Flag	Address Mode

IEC

Figure 49 – DLL frame control format

- Frame Type field is 5-bit in length, defined in Table 65.

Table 65 – Frame type coding

Bit: 0 to 4	Frame Type
00000	Beacon frame
00001	Data frame
00010	Aggregation frame
00011	GACK
00100	NACK
00101	Join request frame
00110	Join response frame
00111	Leave request frame
01000	Leave response frame
01001	Device status report frame
01010	Channel condition report frame
01011	Time synchronization request frame
01100	Time synchronization response frame
01101	Remote attribute get request frame
01110	Remote attribute get response frame
01111	Remote attribute set request frame
10000	Remote attribute set response frame
10001	Key establish request frame (see 11.7.3)
10010	Key establish response frame (see 11.7.4)
10011	Key update request frame (see 11.7.5)
10100	Key update response frame (see 11.7.6)
10101	Security alarm request frame (see 11.7.7)
10110to 11111	Reserved

- Fragment Flag subfield is 1-bit in length, this field specifies whether the frame is a fragment or not. 0 indicates that the frame is not a fragment; 1 indicates that the frame is a frame. If the value of Fragment flag subfield is 0, the subfields of Fragment count and the subfields of Sequence number shall be ignored.
- Preemption Flag subfield is 1-bit in length, this field specifies whether the frame is an occupied frame or not. 0 indicates that the frame is not an occupied frame; 1 indicates that the frame is an occupied frame.
- Address Mode subfield is 1-bit in length, this field specifies the type of source address and destination address in the DLL frame header, see Table 66.

Table 66 – Addressing mode subfields

Bit: 7	Description
0	64-bit long address
1	8/16-bit short address, which is determined by AddressTypeFlag (see 6.7.1.2.1)

8.4.2 Date frame format

The format of the data frame is shown in Figure 50.

7/8/9/10/14/16 octets	Variable
DLL Frame Header	Data Frame Payload

IEC

Figure 50 – DLL Date frame format

- DLL Frame Header: see Figure 48;
- Data Frame Payload: this field is variable in length, indicating the content of data frame, see Payload in 8.4.2.

8.4.3 Aggregation frame format

The format of the data frame is shown in Figure 51.

7/8/9/10/14/16octets	Variable
DLL Frame Header	Aggregation Frame Payload

IEC

Figure 51 – DLL Aggregation frame format

- DLL Frame Header: see Figure 48;
- Aggregation Frame Payload: this field is variable in length, indicating the content of Aggregation frame, see Figure 31 in 8.1.6.

8.4.4 NACK frame format

The format of the data frame is shown in Figure 52.

7/8octets	1 octet	Variable
DLL Frame Header	Retransmitting Devices Count	NACK Information

IEC

Figure 52 – NACK frame format

Variable	2 octets
Retransmission Devices Address List	Relative Timeslot Number

IEC

Figure 53 – NACK information

- DLL Frame Header: see Figure 48, the Peer Address field in the NACK frame header shall be the broadcast address;
- Retransmitting Devices Count: this field is 1-octet in length, indicating the count of field devices requested to retransmit;
- Retransmission Devices List: this field is variable in length, indicating the short address list of field devices requested to retransmit. the length of this list is determined by the value specified in Retransmitting Devices Count field;
- Relative Timeslot Number: the first timeslot used for retransmission. The devices in Retransmission Devices List will take up the timeslot to retransmit in turn according their order in Retransmission Devices List.

8.4.5 GACK frame format

The format of the data frame is shown in Figure 54.

7/8 octets	1 octet	1 octet	Variable
DLL Frame Header	Device Count	GACK Address Mode	GACK List

IEC

Figure 54 – GACK frame format

1/2octets	2 octets
Successful Device Address	Sequence Number

IEC

Figure 55 – GACK_Struct Structure

- DLL Frame Header: see Figure 48, the Peer Address field in the GACK frame header shall be the broadcast address;
- Device Count: this field is 1 octet in length, indicating the count of field devices from which access devices successfully receive frames;
- GACK Address Mode: this field is 1 octet in length, indicating the length of Devices address in GACK information; 0 is 64-bit long address, and 1 is 8/16-bit short address;
- GACK List: this field is variable in length. This field is a list whose type is GACK_Struct Structure. GACK_Struct Structure is defined in Figure 55. And the length of this list is determined by the value specified in Device Count field, GACK Address Mode field and GACK_Struct Structure.
- Successful Device Address: this field is 1/2-octet in length, indicating the short address of a field device from which access devices successfully receive frames; If devices do not transmit successfully, it will retry again in the next shared timeslots;
- Sequence Number: this field is 2-octet in length, indicating the sequence number of a frame successfully received from the corresponding field device.

8.4.6 Beacon frame format

The format of the DLL beacon frame is shown in Figure 56.

7/8octets	2 octets	2 octets	2 octets	2 octets	1 octet	8 octets	Variable
DLL Frame Header	Superframe Length	Timeslot Duration	Beacon Frame Relative Timeslot Number	First Shared Timeslot Number	Shared Timeslot Count (see Figure 57)	Absolute Time Value	BeaconFrame Payload

IEC

Figure 56 – DLL Beacon frame format

Bit: 0 to 3	Bit: 4 to 7
Uplink Shared Timeslot Number	Downlink Timeslot Number

IEC

Figure 57 – Shared timeslot count

- DLL Frame Header: see Figure 48;
- Superframe Length: this field is 2-octet in length, indicating the length of the default superframe, see 8.1.3;
- Timeslot Duration: this field is 2-octet in length, indicating the timeslot length that has be configured, see TimeSlotDuration in 6.7.1.2.1;
- Beacon Frame Relative Timeslot Number: this field is 2-octet in length, indicating the timeslot sending the Beacon frame, see 8.1.3;
- First Shared Timeslot Number: this field is 2-octet in length, indicating the start timeslot of shared timeslots in superframe, see 8.1.3;
- Shared Timeslot Number: this field is 1-octet in length, indicating the total number of shared timeslots, see Figure 57; bits 0 to 3 specify the uplink timeslot number, in which the field devices send the Join request frame to the gateway device, and send the Remote attribute get/set response frame; bits 4 to 7 specify the count of the downlink timeslots that are used by the gateway device to send the Join response frame. The gateway device sends the Join response frame, Remote attribute get/set request, and application configuration in these timeslots, see 8.1.3;
- Absolute Time Value: this field is 8-octet in length, indicating the absolute time sending Beacon frame; see TimeValue in 6.7.1.2.1 for the value of this field;
- Beacon Frame Payload: this field is variable in length, indicating the payload in beacon frame.

8.4.7 Join request frame format

The format of the DLL join request frame is shown in Figure 58.

14 octets	0/8 octets
DLL Frame Header	Security Material

IEC

Figure 58 – DLL join request frame format

- DLL Frame Header: see Figure 48;
- Security Material: this field is 8-octet in length, indicating the authentication information of the new device attempting to join. See SecMaterial in 8.3.4.1 for the value of this field.

8.4.8 Join response frame format

The format of the DLL join response frame is shown in Figure 59.

14 octets	1 octet	1/2 octets
DLL Frame Header	Status	Allocated Short Address

IEC

Figure 59 – DLL join request frame format

- DLL Frame Header: see Figure 48;
- Status: this field is 1-octet in length, indicating the joining result of field device. See Status in 8.3.4.3 for the value of this field;
- Allocated Short Address: this field is 1 or 2-octet in length, indicating the short address allocated by the GW to device attempting to join. See ShortAddr in 8.3.4.3 for the value of this field.

8.4.9 Leave request frame format

The format of the DLL leave request frame is shown in Figure 60.

7/8octets
DLL Frame Header

IEC

Figure 60 – DLL leave request frame format

- DLL Frame Header: see Figure 48.

8.4.10 Leave response frame format

The format of the DLL leave response frame is shown in Figure 61.

7/8octets
DLL FrameHeader

IEC

Figure 61 – DLL leave response frame format

- DLL Frame Header: see Figure 48.

8.4.11 Device status report frame format

The format of the DLL Device status report frame is shown in Figure 62.

7/8octets	Variable
DLL Frame Header	Device Status

IEC

Figure 62 – DLL Device status report frame format

- DLL Frame Header: see Figure 48;
- Device Status: this field is variable in length, indicating the status of device. See PowerSupplyStatus in 8.3.5.1 for the value of this field.

8.4.12 Channel condition report frame format

The format of the DLL Channel condition report frame is shown in Figure 63.

7/8 octets	1 octet	Variable
DLL Frame Header	Count	Channel Condition

IEC

Figure 63 – DLL Channel condition report frame format

- DLL Frame Header: see Figure 48;
- Count: this field is 1 octet in length, indicating the count of Channel condition list;
- Channel Condition: this field is variable in length, indicating the condition of channel. See ChannelConditionInfo in 8.3.6.1 for the value of this field.

8.4.13 Time synchronization request frame format

The format of the DLL time synchronization request frame is shown in Figure 64.

7/8/14octets	8 octets
DLL Frame Header	DeviceTimeValue

IEC

Figure 64 – DLL time synchronization request frame format

- DLL Frame Header: see Figure 48;
- Device Time Value: this field is 8-octet in length, indicating the timestamp when the field device sends time synchronization request frame; see FieldDeviceTimeValue in 8.3.3.3 for the value of this field.

8.4.14 Time synchronization response frame format

The format of the DLL time synchronization response frame is shown in Figure 65.

7/8/14octets	8 octets	8 octets
DLL Frame Header	Device Time Value	ReceiveTimeValue

IEC

Figure 65 – DLL time synchronization response frame format

- DLL Frame Header: see Figure 48;
- DeviceTimeValue: see 8.4.13;
- ReceiveTimeValue: this field is 8-octet in length, indicating the timestamp when the access device receives the Time synchronization request frame; see ReceiveTimeValue in 8.3.3.3 for the value of this field.

8.4.15 Remote attribute get request frame format

The format of the DLL remote attribute get request frame is shown in Figure 66.

7/8octets	1 octet	1 octet	1 octet	0/2 octets	0/2octets
DLL Frame Header	Handle	AttributeID	Attribute Member ID	First Storage Index of Multiple Attribute Records	Number of Records

IEC

Figure 66 – DLL Remote attribute get request frame format

- DLL Frame Header: see Figure 48;
- Handle: the handle of the remote attribute get request command;
- AttributeID: this field is 1-octet in length; see AttributeID in 8.3.7.1 for the value of this field;
- Attribute Member ID: this field is 1-octet in length; see MemberID in 8.3.7.1 for the value of this field;
- First Storage Index of Multiple Attribute records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see FirstStoreIndex in 8.3.7.1 for the value of this field);
- Number of Records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see Count in 8.3.7.1 for the value of this field).

8.4.16 Remote attribute get response frame format

The format of the DLL remote attribute get response frame is shown in Figure 67.

7/8octets	1 octet	1 octet	1 octet	1 octet	0/2 octets	0/2octets	Variable
DLL Frame Header	Handle	Status	AttributeID	Attribute Member ID	First Storage Index of Multiple Attribute records	Number of Records	AttributeValue

IEC

Figure 67 – DLL remote attribute get response frame format

- DLL Frame Header: see Figure 48;
- Handle: the handle of the remote attribute get response command, which is corresponding to the handle of the remote attribute get request command;
- Status: this field is 1-octet in length. See Status in 8.3.7.3 for the value of this field;
- AttributeID: this field is 1-octet in length. See AttributeID in 8.3.7.3 for the value of this field;
- Attribute Member ID: this field is 1-octet in length. See MemberID in 8.3.7.3 for the value of this field;
- First Storage Index of Multiple Attribute records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see FirstStoreIndex in 8.3.7.3 for the value of this field);
- Number of Records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see Count in 8.3.7.3 for the value of this field);
- AttributeValue: this field is variable in length. See AttributeValue in 8.3.7.3 for the value of this field.

8.4.17 Remote attribute set request frame format

The format of the DLL remote attribute set request frame is shown in Figure 68.

7/8octets	1 octet	1 octet	1 octet	1 octet	0/2 octets	0/2octets	Variable
DLL Frame Header	Handle	AttributeOption	AttributeID	Attribute Member ID	First Storage Index of Multiple Attribute records	Number of Records	AttributeValue

IEC

Figure 68 – DLL Remote attribute set request frame format

- DLL Frame Header: see Figure 48;
- Handle: the handle of the remote attribute get request command;
- AttributeOption: this field is 1-octet in length. See AttributeOption in 8.3.8.1 for the value of this field;
- AttributeID: this field is 1-octet in length. See AttributeID in 8.3.8.1 for the value of this field;
- Attribute Member ID: this field is 1-octet in length. See MemberID in 8.3.8.1 for the value of this field;
- First Storage Index of Multiple Attribute records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see FirstStoreIndex in 8.3.8.1 for the value of this field);
- Number of Records: this field is 0/2-octet in length; for unstructured attributes, this field is ignored; for structured attributes, this field is 2 octets in length (see Count in 8.3.8.1 for the value of this field);
- AttributeValue: this field has variable length. See AttributeValue in 8.3.8.1 for the value of this field.

8.4.18 Remote attribute set response frame format

The format of the DLL remote attribute set response frame is shown in Figure 69.

7/8octets	1 octet	1 octet
DLL Frame Header	Handle	Status

IEC

Figure 69 – DLL remote attribute set response frame format

- DLL Frame Header: see Figure 48;
- Handle: the handle of the remote attribute set response command, which is corresponding to the handle of the remote attribute set request command;
- Status: this field is 1-octet in length; see Status in 8.3.8.3 for the value of this field.

8.5 Data link layer state machines

8.5.1 DLL state machine of access device

The state machine of an access device is shown in Figure 70.

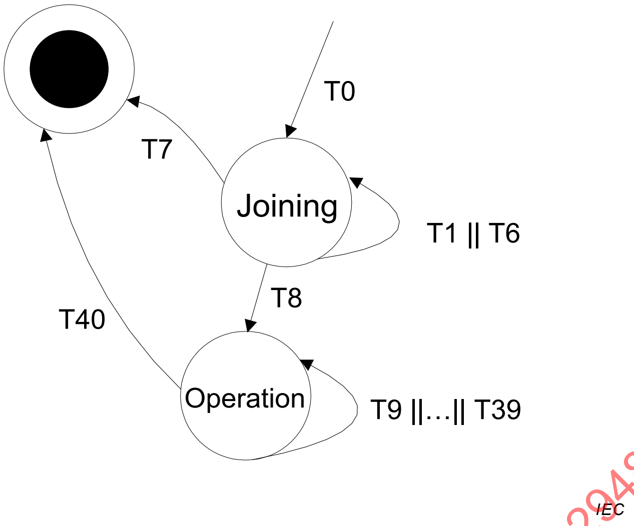


Figure 70 – DLL state machine of access device

The DLL state transition of an access device is listed in Table 67.

Table 67 – DLL state transition of access device

#	Current State	Event or condition => \action	Next state
T0	-	IsDLLInitDone() == TRUE =>	Joining

T1	Joining	PrimitiveType == DLME-JOIN.request
		=>
		SendWiredFrameToGW(PeerAddress:= 0x02,
		FrameName:= AD join request
		);
		Start JoinTimer;
			Joining

T2	Joining	Receive wired frame from GW && PeerAddress == 0x02 && FrameName == AD join response => Cancel JoinTimer; DLME-JOIN.confirm(Status, ShortAddr);	Joining
T3	Joining	JoinTimer time out && No AD join response frame received from GW DLME-JOIN.confirm(Status:= TIMEOUT, ShortAddr);	Joining
T4	Joining	Receive wired frame from GW && PeerAddress == 0x02 && FrameName == GW-AD remote attribute get request => DLME-INFO-GET.indication(Handle, Attribute, MemberID, FirstStroreIndex, Count);	Joining

#	Current State	Event or condition => \action	Next state
T5	Joining	PrimitiveType == DLME-INFO-GET.response => SendWiredFrameToGW(PeerAddress:= 0x02, FrameName:= AD-GW remote attribute get response);	Joining
T6	Joining	(PrimitiveType != DLME-JOIN.request) && (PrimitiveType != DLME-INFO-GET.response) && ((Receive wired frame from GW) && (PeerAddress != 0x02 ||((PeerAddress == 0x02) &&(FrameName != AD join response))&& (FrameName != GW-AD remote attribute get request))) => Ignore;	Joining
T7	Joining	DeviceStatus == End; =>	End
T8	Joining	DeviceStatus == Operation; =>	Operation
T9	Operation	PrimitiveType == DLME-TIME-SYN.response => BuildFrame(FrameType:= TwoWayTimeSynchronizationResponse);	Operation
T10	Operation	PrimitiveType == PHY-DATA.indication && FrameType == TwoWayTimeSynchronizationRequest => DLDE-TIME-SYN.indication(SrcAddr, FieldDeviceTimeValue);	Operation
T11	Operation	Receive wired frame from GW && PeerAddress == 0x02 && FrameName == GW-AD remote attribute get request => DLME-INFO-GET.indication(Handle, Attribute, MemberID, FirstStroreIndex, Count);	Operation
T12	Operation	PrimitiveType == DLME-INFO-GET.response => SendWiredFrameToGW(PeerAddress:= 0x02, FrameName:= AD-GW remote attribute get response);	Operation

#	Current State	Event or condition => \action	Next state
T13	Operation	Receive wired frame from GW && PeerAddress == 0x02 && FrameName == GW-AD remote attribute set request => DLME-INFO-SET.indication(Handle, AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, AttributeVale);	Operation
T14	Operation	PrimitiveType == DLME-INFO-SET.response => SendWiredFrameToGW(PeerAddress:= 0x02, FrameName:= AD-GW remote attribute set response);	Operation
T15	Operation	PrimitiveType == DLME-DEVICE-STATUS.request => SendWiredFrameToGW(PeerAddress:= 0x02, FrameName:= AD-GW device status report);	Operation
T16	Operation	Receive wired frame from GW && PeerAddress == 0x02 && FrameName ==NACK => HandleNACK();	Operation
T17	Operation	Receive wired frame from GW && PeerAddress == 0x02 && FrameName ==GACK => HandleGACK();	Operation
T18	Operation	PrimitiveType == DLME-CHANNEL-STATUS.request => SendWiredFrameToGW(PeerAddress:= 0x02, FrameName:= AD-GW channel condition report);	Operation
T19	Operation	Receive wired frame from GW && PeerAddress == 0x02 && FrameName == GW-AD device leave request => DLME-LEAVE.indication();	Operation
T20	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==NACK => BuildFrame(FrameType:= NACK);	Operation
T21	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GACK => BuildFrame(FrameType:= GACK);	Operation

#	Current State	Event or condition => \action	Next state
T22	Operation	PrimitiveType == PHY-DATA.indication && FrameType == JoinRequest Command => SendWiredFrameToGW(FrameName:= AD-GW join request);	Operation
T23	Operation	PrimitiveType == PHY-DATA.indication && FrameType == Data => SendWiredFrameToGW(FrameName:= AD-GW data);	Operation
T24	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GW-AD data => BuildFrame(FrameType:= Data);	Operation
T25	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GW-AD join response => BuildFrame(FrameType:= JoinResponse);	Operation
T26	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GW-AD remote attribute set request => BuildFrame(FrameType:= RemoteAttributeSetRequest);	Operation
T27	Operation	PrimitiveType == PHY-DATA.indication && FrameType == RemoteAttributeSetResponse => SendWiredFrameToGW(FrameName:= AD-GW remote attribute set response);	Operation
T28	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GW-AD remote attribute get request => BuildFrame(FrameType:= RemoteAttributeGetRequest);	Operation
T29	Operation	PrimitiveType == PHY-DATA.indication && FrameType == RemoteAttributeGetResponse => SendWiredFrameToGW(FrameName:= AD-GW remote attribute get response);	Operation
T30	Operation	PrimitiveType == PHY-DATA.indication && FrameType == DeviceStatusReport => SendWiredFrameToGW(FrameName:= AD-GW device status report);	Operation

#	Current State	Event or condition => \action	Next state
T31	Operation	PrimitiveType == PHY-DATA.indication && FrameType == ChannelConditionReport => SendWiredFrameToGW(FrameName:= AD-GW channel condition report);	Operation
T32	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName ==GW-AD device leave request => BuildFrame(FrameType:= LeaveRequest);	Operation
T33	Operation	PrimitiveType == PHY-DATA.indication && FrameType == LeaveResponse => SendWiredFrameToGW(FrameName:= AD-GW device leave response);	Operation
T34	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName == GW-AD key establish request => BuildFrame(FrameType:= KeyEstablishRequest);	Operation
T35	Operation	PrimitiveType == PHY-DATA.indication && FrameType == KeyEstablishResponse => SendWiredFrameToGW(FrameName:= AD-GW key establish response);	Operation
T36	Operation	Receive wired frame from GW && PeerAddress != 0x02 && FrameName == GW-AD key update request => BuildFrame(FrameType:= KeyUpdateRequest);	Operation
T37	Operation	PrimitiveType == PHY-DATA.indication && FrameType == KeyUpdateResponse => SendWiredFrameToGW(FrameName:= AD-GW key update response);	Operation
T38	Operation	PrimitiveType == PHY-DATA.indication && FrameType == SecAlarmRequest => SendWiredFrameToGW(FrameName:= AD-GW security alarm);	Operation

#	Current State	Event or condition => \action	Next state
T39	Operation	(PrimitiveType != DLME-TIME-SYN.response) && (PrimitiveType != DLME-INFO-GET.response) && (PrimitiveType != DLME-INFO-SET.response) && (PrimitiveType != DLME-DEVICE-STATUS.request) && (PrimitiveType != DLME-CHANNEL-STATUS.request) && ((PrimitiveType == PHY-DATA.indication) && (FrameType != TwoWayTimeSynchronizationRequest) && (FrameType != JoinRequest Command) && (FrameType != Data) && (FrameType != RemoteAttributeGetResponse) && (FrameType != RemoteAttributeSetResponse) && (FrameType != DeviceStatusReport) && (FrameType != ChannelConditionReport) && (FrameType != LeaveResponse) && (FrameType != KeyEstablishResponse) && (FrameType != KeyUpdateResponse) && (FrameType != SecAlarmRequest)) && ((Receive wired frame from GW) && (((PeerAddress != 0x02) &&(FrameName !=NACK) &&(FrameName !=GACK) &&(FrameName !=GW-AD data) && (FrameName != GW-AD join response) && (FrameName != GW-AD remote attribute set request) && (FrameName != GW-AD remote attribute get request) && (FrameName != GW-AD device leave request) && (FrameName != GW-AD key establish request) && (FrameName != GW-AD key update request)) || ((PeerAddress == 0x02) && (FrameName != GW-AD remote attribute get request) && (FrameName != GW-AD remote attribute set request) && (FrameName != NACK) && (FrameName != GACK) && (FrameName != GW-AD device leave request))) => Ignore;	Operation
T40	Operation	DeviceStatus == End; =>	End

– Joining state

After the DLL of access device finishes initialization, the state machine shall enter to Joining state. In this state, the following events and responding actions can occur:

- If DLME-JOIN.request is invoked by DMAP, DLL shall generate an AD join request frame, send it to GW using the wired link and start a JoinTimer.

- If an AD join response frame is received from GW through the wired link and PeerAddress is equal to 0x02, DLL shall cancel JoinTimer and invoke DLME-JOIN.confirm.
- If JoinTimer expires and no AD join response frame is received from GW through the wired link during this period, DLL shall invoke DLME-JOIN.confirm with status is equal to TIMEOUT.
- If a GW-AD remote attribute get request frame is received from GW through the wired link, DLL shall invoke DLME-INFO-GET.indication.
- If DLME-INFO-GET.response is invoked by DMAP, DLL shall generate an AD-GW remote attribute get response frame and send it to GW using the wired link.

– Operation state

When DeviceStatus is equal to Operation, the state machine shall enter to Operation state. In this state, the following events and responding actions can occur:

- If PHY-DATA.indication is invoked by PHY and a two way synchronization request frame is received, DLL shall invoke DLME-TIME-SYN.indication.
- If DLME-TIME-SYN.response is invoked by DMAP, DLL shall generate the two way synchronization response frame and post it to the transmission queue.
- If a GW-AD remote attribute get request frame is received from GW through the wired link, DLL shall invoke DLME-INFO-GET.indication.
- If DLME-INFO-GET.response is invoked by DMAP, DLL shall generate an AD-GW remote attribute get response frame and send it to GW using the wired link.
- If a GW-AD remote attribute set request frame is received from GW through the wired link, DLL shall invoke DLME-INFO-SET.indication.
- If DLME-INFO-SET.response is invoked by DMAP, DLL shall generate an AD-GW remote attribute set response frame and send it to GW using the wired link.
- If a NACK frame is received from GW through the wired link and PeerAddress is equal to 0x02, DLL shall handle NACK frame.
- If a GACK frame is received from GW through the wired link and PeerAddress is equal to 0x02, DLL shall handle GACK frame.
- If a NACK frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a NACK frame and post it to the transmission queue.
- If a GACK frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a GACK frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a data frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD device leave request frame is received from GW through the wired link and PeerAddress is equal to 0x02, DLL shall invoke DLME-LEAVE.indication.
- If DLME-DEVICE-STATUS.request is invoked by DMAP, DLL shall generate a device status report frame and and send it to GW using the wired link.
- If DLME-CHANNEL-STATUS.request is invoked by DMAP, DLL shall generate a channel status report frame and send it to GW using the wired link.
- If PHY-DATA.indication is invoked by PHY and a join request frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD data frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a data frame and post it to the transmission queue.
- If a GW-AD join response frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a join response frame and post it to the transmission queue.

- If a GW-AD remote attribute set request frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a remote attribute set request frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a remote attribute set response frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD remote attribute get request frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a remote attribute get request frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a remote attribute get response frame is received, DLL shall send it to GW using the wired link.
- If PHY-DATA.indication is invoked by PHY and a device status report frame is received, DLL shall send it to GW using the wired link.
- If PHY-DATA.indication is invoked by PHY and a channel status report frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD device leave request frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a device leave request frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a device leave response frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD key establish request frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a key establish request frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a key establish response frame is received, DLL shall send it to GW using the wired link.
- If a GW-AD key update request frame is received from GW through the wired link and PeerAddress is not equal to 0x02, DLL shall generate a key update request frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a key update response frame is received, DLL shall send it to GW using the wired link.
- If PHY-DATA.indication is invoked by PHY and a security alarm request frame is received, DLL shall send it to GW using the wired link.

8.5.2 DLL state machine of field device

The state machine of a field device is shown in Figure 71.

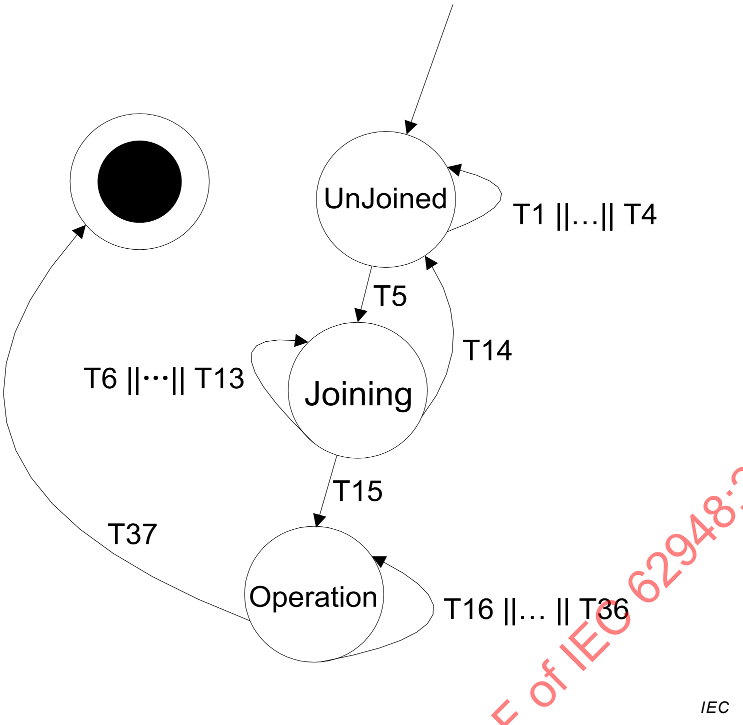


Figure 71 – DLL state machine of field device

The DLL state transition of a field device is listed in Table 68.

Table 68 – DLL state transition of field device

#	Current State	Event or condition => \action	Next state

T1	Unjoined	PrimitiveType == DLME-DISCOVERY.request
		=>
		Start ScanTimer;
		ScanChanels(Channels);
			Unjoined

T2	Unjoined	ScanTimer time out &&PrimitiveType == PHY-DATA.indication &&FrameType == Beacon => DoSynchronization(); DLME-DISCVOERY.confirm(Status:= SUCCESS);	Unjoined
T3	Unjoined	ScanTimer time out && No Beacon frame received => DLME-DISCVOERY.confirm(Status:= TIMEOUT);	Unjoined
T4	Unjoined	PrimitiveType != DLME-DISCOVERY.request && ((PrimitiveType == PHY-DATA.indication) && (FrameType != Beacon)) => Ignore;	Unjoined
T5	Unjoined	DeviceStatus == Joining; =>	Joining

#	Current State	Event or condition => \action	Next state

T6	Joining	PrimitiveType == DLME-JOIN.request
		=>
		BuildFrame(FrameType:= JoinRequest);
		Start JoinTimer;
			Joining

T7	Joining	PrimitiveType == PHY-DATA.indication && FrameType == JoinResponse => Cancel JoinTimer; DLME-JOIN.confirm(Status, ShortAddr);	Joining
T8	Joining	JoinTimer time out && No Join response frame received DLME-JOIN.confirm(Status:= TIMEOUT, ShortAddr);	Joining
T9	Joining	PrimitiveType == PHY-DATA.indication && FrameType == GACK => HandleGACK();	Joining
T10	Joining	PrimitiveType == PHY-DATA.indication &&FrameType == Beacon => DoSynchronization();	Joining
T11	Joining	PrimitiveType == PHY-DATA.indication && FrameType == RemoteAttributeGetRequest => DLME-INFO-GET.indication(Handle, Attribute, MemberID, FirstStroreIndex, Count);	Joining
T12	Joining	PrimitiveType == DLME-INFO-GET.response => BuildFrame(FrameType:= RemoteAttributeGetResponse);	Joining
T13	Joining	((PrimitiveType == PHY-DATA.indication) &&((FrameType != Beacon) && (FrameType != RemoteAttributeGetRequest) && (FrameType != JoinResponse) && (FrameType != GACK)) && (PrimitiveType != DLME-DISCOVERY.request) && (PrimitiveType != DLME-JOIN.request) => Ignore;	Joining
T14	Joining	DeviceStatus == Unjoined; =>	Unjoined
T15	Joining	DeviceStatus == Operation =>	Operation

#	Current State	Event or condition => \action	Next state
T16	Operation	PrimitiveType == PHY-DATA.indication && FrameType == RemoteAttributeSetRequest => DLME-INFO-SET.indication(Handle, AttributeOption, AttributeID, MemberID, FirstStoreIndex, Count, AttributeVale);	Operation
T17	Operation	PrimitiveType == DLME-INFO-SET.response => BuildFrame(FrameType:= RemoteAttributeSetResponse);	Operation
T18	Operation	PrimitiveType == PHY-DATA.indication && FrameType == RemoteAttributeGetRequest => DLME-INFO-GET.indication(Handle, Attribute, MemberID, FirstStroreIndex, Count);	Operation
T19	Operation	PrimitiveType == DLME-INFO-GET.response => BuildFrame(FrameType:= RemoteAttributeGetResponse);	Operation
T20	Operation	PrimitiveType == PHY-DATA.indication && FrameType == GACK => HandleGACK();	Operation
T21	Operation	PrimitiveType == PHY-DATA.indication && FrameType == NACK => HandleNACK();	Operation
T22	Operation	PrimitiveType == PHY-DATA.indication &&FrameType == Beacon => DoSynchronization();	Operation
T23	Operation	PrimitiveType == PHY-DATA.indication && FrameType ==Data => DLDE-DATA.indication(SrcAddr, DataType, PayloadLength, Payload);	Operation
T24	Operation	PrimitiveType == PHY-DATA.indication && FrameType == LeaveRequest => DLME-LEAVE.indication();	Operation
T25	Operation	PrimitiveType == DLME-DEVICE-STATUS.request => BuildFrame(FrameType:= DeviceStatusReport);	Operation

T26	Operation	PrimitiveType == DLME-TIME-SYN.request
		=>
		BuildFrame(FrameType:= TwoWayTimeSynchronizationRequest);
		Start TwoWayOverTimer;
			Operation

#	Current State	Event or condition => \action	Next state
T27	Operation	PrimitiveType == DLME-CHANNEL-STATUS.request => BuildFrame(FrameType:= ChannelStatusReport);	Operation
T28	Operation	PrimitiveType == PHY-DATA.indication && FrameType == TwoWayTimeSynchronizationResponse => DLDE-TIME-SYN.confirm(Status:= SUCCESS, FieldDeviceTimeValue, ReceiveTimeValue);	Operation
T29	Operation	TwoWayOverTimer time out && No TwoWayTimeSynchronizationResponse frame received => DLDE-TIME-SYN.confirm(Status:= OVERTIME, FieldDeviceTimeValue, ReceiveTimeValue);	Operation
T30	Operation	PrimitiveType == PHY-DATA.indication && FrameType == KeyEstablishRequest => KEY-ESTABLISH.indication(KeyMaterial);	Operation
T31	Operation	PrimitiveType == KEY-ESTABLISH.response => BuildFrame(FrameType:= KeyEstablishResponse);	Operation
T32	Operation	PrimitiveType == PHY-DATA.indication && FrameType == KeyUpdateRequest => KEY-UPDATE.indication(KeyMaterial);	Operation
T33	Operation	PrimitiveType == KEY- UPDATE.response => BuildFrame(FrameType:= KeyUpdateResponse);	Operation
T34	Operation	PrimitiveType == SEC-ALARM.request => BuildFrame(FrameType:= SecAlarmRequest);	Operation
T35	Operation	PrimitiveType == DLME- LEAVE.response => BuildFrame(FrameType:= LEAVEResponse);	Operation

#	Current State	Event or condition => \action	Next state
T36	Operation	((PrimitiveType == PHY-DATA.indication) && ((FrameType != RemoteAttributeSetRequest) && (FrameType != RemoteAttributeGetRequest) && (FrameType != GACK) &&(FrameType != NACK) &&(FrameType!= Beacon) &&(FrameType !=Data) &&(FrameType != LeaveRequest) && (FrameType != TwoWayTimeSynchronizationResponse) && (FrameType != KeyEstablishRequest)) && (PrimitiveType != DLME-INFO-SET.response) && (PrimitiveType!= DLME-INFO-GET.response) && (PrimitiveType != DLME-DEVICE-STATUS.request) && (PrimitiveType != DLME-TIME-SYN.request) && (PrimitiveType != DLME-CHANNEL-STATUS.request) && (PrimitiveType != KEY-ESTABLISH.response) => Ignore;	Operation
T37	Operation	DeviceStruct.Status:= UnJoined; =>	End

– Unjoined state

After the DLL of field device finishes initialization, the state machine shall enter to Unjoined state. If DLME-DISCOVERY.request is invoked by the DMAP, the DLL shall start ScanTimer and scan channels. When the ScanTimer expires, if Beacon frames are received, DLL shall do synchronization and invoke DLME-DISCVOERY.confirm with status being equal to SUCCESS; otherwise if no Beacon frames are received, DLL shall invoke DLME-DISCVOERY.confirm with status being equal to TIMEOUT.

– Joining state

When DeviceStatus is equal to Joining, the state machine shall enter to Joining state. In Joining state, the following events and responding actions can occur:

- If DLME-JOIN.request is invoked by DMAP, DLL shall generate a Join request frame, post it to the transmission queue, and start a JoinTimer.
- If PHY-DATA.indication is invoked by PHY and a Join response frame is received, DLL shall cancel JoinTimer and invoke DLME-JOIN.confirm
- If JoinTimer expires and no Join response frame is received during this period, DLL shall invoke DLME-JOIN.confirm with status is equal to TIMEOUT.
- If PHY-DATA.indication is invoked by PHY and a GACK frame is received, DLL shall handle GACK.
- If PHY-DATA.indication is invoked by PHY and a Beacon frame is received, DLL shall do synchronization.
- If PHY-DATA.indication is invoked by PHY and a Remote attribute get request frame is received, DLL shall invoke DLME-INFO-GET.indication.
- If DLME-INFO-GET.response is invoked by DMAP, DLL shall generate a Remote attribute get response frame, and post it to the transmission queue.

– Operation state

When DeviceStatus is equal to Operation, the state machine shall enter to Operation state. In this state, the following events and responding actions can occur:

- If PHY-DATA.indication is invoked by PHY and a Remote attribute get request frame is received, DLL shall invoke DLME-INFO-GET.indication.
- If DLME-INFO-GET.response is invoked by DMAP, DLL shall generate a Remote attribute get response frame, and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a Remote attribute set request frame is received, DLL shall invoke DLME-INFO-SET.indication.
- If DLME-INFO-SET.response is invoked by DMAP, DLL shall generate a Remote attribute set response frame, and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a GACK frame is received, DLL shall handle GACK.
- If PHY-DATA.indication is invoked by PHY and a NACK frame is received, DLL shall handle NACK.
- If PHY-DATA.indication is invoked by PHY and a Beacon frame is received, DLL shall do synchronization.
- If PHY-DATA.indication is invoked by PHY and a Data frame is received, DLL shall invoke DLDE-DATA.indication.
- If PHY-DATA.indication is invoked by PHY and a Leave request frame is received, DLL shall invoke DLME-LEAVE.indication.
- If DLME-DEVICE-STATUS.request is invoked by DMAP, DLL shall generate a Device status report frame and post it to the transmission queue.
- If DLME-TIME-SYN.request is invoked by DMAP, DLL shall generate a two way Time synchronization request frame, post it to the transmission queue, and start a TwoWayOverTimer.
- If DLME-CHANNEL-STATUS.request is invoked by DMAP, DLL shall generate a Channel condition report frame and post it to the transmission queue.
- If PHY-DATA.indication is invoked by PHY and a two way Time synchronization response frame is received, DLL shall invoke DLME-TIME-SYN.confirm.
- If the TwoWayOverTimer expires and no two-way Time synchronization response frame is received, DLL shall invoke DLME-TIME-SYN.confirm with status is equal to OVERTIME.
- If PHY-DATA.indication is invoked by PHY and a Key establish request frame is received, DLL shall invoke KEY-ESTABLISH.indication.
- If PHY-DATA.indication is invoked by PHY and a Key update request frame is received, DLL shall invoke KEY-UPDATE.indication.
- If KEY-ESTABLISH.response is invoked, DLL shall generate a Key establish response frame and post it to the transmission queue.
- If KEY-UPDATE.response is invoked, DLL shall generate a Key update response frame and post it to the transmission queue.
- If SEC-ALARM.response is invoked, DLL shall generate a Security alarm request frame and post it to the transmission queue.
- If DLME-LEAVE.response is invoked, DLL shall generate a Leave response frame and post it to the transmission queue.

8.5.3 Functions used in DLL state machines

The functions used in DLL state machines are listed in Table 69.

Table 69 – Functions used in DLL state machines

Function	Input	Output	Description
BuildFrame()	FrameType		Generating frame and putting into queue
DoSynchronization()			Performing time synchronization
ScanChanels()	Channels		Scanning channels

9 Wired specifications between GW and AD

9.1 Overview

The wired services and frame formats between the gateway device and access devices are defined in this document. The wired communication methods are out of the scope of this document.

9.2 Join process of access device

An access device shall not scan channels when it attempts to join a WIA-FA network.

The join process of an access device is as follows.

- The access device shall send an AD join request frame to the gateway device (see 9.3);
- The NM residing on the gateway device shall return an AD join response frame to the access device (see 9.3);
- Subsequently, the NM allocates communication resources (superframes, links, etc.) for the access device by using GW-AD remote attribute set request frame (see 9.3).

See 9.3 for the service identifiers.

The 8/16-bit short address of an access device (see 6.3) is unique in a WIA-FA network.

9.3 Frame formats between GW and AD

The general format of frames between the gateway device and an access device is shown in Figure 72.

1 octet	0/1 octet	0/8 octet(s)	1/2/8 octet(s)	2 octets	2 octets	Variable length
Frame Type	AdID	AD Long Address	Peer Address	Sequence Number	Length	Payload

IEC

Figure 72 – General frame format between GW and AD

Each field in Figure 72 is defined as follows:

- Frame Type: this field is 1-octet in length; Frame Type is used to identify different frames (see Table 70) between the gateway device and an access device;
- AdID: this field is 1-octet in length; AdID is used to identify ADs (see Table 21). AdID is invalid if Frame type is 0 or 1;
- AD Long Address: this field is 0 octet or 8 octets) in length; if the Frame Type is 0 or 1, AD Long Address is the EUI-64 of an access device; otherwise, AD Long Address is invalid;

- Peer Address: this field is 1 or 2 or 8-octet in length; if frames are locally generated by AD, this field value is the short address of AD (0x02); 1 or 2-octet in length indicate the short address of a field device; 8-octet length indicates the long address of a field device;
- Sequence Number: this field is 2-octet in length; this field specifies frame sequence; if the frame is from AD to GW, this field is filled with the sequence number in DLL frame header of a field device; if the frame is from GW to AD, this field is ignored;
- Length: this field is 2-octets in length;
- Payload: this field is variable in length. Payload is wired data between the gateway device and an access device. See Table 70.

Frames transmitted between GW and AD via the wired links are defined in Table 70.

Table 70 – Frames between GW and AD

Frame type	Frame name	Frame fields	Transmission direction
0	AD join request	See Table 71	AD→GW
1	AD join response	See Table 72	GW→AD
2	GW-AD GACK	See Table 73 and Table 74	GW→AD
3	GW-AD NACK	See Table 75	GW→AD
4	GW-AD data	Parameters defined in DLDE-DATA.request (see 8.2.2)	GW→AD
5	AD-GW data	Parameters defined in DLDE-DATA.indication (see 8.2.3)	AD→GW
6	AD-GW join request	Parameters defined in DLME-JOIN.indicaiton (see 8.3.4.2)	AD→GW
7	GW-AD join response	Parameters defined in DLME-JOIN.response (see 8.3.4.3)	GW→AD
8	AD-GW device status report	Parameters defined in DLME-DEVICE-STATUS.indication (see 8.3.5.2)	AD→GW
9	AD-GW channel condition report	Parameters defined in DLME-CHANNEL-CONDITION.indication (see 8.3.6.2)	AD→GW
10	GW-AD remote attribute get request	Parameters defined in DLME-INFO-GET.request (see 8.3.7.1)	GW→AD
11	AD-GW remote attribute get response	Parameters defined in DLME-INFO-GET.confirm (see 8.3.7.4)	AD→GW
12	GW-AD remote attribute set request	Parameters defined in DLME-INFO-SET.request (see 8.3.8.1)	GW→AD
13	AD-GW remote attribute set response	Parameters defined in DLME-INFO-SET.confirm (see 8.3.8.4)	AD→GW
14	GW-AD device leave request	Parameters defined in DLME-LEAVE.request (see 8.3.9.1)	GW→AD
15	AD-GW device leave response	Parameters defined in DLME-LEAVE.confirm (see 8.3.9.4)	AD→GW
16	GW-AD key establish request	Parameters defined in KEY-ESTABLISH.request (see 11.2.2.1)	GW→AD
17	AD-GW key establish response	Parameters defined in KEY-ESTABLISH.confirm (see 11.2.2.4)	AD→GW
18	GW-AD key update request	Parameters defined in KEY-UPDATE.request (see 11.2.3.1)	GW→AD
19	AD-GW key update response	Parameters defined in KEY-UPDATE.confirm (see 11.2.3.4)	AD→GW
20	AD-GW security alarm	Parameters defined in SEC-ALARM.request (see 11.2.4.1)	AD → GW

Frame type	Frame name	Frame fields	Transmission direction
21 to 255	Reserved		

The frames identified by Frame type 0 and 1 are used for ADs joining WIA-FA network. The fields of an AD join request frame and an AD join response frame are shown in Table 71 and Table 72, respectively.

Table 71 – Payload of AD join request frame

Parameter	Data type	Valid range	Description
NetworkID	Unsigned8	0 to 255	Network identifier, used for multiple networks coexisting (see NetworkID in 6.7.1.2.1)
PhyAddr	Unsigned64	0 to (2^{64}-1)	Long address of the new access device attempting to join (see LongAddress in 6.7.1.2.1)
SecMaterial	Unsigned64	0 to (2^{64}-1)	Device security material for authentication (see 11.3). If the SecLevel is 0, this parameter is ignored.

Table 72 – Payload of AD join response frame

Parameter	Data type	Valid range	Description
Status	Unsigned8	0 to 255	Result of join request: 0 = SUCCESS; 1 = NetworkID mismatched (The NetworkID of the device attempting to join network is different from the NetworkID of the gateway device); 2 = Authentication fail (SecMaterial in the join request frame is different from the HMAC calculation result of SM, see 11.3); 3 = Network overload (no more AdID is available); 4 = Join timeout (When the access device does not receive AD join response frame within length of 256 default superframes, the join process is failed); others are reserved.
AdID	Unsigned	0 to 255	Identifier of an access device allocated by the gateway device (see AdID in 6.7.1.2.2). AdID is valid if Status is SUCCESS
ADAddr	Unsigned8/ Unsigned16	0 to 255/ 0 to 65 535	The 8/16-bit short address of an access device allocated by the gateway device

The frames identified by Frame type 2 and 3 are used by GW to request ADs sending GACKs or NACKs. The related fields of an AD GACK frame and an AD NACK frame are shown in Table 73, Table 74 and Table 75 respectively.

Table 73 – Payload of GW requesting AD to send GACK

Field name	Data type	Valid range	Description
Device count	Unsigned8	0 to 255	Count of field devices received frames by any access device
GACK information	GACKInfo_Struct List		Information of DeviceCount field devices, indicating by GACKInfo_Struct (see Table 74)

Table 74 – Definition of GACKInfo_Struct

Member name	Data type	Valid range	Description
Destination address	Unsigned8/ Unsigned16/ Unsigned64	0 to 255/ 0 to 65 535/ 0 to $(2^{64}-1)$	Being short address or long address of a field device
Sequence number	Unsigned16	0 to 65 535	Sequence number of a received frame

Table 75 – Payload of GW requesting AD to send NACK

Field name	Data type	Valid range	Description
RetryDeviceCount	Unsigned8	0 to 255	Count of field devices required retransmission
DstAddressList	Unsigned8/ Unsigned16		Destination address. See DeviceShortAddress in 6.7.1.2.2.

If ADs are used for transmitting frames between field devices and the gateway device, wired frames identified Frame type 2 to 21 are used, in which Peer address is filled with a field device address and Sequence number is filled with the sequence number in DLL frame header of a field device; if AD transmits frames generated by itself, Peer address is filled with the short address of AD and Sequence number is filled with the sequence number maintained by AD.

9.4 Communication based on multiple access device

The gateway device would receive redundant messages from multiple access devices. The following specifications process the redundant messages.

a) The wired frame formats between the gateway device and access devices define the Sequence Number field, which is used to distinguish the frame, See 9.3 for detail.

b) The sequence number field in these frames is used to identity whether the frame is duplicated. The redundant frames have the same sequence number. If the frame is duplicated, it will be dropped.

10 Application Layer

10.1 Overview

WIA-FA application layer (AL) supports distributed applications for users. AL comprises of UAPs and ASL. Each UAP is composed of one or more UAOs that interact with industrial processes. DMAP is a special UAP (see 6.2.1). ASL defines communication services among UAPs on different devices.

10.2 AL protocol stack

Figure 73 shows AL within the protocol architecture of WIA-FA, and the grey parts denote AL-related parts.

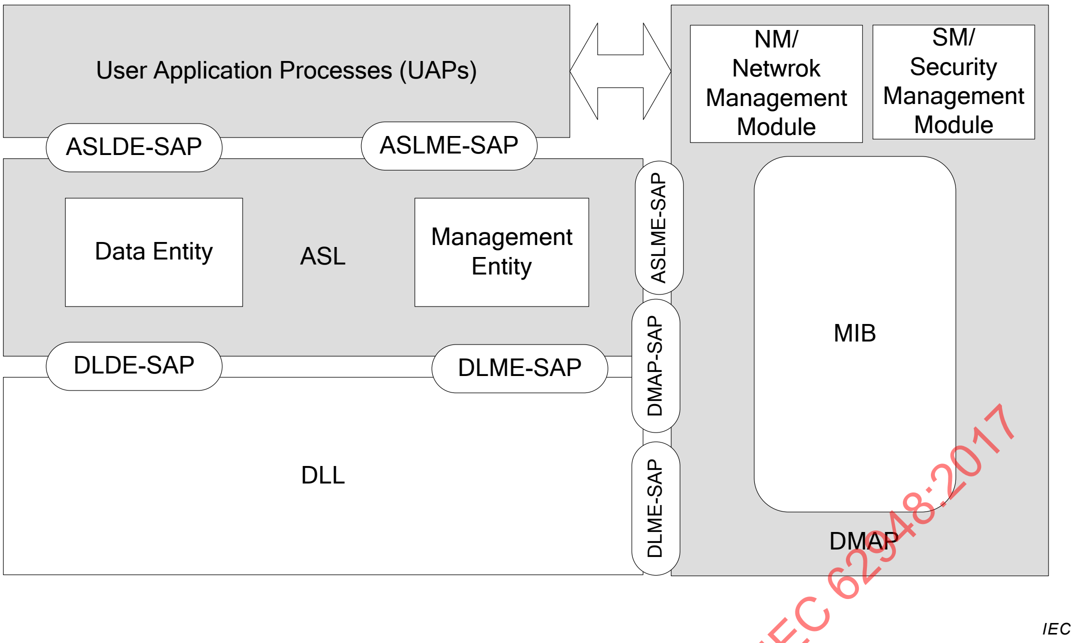


Figure 73 – AL within the protocol architecture of WIA-FA

The functions of WIA-FA ASL are performed by ASL data entity (ASLDE) and ASL management entity (ASLME). ASLDE shall perform ASL data transmission functions (see 10.7.2), and the data service interfaces are provided by ASLDE-SAPs. ASLME shall perform application management functions (see 10.3.2), and the management service interfaces are provided by ASLME-SAPs.

10.3 AL functions

10.3.1 Data functions

WIA-FA supports three types of application data transferred between the gateway device and field devices, including:

- periodical input data (e.g., sensor measurement values, switch status, actuator feedback values) and periodical output data (e.g., actuator setpoints, switch set values), with data priority RT1;
- aperiodical requests and responses of attribute read and write accesses as well as alarm acknowledgements, with data priority NRT;
- aperiodical alarm reports with data priority RT2; urgent commands such as start and stop commands with data priority RT0.

WIA-FA defines corresponding application services (see 10.6) and virtual communication relationships (see 10.5.5.4) to support the usages and transmissions for the different application data.

10.3.2 Management functions

UAP can get and set MIB attributes locally or remotely. If locally getting or setting the device's own MIB attributes, UAP shall directly use DMAP attribute get/set services (see 6.7.2.2 and 6.7.2.3). If remotely getting or setting MIB attributes from other devices, UAP may operate via DMAP-SAP, and then DMAP shall use DLL Remote Attribute Get/Set services (see 8.3.7 and 8.3.8). Alternately, AL may also use ASL Read/Write services (see 10.6.2 and 10.6.3) to access MIB attributes.

In addition, if a UAP on a field device receives Start or Stop command from the gateway device, it shall activate or deactivate its associated P/S VCR(s).

10.3.3 Communication models

WIA-FA ASL supports three communication models: Client/Server (C/S) model, Publisher/Subscriber (P/S) model, and Report source/Report sink (R/S) model. These communication models shall be used to transmit the application data with corresponding priority.

- C/S communication model: applicable to aperiodic non-realtime data with priority NRT. The transmission is bidirectional and shall be unicast.
- P/S communication model: applicable to periodic real-time data with priority RT1. The transmission is unidirectional and may be unicast or broadcast.
- R/S communication model: applicable to aperiodic urgent data with priority RT0 or aperiodic non-urgent data with priority RT2. The transmission is unidirectional and can be unicast or broadcast.

The roles played by the gateway device and field devices in different communication models are shown in Table 76.

Table 76 – Communication models between gateway device and field devices

Gateway Device	Field Device	Communication Mode	Priority	Description
Client	Server	Unicast, bidirectional	NRT	Gateway device reads or writes the attributes of MIB or UAOs, or acknowledges the alarms for field devices.
Publisher	Subscriber	Unicast, unidirectional	RT1	Gateway device publishes the output data to a field device.
		Broadcast, unidirectional	RT1	Gateway device publishes the output data to all field devices.
Subscriber	Publisher	Unicast, unidirectional	RT1	Field device publishes the input data to the gateway device.
Report sink	Report source	Unicast, unidirectional	RT2	Field device reports alarms to the gateway device.
Report source	Report sink	Unicast, unidirectional	RT0	Gateway device sends the start or stop command to a field device.
		Broadcast, unidirectional	RT0	Gateway device sends the start or stop command to all field devices.

These three communication models are accomplished by corresponding types of VCRs. VCRs define the logic communication relationships between the gateway device and field devices.

10.4 Application data

10.4.1 General

In WIA-FA devices, the application data that is used for UAOs includes attribute data, process data, and event data.

10.4.2 Attribute data

Attribute data includes MIB attributes (see 6.7.1.2) and UAO parameters, which may be aperiodically read and written by the gateway device to field devices. In addition, MIB attributes may also be gotten or set by DMAP. UAO parameters are associated with process or technology, and the definitions of UAO parameters are out of the scope of this document.

10.4.3 Process data

Process data includes input data and output data, which shall be periodically exchanged between field devices and the gateway device. Input data includes sensor measurement values, switch status, actuator feedback values, etc., which is transmitted from field devices to the gateway device. Output data includes actuator setpoints, switch set values, control values, etc., which is transmitted from the gateway device to field devices.

Process data are also divided into analog data and digital data. Such data can be value (e.g. float, integer, or bit type) or value with status. In the latter case, the status is used to indicate the quality of its value. Manufacturer specific process data is allowed in WIA-FA.

10.4.4 Event data

Event data are alarm events reported from field devices to the gateway device. WIA-FA defines EventData (see Table 77) for field devices to maintain alarm events. Each bit of EventFlag represents an alarm event (see Table 78). When an alarm event appears or disappears, its corresponding EventFlag bit shall be set (appeared) or reset (disappeared) accordingly. Each bit of AckFlag is used to indicate whether the corresponding alarm event should be acknowledged. If the alarm event is acknowledged, its EventFalg bit shall be reset. Each bit of PrioFlag is used to indicate the priority of the corresponding alarm event.

Table 77 – EventData definition

MemberID	Member name	Data type	Data length (in octet)	Description
1	EventFlag	BitField16	2	Each bit of EventFlag represents one alarm event, which is specified in Table 78 The coding for each bit is: 0 = event doesn’t exist; 1 = event exists.
2	AckFlag	BitField16	2	Each bit of AckFlag corresponds to the bit of EventFlag, and indicates whether the corresponding alarm event needs to be acknowledged. The coding for each bit is: 0 = Acknowledgement is not needed; 1 = Acknowledgement is needed.
3	PrioFlag	BitField16	2	Each bit of PrioFlag corresponds to the bit of EventFlag, and indicates the priority of the corresponding alarm event. The coding for each bit is: 0 = RT2; 1 = RT0.

Table 78 – UAO events definitions

Bit	Event Type
Bit 0	CONFIGURATION_ERROR
Bit 1	SENSOR_FAULT
Bit 2	ACTUATOR_FAULT
Bit 3	INPUT_EXCEED_UPPER_LIMIT
Bit 4	INPUT_EXCEED_LOWER_LIMIT
Bit 5	OUTPUT_EXCEED_UPPER_LIMIT
Bit 6	OUTPUT_EXCEED_LOWER_LIMIT
Bit 7	PROCESS_DATA_NOT_UPDATED
Bit 8	PROCESS_DATA_LENGTH_INCONSISTENT
Bit 9	OPEN_CIRCUIT
Bit 10	SHORT_CIRCUIT
Bit 11 to Bit 19	Reserved
Bit 20 to Bit23	Manufacturer specific events

10.5 User application process

10.5.1 General

WIA-FA defines distributed application process (DAP) to accomplish the distributed applications in the industrial field. A DAP may locate on one or more than one device over WIA-FA network. DAP is implemented as UAPs on the devices. UAP shall be identified by UAP_ID. Different UAPs on different devices belonging to one DAP shall have the same UAP_ID. A field device can support one or more UAPs, which shall be uniquely identified within the device. Figure 74 shows the relationships between DAPs and UAPs over WIA-FA network. DMAP is a special UAP implementing network and security management functions (see 6.2.1). Each WIA-FA device shall implement only one DMAP. The UAP_ID of DMAP is 0.

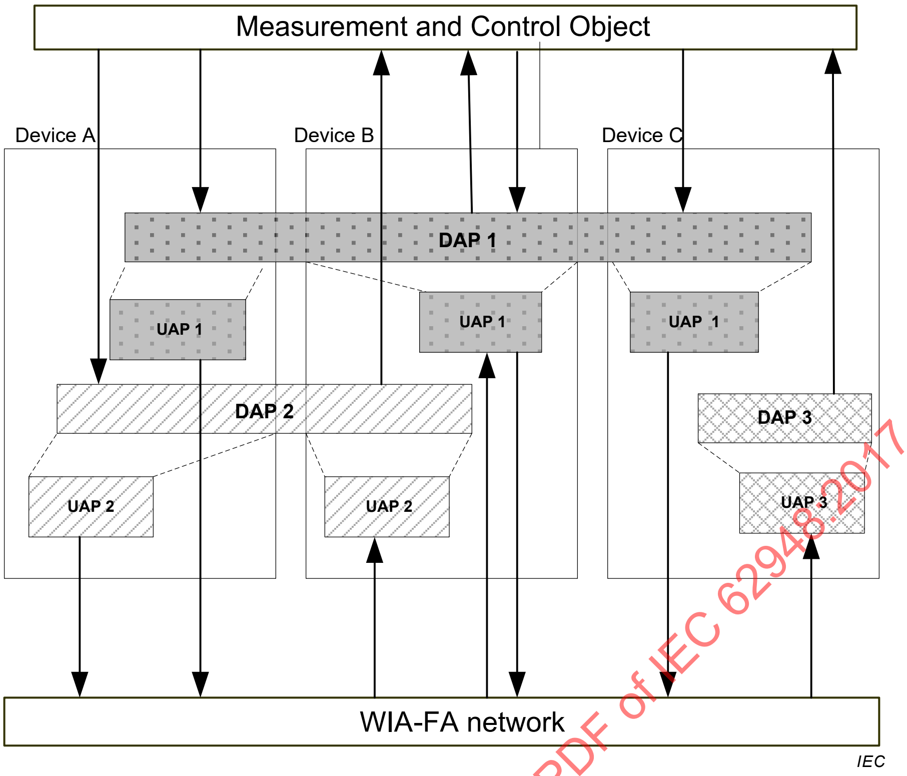


Figure 74 – The relationships between UAPs and DAPs

10.5.2 User application object

A UAP is composed of one or more than one UAO within one field device, shown in Figure 75. Each UAO shall process and exchange data across the network or within the device.

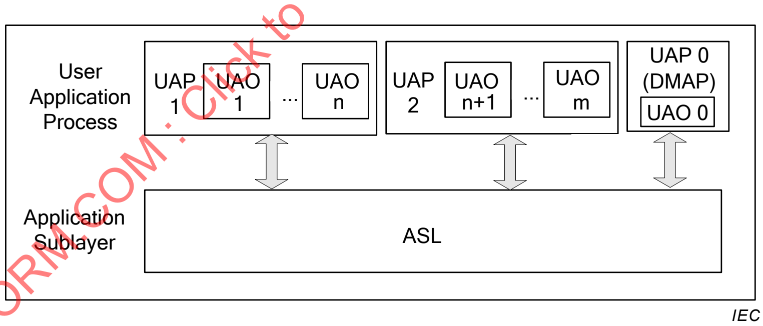


Figure 75 – User application objects in a field device

WIA-FA supports four types of UAOs: analog input (AI), analog output (AO), digital input (DI), and digital output (DO). Manufacturer specific UAOs are allowed in WIA-FA. The supported UAO types are optional for field devices. The definitions of UAO parameters are out of the scope of this document.

UAO shall be identified uniquely by UAO_ID within a device. UAO_ID shall be assigned by NM on the gateway device. DMAP is a special UAP and has only one UAO (UAO_ID = 0).

10.5.3 IO data image on gateway device

It is optional for the gateway device to implement IO data images for field devices. After a field device joins WIA-FA network successfully, the gateway device may allocate an IO data image for the field device, which is used to buffer the periodically transferred input and output data of the field device. Thus, if the gateway device interconnects with other control system (e.g. fieldbus), it acts as a remote I/O device and communicates with other devices over the control network.

The gateway device shall save the application configuration information of a field device, in order to interpret the input and output data of the field device.

When a field device leaves WIA-FA network, the gateway device shall release the IO data image (if implemented) allocated for the field device.

Figure 76 shows an example of the IO data images on the gateway device. Implementation is manufacturer specific and out of the scope of this document.

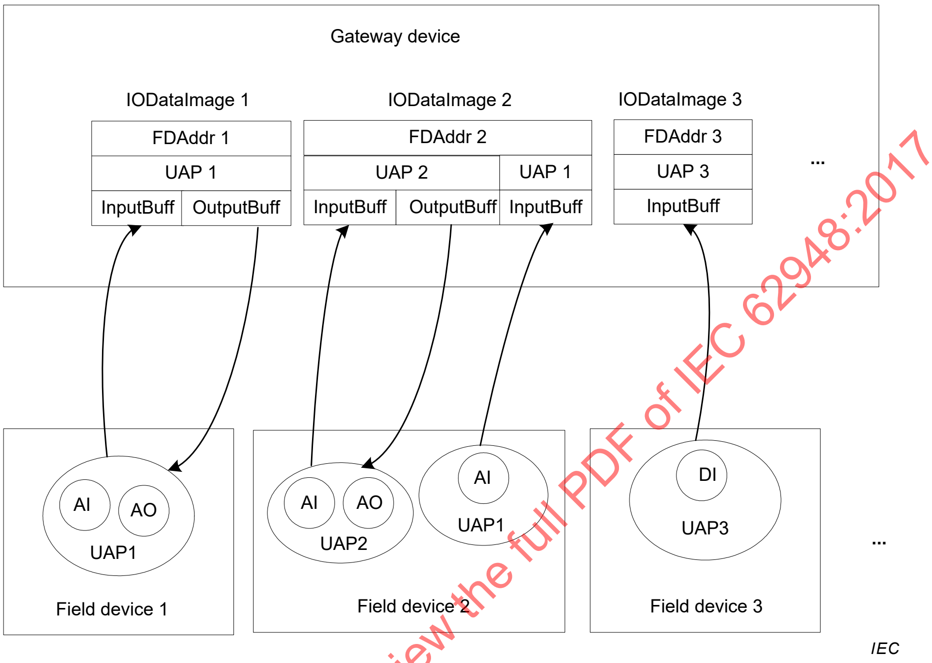


Figure 76 – Example of IO data images on the gateway device

10.5.4 Alarm mechanism

Each UAO in the field device shall maintain EventData (see Table 77) for handling alarms as follows.

- If an alarm event related to the UAO appears or disappears, the corresponding EventFlag bit of EventData shall be set (appeared) or reset (disappeared).
- The AckFlag bits of the EventData shall indicate whether the corresponding events need to be acknowledged by the host computer. The AckFlag of EventData can be configured by the host computer, via setting the AckFlag member value of UAO instance description (see Table 26).
- The PrioFlag bits of the EventData shall indicate the priorities of the corresponding events, including RT0 and RT2. The PrioFlag of EventData can be configured by the host computer, via setting the PrioFlag member value of UAO instance description (see Table 26).
- If an alarm event is acknowledged, the corresponding EventFalg bit of EventData shall also be reset.

Regardless of whether an alarm event appears or disappears, the UAO shall signal its UAP to send a Report request message (see 10.6.5.1) to the gateway device. The Report request message shall be generated on the basis of UAO's EventData. More than one alarm event may be reported in one Report request message. If the alarm events have different priority, the highest priority shall be used.

The host computer should maintain an alarm queue for each field device to save the received alarm events from this field device. Depending on the values of AckFlag bits in the received Report request message, the host computer may acknowledge the alarm events by returning a Report ACK request message (see 10.6.6.1) to the field device. More than one alarm event may be acknowledged in one Report ACK request message.

For those alarm events requiring acknowledgement, if the Report ACK request message is received within AlarmRptDur (see Table 16) time interval, UAP shall signal its UAO to clear the alarm events that are reported previously, i.e. resetting the corresponding EventFlag bits. If the Report ACK request message is not received within AlarmRptDur time interval, UAP shall report the alarm events again. AlarmRptDur can be configured by the host computer, via setting the AlarmRptDur attribute value in MIB.

10.5.5 Application configuration

10.5.5.1 General

At the beginning of configuring a WIA-FA network, the host computer (engineer) may firstly set unstructured MIB attributes (see Table 16) of the gateway device. Afterwards, NM on the gateway device may set unstructured MIB attributes to a successfully joined device.

In addition, the host computer should initiate the application configuration procedure for the field device. In other words, the host computer should assign UAPs and configure UAOs for each UAP according to application requirements.

After the host computer has completed the application configuration, NM on the gateway device shall configure VCRs, allocate communication resources, and set other network-related attributes for the field device.

The host computer and NM may use ASL Read/Write services, or operate via DMAP-SAP to use DLL Remote Attribute Get/Set services of the gateway device to implement configurations.

10.5.5.2 UAP configuration

According to application requirements, the host computer should assign UAPs for each field device. A field device should be assigned one or more UAPs, which shall be uniquely identified within the device. This is implemented by specifying UAP_IDs and allocate UAOs to individual UAP (see 10.5.5.3).

10.5.5.3 UAO configuration

If a field device joins WIA-FA network successfully, NM on the gateway device should firstly get its NumOfSupUAO (see Table 21) attribute to obtain the number of UAO classes supported by the field device, and get SupUAOList (see Table 17) to obtain all UAO class descriptions. The UAO class description includes ClassID, UAO type, minimum data update rate, as well as the data type and maximum length of the input data and output data supported by the UAO class. See Table 24 for the definition of UAO class description. NM shall deliver the attribute values of NumOfSupUAO and SupUAOList to the host computer.

The host computer should determine the number of UAPs on the field device according to application requirements, and instantiate UAOs derived from the UAO classes that are supported by the field device. UAO instantiation shall include:

- assigning UAOs to each UAP;
- specifying data types of the input data and output data for each UAO;
- specifying the count of data with corresponding data type to be periodically transferred;
- specifying the input and output data update rates.

See Table 26 for the definition of UAO instance description. All UAOs belonging to one UAP shall have the same data update rate.

Therefore, the host computer shall set NumOfCfgUAO (see Table 21) and CfgUAOList (see Table 17) to the field device. The sum of all input/output data length of a configured UAO shall be less than the maximum input/output data length of the UAO class. The length of periodically transferred input/output data shall be consistent with the application configurations. And the data order shall be consistent with the order of UAOs that are assigned to one UAP in CfgUAOList. When a field device leaves WIA-FA network, it shall clear its CfgUAOList and set NumCfgUAO to 0.

10.5.5.4 VCR configuration

10.5.5.4.1 General

After configuring UAOs, NM on the gateway device shall configure VCRs for the field device according to the UAO application configuration. NM shall use DLME Remote Attribute Set service to write VCRList (see Table 17) to the field device for configuring C/S VCRs, P/S VCRs, and R/S VCRs.

A VCR endpoint (see Table 23) is the implementation of a VCR in a device. The VCR endpoint shall take the corresponding role and specify attributes to associate application to communication. A VCR shall be identified uniquely by VCR_ID in a device.

Table 79 shows the overview of attribute settings for different role types of VCR endpoints on a field device.

Table 79 – VCR attribute configuration for a field device

MemberID	Member name	VcrEP_Type			
		SERVER	PUBLISHER	SUBSCRIBER	REPORT SOURCE
0	VCR_ID	Configured	Configured	Configured	Configured
1	VcrEP_Type	Configured to 1	Configured to 2	Configured to 3	Configured to 4
2	UAP_ID	Configured	Configured	Configured	Configured
3	PeerAddr	Configured to short address of gateway device	Configured to short address of gateway device	Configured to short address of gateway device	Configured to short address of gateway device
4	VCRActiveTime	Invalid and should be set to 0	Configured	Configured	Invalid and should be set to 0
5	DataUpdateRate	Invalid and should be set to 0	Configured	Configured	Invalid and should be set to 0
6	Deadline	Invalid and should be set to 0	Configured	Configured	Invalid and should be set to 0
7	WatchdogTime	Configurable and the default value is 10 000 ms	Invalid and should be set to 0	Invalid and should be set to 0	Invalid and should be set to 0

If NM on the gateway device configures the VCRList for a field device, it shall also create the corresponding peer VCR endpoints in order to establish VCRs with the field device.

If a field device leaves WIA-FA network, it should clear the VCRList.

10.5.5.4.2 C/S VCR configuration

Each UAP (including DMAP with UAP_ID = 0) on the field device shall be assigned one C/S VCR between the gateway device and the field device. The role of the VCR endpoint on the gateway device is Client while that on the field device is Server.

Figure 77 shows the C/S VCR relationship between the gateway device and field devices. Arrows indicate the data transmission directions.

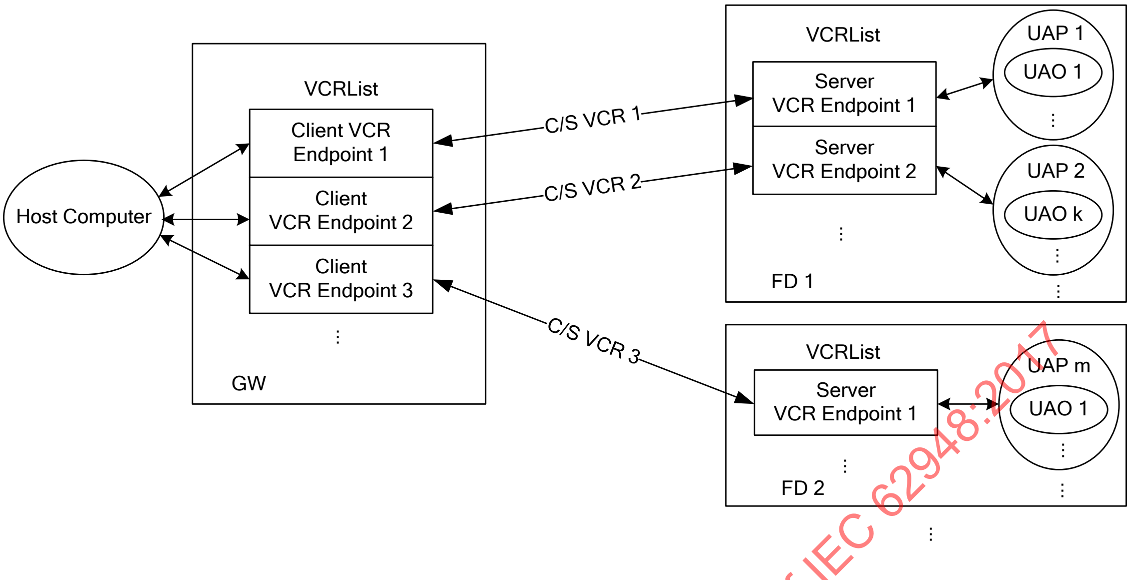


Figure 77 – C/S VCR relationships between GW and FDs

10.5.5.4.3 P/S VCR configuration

Multiple UAOs may be allocated to one UAP on a field device. If the UAOs of a UAP have input data, a Publisher VCR endpoint shall be configured for the UAP. Similarly, if the UAOs of a UAP have output data, a Subscriber VCR endpoint shall be configured for the UAP. Meanwhile, NM shall create the corresponding Subscriber or Publisher VCR endpoint on the gateway device. Each Publisher or Subscriber VCR endpoint should be allocated a buffer to store the input or output data. Thus, when new input or output data comes, the new data always overwrites the old data in the buffer, so that the new data is always transmitted in the network.

Figure 78 shows the P/S VCR relationships between the gateway device and field devices. Arrows indicate the data transmission directions. Dash line means the gateway device may or may not exchange the process data of all configured field devices with the host computer or other automation network (e.g., fieldbus).

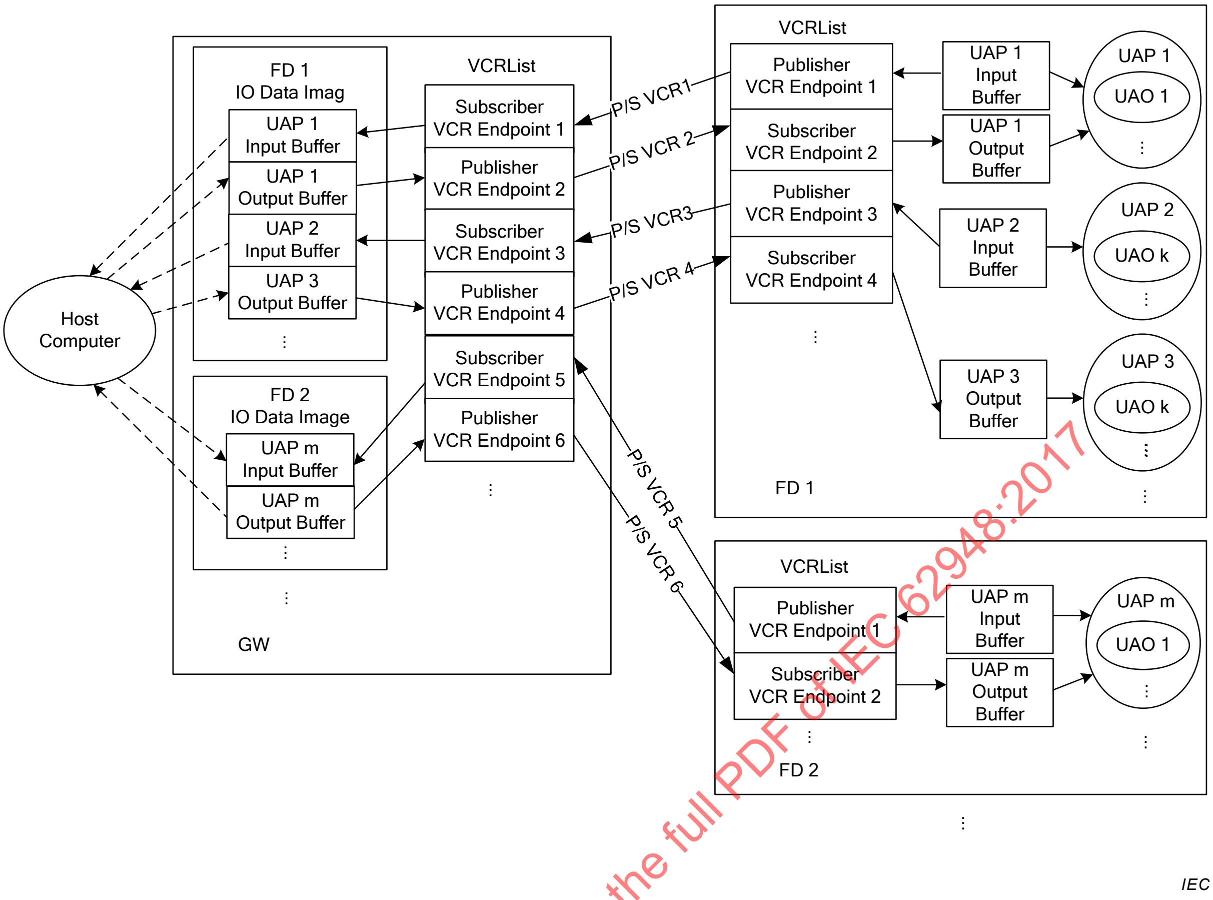


Figure 78 – P/S VCR relationships between GW and FDs

10.5.5.4.4 R/S VCR configuration

Each UAP on the field device shall be assigned a pair of R/S VCRs between the gateway device and the field device. One UAP is used by the field device to report alarms to the gateway device. The role of this VCR endpoint on the field device is Report source, while that on the gateway device is Report sink.

The other R/S VCR is used by the gateway device to send start or stop command to the field device. The role of this VCR endpoint on the gateway device is Report source, while that on the field device is Report sink.

Figure 79 shows the R/S VCR relationships between the gateway device and field devices. Arrows indicate the data transmission directions.

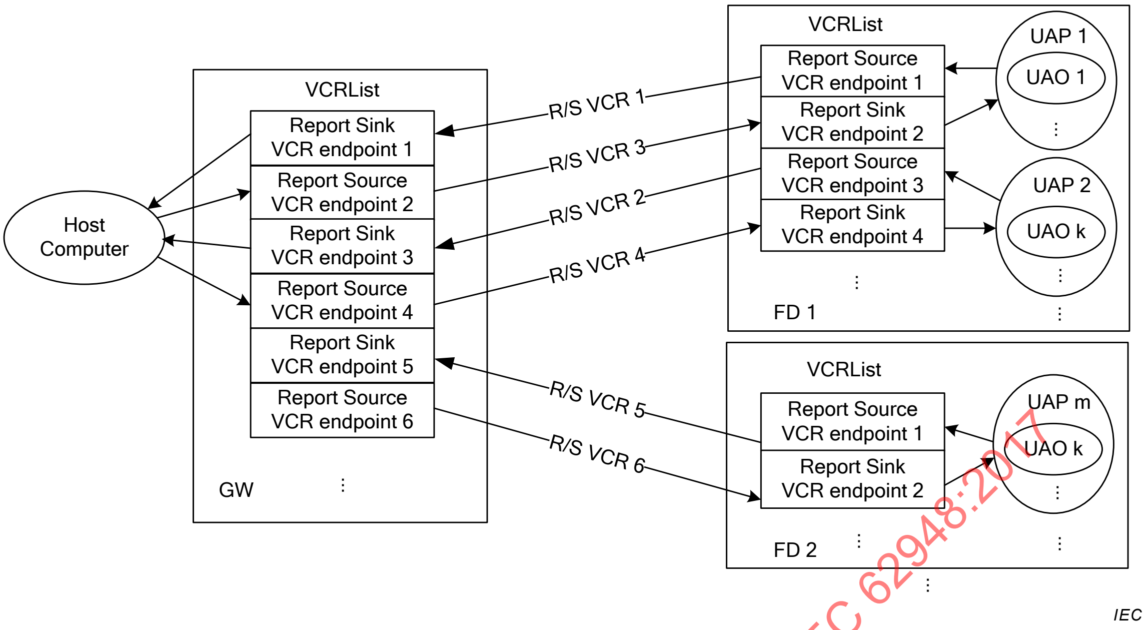

VCRList
Report Sink VCR endpoint 1
Report Source VCR endpoint 2
Report Sink VCR endpoint 3
Report Source VCR endpoint 4
Report Sink VCR endpoint 5
Report Source VCR endpoint 6
GW
Host Computer
R/S VCR 1
R/S VCR 3
R/S VCR 2
R/S VCR 4
R/S VCR 5
R/S VCR 6
VCRList
Report Source VCR endpoint 1
Report Sink VCR endpoint 2
Report Source VCR endpoint 3
Report Sink VCR endpoint 4
UAP 1
UAO 1
UAP 2
UAO k
FD 1
VCRList
Report Source VCR endpoint 1
Report Sink VCR endpoint 2
UAP m
UAO k
FD 2
IEC

Figure 79 – R/S VCR relationships between GW and FDs

10.5.5.5 Application configuration procedure

Figure 80 shows the application configuration procedure from the gateway device to a field device.

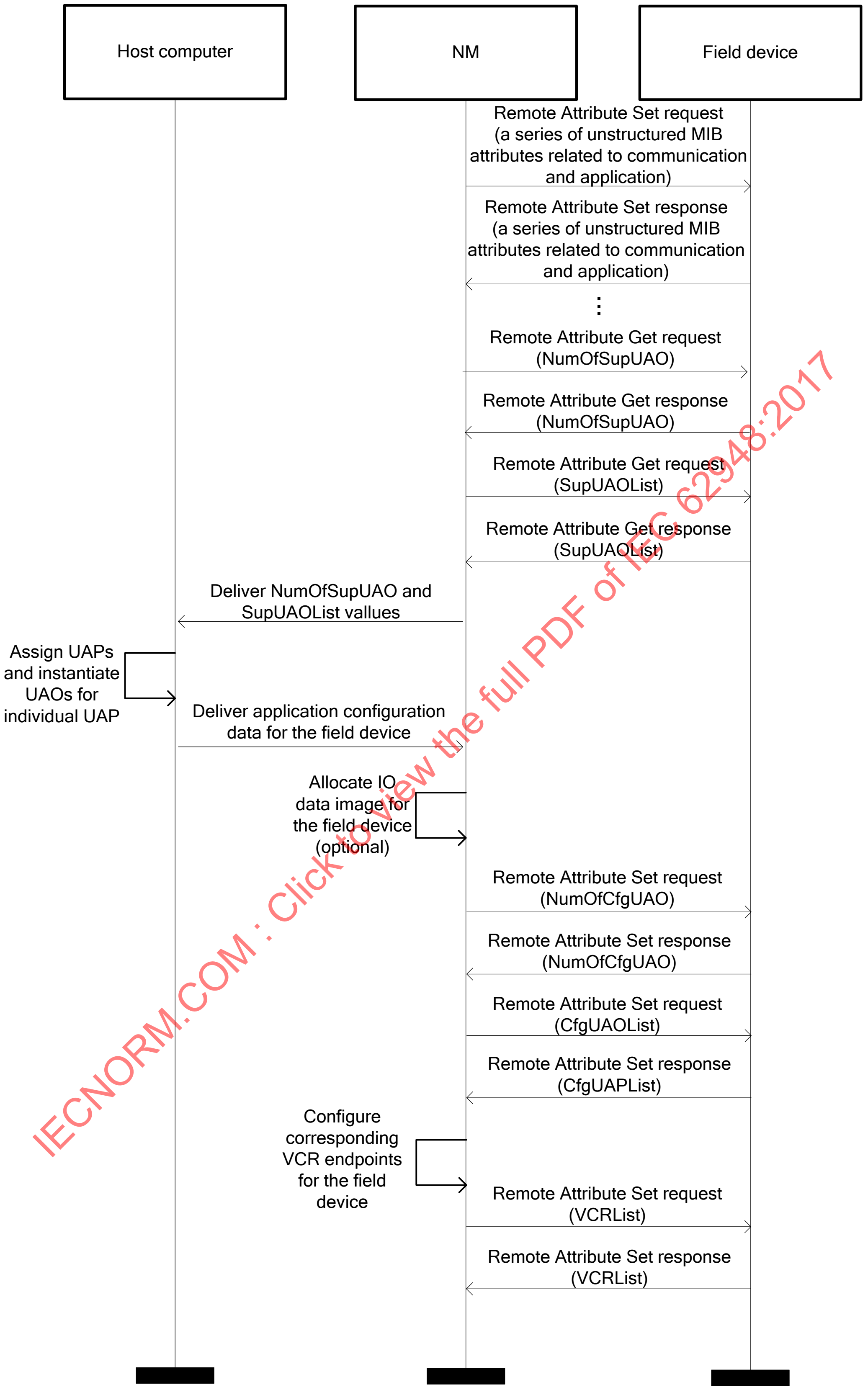

Host computer
NM
Field device
Remote Attribute Set request (a series of unstructured MIB attributes related to communication and application)
Remote Attribute Set response (a series of unstructured MIB attributes related to communication and application)
⋮
Remote Attribute Get request (NumOfSupUAO)
Remote Attribute Get response (NumOfSupUAO)
Remote Attribute Get request (SupUAOList)
Remote Attribute Get response (SupUAOList)
Deliver NumOfSupUAO and SupUAOList vallues
Assign UAPs and instantiate UAOs for individual UAP
Deliver application configuration data for the field device
Allocate IO data image for the field device (optional)
Remote Attribute Set request (NumOfCfgUAO)
Remote Attribute Set response (NumOfCfgUAO)
Remote Attribute Set request (CfgUAOList)
Remote Attribute Set response (CfgUAPList)
Configure corresponding VCR endpoints for the field device
Remote Attribute Set request (VCRList)
Remote Attribute Set response (VCRList)
IEC

Figure 80 – Application configuration procedure for a field device

10.5.5.6 Aggregation and disaggregation of process data

Data of all UAOs allocated to one UAP shall be aggregated on the device. As a publisher, the UAP shall collect all process data from its UAOs according the application configuration, and ASL shall pack the input data into one Publish request message. This is the AL aggregation.

As a subscriber, ASL shall parse the received Publish request message, and UAP shall distribute the process data to its respective UAOs according application configuration. This is the AL disaggregation.

Figure 81 shows an example of the UAO data aggregation and disaggregation procedures.

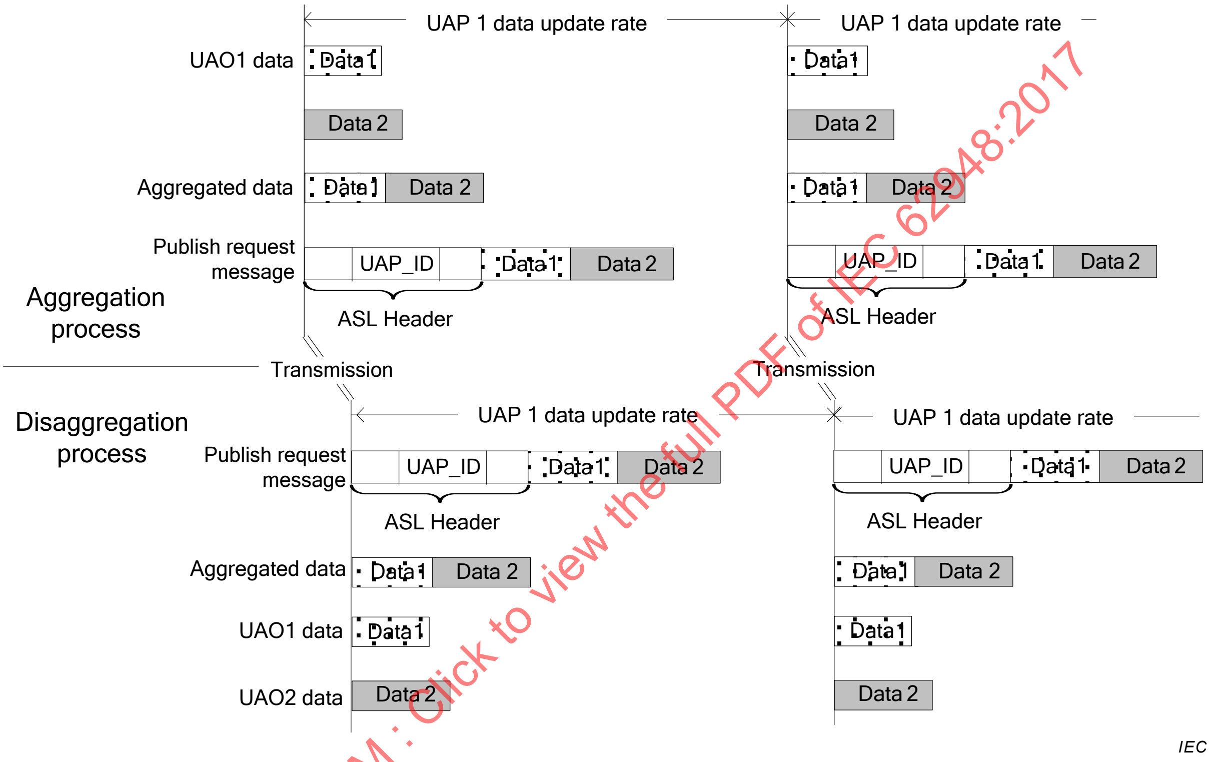


Figure 81 – Example of UAO data aggregation and disaggregation process

10.6 Application services

10.6.1 Confirmed services and unconfirmed services

WIA-FA application layer defines application services to support aperiodic attribute read/write access, periodic process data publishment, alarm event reports, and urgent command transmission, as shown in Table 80.

The application services include confirmed services and unconfirmed services. Confirmed services are used for bidirectional request/response between UAPs on different devices, and the unconfirmed services are used for unidirectional data transmission from one UAP to one or more UAPs on different devices.

Table 80 – Application services used by UAPs

Service Name	Service Identifier	Message Type	Description
Read	0x01	REQUEST	Request to read the value of a UAO parameter or MIB attribute
		RESPONSE_P	Positive response indicating that the UAO parameter or MIB attribute is read successfully, and the value of the attribute shall be returned in the response.
		RESPONSE_N	Negative response indicating that the UAO parameter or MIB attribute is read unsuccessfully. The failure reason shall be returned in the response.
Write	0x02	REQUEST	Request to write the value of a UAO parameter or MIB attribute
		RESPONSE_P	Positive response indicating that the UAO parameter or MIB attribute is written successfully
		RESPONSE_N	Negative response indicating that the UAO parameter or MIB attribute is written unsuccessfully. The failure reason shall be returned in the response.
Publish	0x03	REQUEST	Request to publish the input or output data
Report	0x04	REQUEST	Request to report a or multiple UAO alarm event(s)
Report ACK	0x05	REQUEST	Acknowledge the reported alarm event(s)
		RESPONSE_P	Positive response indicating that the alarm event(s) is/are acknowledged successfully
		RESPONSE_N	Negative response indicating that the alarm event(s) is/are acknowledged unsuccessfully. The failure reason shall be returned in the response.
Start	0x06	REQUEST	Request to activate UAP(s)
Stop	0x07	REQUEST	Request to deactivate UAP(s)

Subclauses 10.6.2 to 10.6.8 define the formats of application services message, and describe the corresponding service processes. These application service messages are the user data transferred between UAP and ASL. ASL shall pack the application service message into ASL message by adding ASL header according to ASL general message format (see 10.7.4).

10.6.2 Read service

10.6.2.1 Read message formats

Figure 82 shows the format of Read request message.

1 octet	1 octet	2 octets	1 octet
UAO Identifier	Attribute Identifier	Storage Index	Member Identifier

IEC

Figure 82 – Read request message format

The fields of the Read request message are described as follows:

- UAO Identifier: UAO_ID of the UAO to be read; value 0 is used for MIB;

- Attribute Identifier: AttributeID of the attribute in UAO or MIB;
- Storage Index: index of a record in the list, which is valid only for List structured attributes; value 0xFFFF is reserved to indicate all records in the list, and in this case the field of Member identifier is invalid;
- Member Identifier: MemberID of a member in a structured attribute, which is valid for the structured attributes; value 255 is reserved to indicate all members.

Figure 83 shows the format of Read positive response message.

1 octet	1 octet	2 octets	1 octet	Variable
UAO Identifier	Attribute Identifier	Storage Index	Member Identifier	Data

IEC

Figure 83 – Read positive response message format

Figure 84 shows the format of Read negative response message.

1 octet	1 octet	2 octets	1 octet	1 octet	1 octet
UAO Identifier	Attribute Identifier	Storage Index	Member Identifier	Error Code	Additional Information

IEC

Figure 84 – Read negative response message format

The fields of the Read response messages are described as follows:

- UAO Identifier: UAO_ID of the UAO that is read; value 0 is used for MIB;
- Attribute Identifier: AttributeID of the attribute in UAO or MIB;
- Storage Index: index of a record in the list, which is valid only for List structured attributes; value 0xFFFF is reserved to indicate all records, and in this case the field of Member identifier is invalid;
- Member Identifier: MemberID of a member in a structured attribute, which is valid for the structured attributes; value 255 is reserved to indicate all members;
- Data: values that are read;
- ErrorCode: failure reason codes; the definition is shown in Table 81;
- Additional Information: manufacturer specific additional information for failure reasons.

Table 81 – Error code definition for Read negative response message

Coding	Definition	Description
1	SERVICE_EXPIRATION	Service response has not been received within predefined WatchdogTime interval
2	SERVICE_NOT_SUPPORTED	Service requested is not supported
3	UAO_NOT_EXISTENT	UAO_ID does not exist
4	ATTRIBUTE_NOT_EXISTENT	AttributeID in the UAO does not exist
5	STOREINDEX_NOT_EXISTENT	Index in a list does not exist
6	MEMBER_NOT_EXISTENT	MemberID in a structured attribute does not exist
7	UAP_NOT_EXISTENT	UAP_ID does not exist
8	OTHERS	Other reasons
8 to 255	Reserved	Reserved for future use

10.6.2.2 Read service process

This confirmed service transmits NRT data via C/S VCR. The gateway device shall use this service to aperiodically read an attribute or an attribute member in a UAO or MIB. If the UAO_ID, AttributeID, or MemberID is wrong, or the service is not supported by the field device, it shall return the appropriate error code. Figure 85 shows the Read service process.

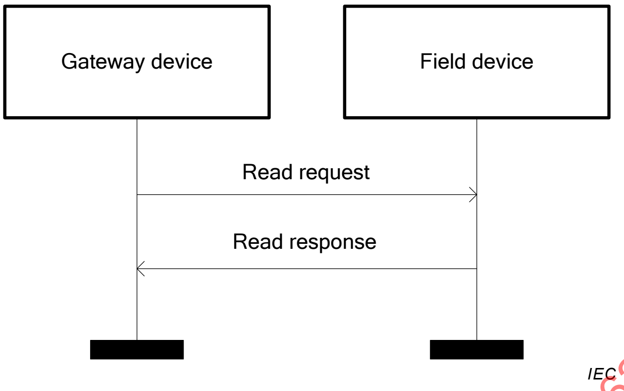


Figure 85 – Read service process

10.6.3 Write service

10.6.3.1 Write message formats

Figure 86 shows the format of Write request message.

1 octet	1 octet	2 octet	1 octet	Variable
UAO Identifier	Attribute Identifier	Storage Index	Member Identifier	Data

IEC

Figure 86 – Write request message format

The fields of the Write request message are described as follows:

- UAO Identifier: UAO_ID of the UAO to be written; value 0 is used for MIB;
- Attribute Identifier: AttributeID of the attribute in UAO or MIB;
- Storage Index: index of a record in a list, which is valid only for List structured attributes; value 0xFFFF is reserved to indicate all records in the list, and in this case the field of Member identifier is invalid;
- Member Identifier: MemberID of a member in a structured attribute, which is valid for the structured attributes; value 255 is reserved to indicate all members shall be read;
- Data: values that are written.

No format is defined for Write positive response message.

Figure 87 shows the format of Write negative response message.

1 octet	1 octet	2 octets	1 octet	1 octet	1 octet
UAO Identifier	Attribute Identifier	Storage Index	Member Identifier	Error Code	Additional Information

IEC

Figure 87 – Write negative response message format

The fields of the Write response messages are described as follows:

- UAO Identifier: UAO_ID of the UAO that is written; value 0 is used for MIB;

- Attribute Identifier: AttributeID of the attribute in UAO or MIB.
- Storage Index: index of a record in the list, which is valid only for List structured attributes; value 0xFFFF is reserved to indicate all records, and in this case the field of Member identifier is invalid;
- Member Identifier: MemberID of a member in a structured attribute, which is valid for the structured attributes; value 255 is reserved to indicate all members;
- ErrorCode: Failure reason codes; the definition is shown in Table 82;
- Additional Information: manufacturer specific additional information for failure reasons.

Table 82 – Error code definition for Write negative response message

Coding	Definition	Description
1	SERVICE_EXPIRATION	Service response has not been received within pre-defined WatchdogTime interval
2	SERVICE_NOT_SUPPORTED	Service requested is not supported
3	UAO_NOT_EXISTENT	UAO_ID does not exist
4	ATTRIBUTE_NOT_EXISTENT	AttributeID in the UAO does not exist
5	STOREINDEX_NOT_EXISTENT	Index in a list does not exist
6	MEMBER_NOT_EXISTENT	MemberID in a structured attribute does not exist
7	LENGTH_NOT_MATCH	The length of data to be written is not consistent with the length of the objective
8	VALUE_EXCEED_SCOPE	The value of data to be written exceeds the valid value range of the objective
9	UAP_NOT_EXISTENT	UAP_ID does not exist
10	OTHERS	Other reasons
11 to 255	Reserved	Reserved for future use

10.6.3.2 Write service process

This confirmed service transmits NRT data via C/S VCR. The gateway device shall use this service to aperiodically write an attribute or an attribute member in a UAO or MIB. If the UAO_ID, AttributeID, or MemberID is wrong, or the value exceeds valid range, or the data length does not match, or the service is not supported by the field device, it shall return the appropriate error code. Figure 88 shows the Write service process.

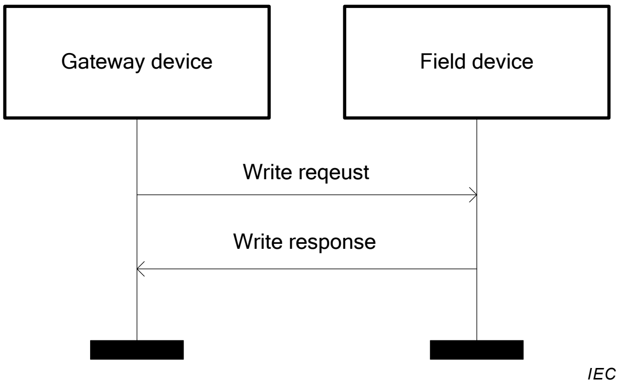


Figure 88 – Write service process

10.6.4 Publish service

10.6.4.1 Publish message format

Figure 89 shows the format of the Publish request message.

Variable Length in octet
Process Data

IEC

Figure 89 – Publish request message format

The fields of the Publish request message are described as follows:

– Process Data: all input or output data that is published by UAP.

Publish service is an unconfirmed service and has no response.

10.6.4.2 Publish service process

This unconfirmed service transmits RT1 data via P/S VCR. The gateway device or field devices shall use this service to periodically publish the process data with DataUpdateRate cycle. Data of all UAOs belonging to one UAP shall be aggregated in one Publish request message. Figure 90 shows the Publish service process from a field device to the gateway and Figure 91 shows the opposite direction.

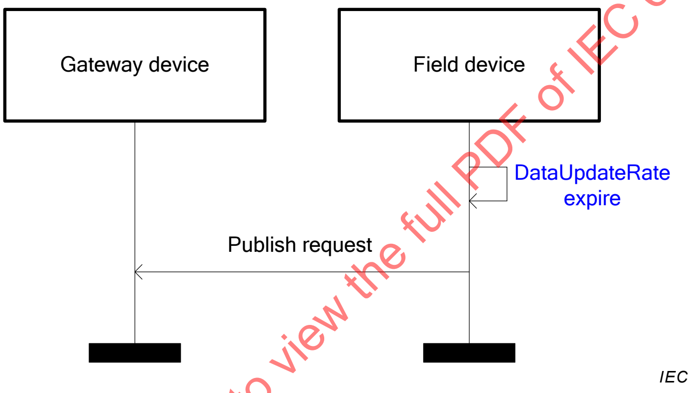


Figure 90 – Publish process from FD to GW

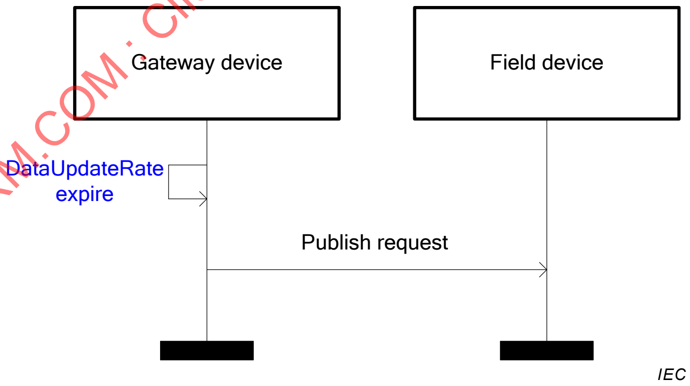


Figure 91 – Publish process from GW to FD

10.6.5 Report service

10.6.5.1 Report message format

Figure 92 shows the format of Report request message.

1 octet	2 octets	2 octets	2 octets	1 octet
UAO Identifier	Event Flag	Appear Flag	Ack Flag	Additional Information

IEC

Figure 92 – Report request message format

The fields of the Report request message are described as follows:

- UAO Identifier: UAO_ID of the UAO that reports the alarm events; value 0 is used for MIB;
- Event Flag: the reported alarm events; regardless of whether the alarm events appear or disappear, they shall be reported from the field device to the gateway device; see Table 78 for definition of each bit;
- Appear Flag: each bit corresponds to the bit of Event flag and indicates whether the corresponding alarm event appears or disappears; the bit value 1 means the corresponding alarm event appears, and the bit value 0 means the corresponding alarm event disappears or does not exist;
- Ack Flag: each bit corresponds to the bit of Event flag and indicates whether the corresponding alarm event needs to be acknowledged; the value shall be set as the value of AckFlag bits of the UAO's EventData (see Table 77);
- Additional Information: manufacturer specific additional information.

Report service is an unconfirmed service and has no response.

10.6.5.2 Report message process

This unconfirmed service transmits RT0 or RT2 data via R/S VCR. The field device shall use this service to report the appearance or disappearance of one or more than one alarm event to the gateway device. Figure 93 shows the Report service process.

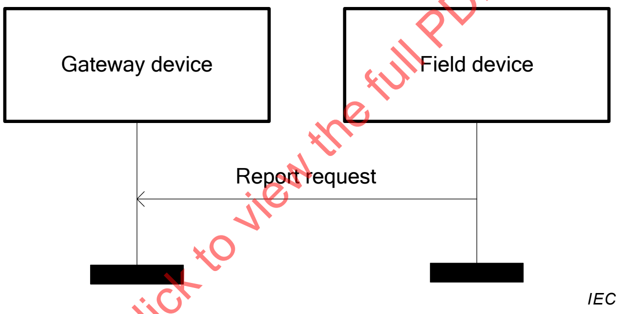


Figure 93 – Report service process

10.6.6 Report ACK service

10.6.6.1 Report ACK message format

Figure 94 shows the format of Report ACK request message.

1 octet	2 octets
UAO Identifier	AckEvent

IEC

Figure 94 – Report ACK request message format

Figure 95 shows the format of Report ACK positive response message.

1 octet
UAO IdentifierID

IEC

Figure 95 – Report ACK positive response message format

Figure 96 shows the format of Report ACK negative response message.

1 octet	1 octet	1 octet
UAO Identifier	ErrorCode	Additional information

IEC

Figure 96 – Report ACK negative response message format

The fields of the Report ACK response messages are described as follows:

- UAO Identifier: UAO_ID of the UAO whose alarm events are acknowledged;
- AckEvent: acknowledged alarm events; each bit represents one event defined in Table 78; the bits with value 1 indicate the acknowledgements of previously reported alarm events;
- ErrorCode: failure reason code; see Table 83 for definition;
- Additional Information: manufacturer specific additional information.

Table 83 – Error code definition for Report ACK negative response

Coding	Definition	Description
1	SERVICE_EXPIRATION	Service response has not been received within pre-defined WatchdogTime interval
2	SERVICE_NOT_SUPPORTED	Service requested is not supported
3	UAO_NOT_EXISTENT	UAO_ID does not exist
4	EVENT_NOT_EXISTENT	Event to be acknowledged does not exist
5	ACKNOWLEDGEMENT_NOT_REQUIRED	Event to be acknowledged has not been reported previously
6	OTHER	Other reasons
7 to 255	Reserved	Reserved for future use

10.6.6.2 Report ACK service process

As a confirmed service, this service transmits NRT data via C/S VCR. The gateway device shall use this service to acknowledge all or part of the alarm events that have been reported previously by the field device. Figure 97 shows the Report ACK service process.

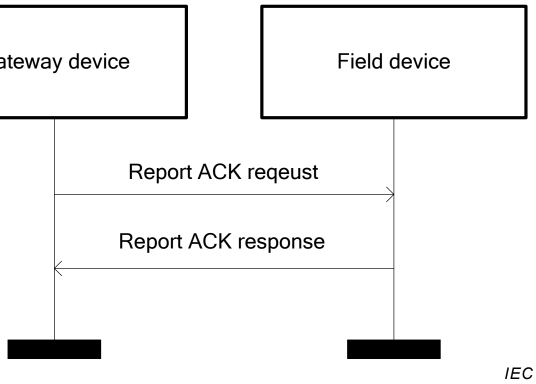


Figure 97 – Report ACK service process

10.6.7 Start service

10.6.7.1 Start message format

Start request has no data so there is no special message format. It shall be identified by Service identifier subfield of Message control field in ASL header.

Start service is an unconfirmed service and has no response message.

10.6.7.2 Start service process

This unconfirmed service transmits RT0 data via R/S VCR. The gateway device shall use this service to activate one UAP or all UAPs immediately. UAP is identified by UAP_ID in ASL header. This service may be used in the following cases.

- The gateway device unicast the Start request message to a field device with UAP_ID != 0xFFFF, which means to enable one UAP on the field device active immediately.
- The gateway device unicast the Start request message to a field device with UAP_ID = 0xFFFF, which means to enable all UAPs on the field device active immediately.
- The gateway device broadcast the Start request message to all field devices with UAP_ID != 0xFFFF, which means to enable one UAP on multiple field devices active immediately.
- The gateway device broadcast the Start request message to all field devices with UAP_ID = 0xFFFF, which means to enable all UAPs on all field devices active immediately.

If a UAP on a device receives the Start request (also named Start command) and the UAP is not active, it shall enable all its UAOs to exchange data with the process as well as activate its associated P/S VCR. If the UAP has been active already, it shall ignore the Start request.

Figure 98 shows the Start service process.

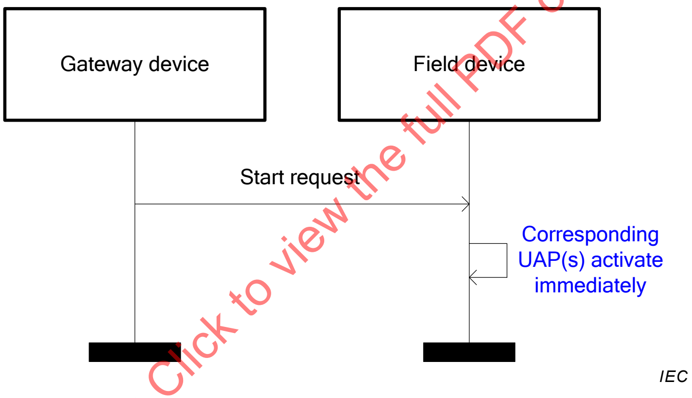


Figure 98 – Start service process

10.6.8 Stop service

10.6.8.1 Stop message format

Stop request has no data so there is no special message format. It shall be identified by Service identifier subfield of Message control field in ASL header.

Stop service is an unconfirmed service and has no response message.

10.6.8.2 Stop service process

This unconfirmed service transmits RT0 data via R/S VCR. The gateway device shall use this service to deactivate one UAP or all UAPs immediately. UAP is identified by UAP_ID in ASL header. This service may be used in the following cases.

- The gateway device unicast the Stop request message to a field device with UAP_ID != 0xFFFF, which means to deactivate one UAP on the field device immediately.
- The gateway device unicast the Stop request message to a field device with UAP_ID = 0xFFFF, which means to deactivate all UAPs on the field device immediately.
- The gateway device broadcast the Stop request message to all field devices with UAP_ID != 0xFFFF, which means to deactivate one UAP on multiple field devices immediately.

– The gateway device broadcast the Stop request message to all field devices with UAP_ID = 0xFFFF, which means to deactivate all UAPs on all field devices immediately.

If a UAP on a device receives the Stop request (also named Stop command) and the UAP is active, it shall disable all its UAOs to exchange data with the process as well as deactivate its associated P/S VCR. If the UAP has been not active, it shall ignore the Stop request.

Figure 99 shows the Stop service process.

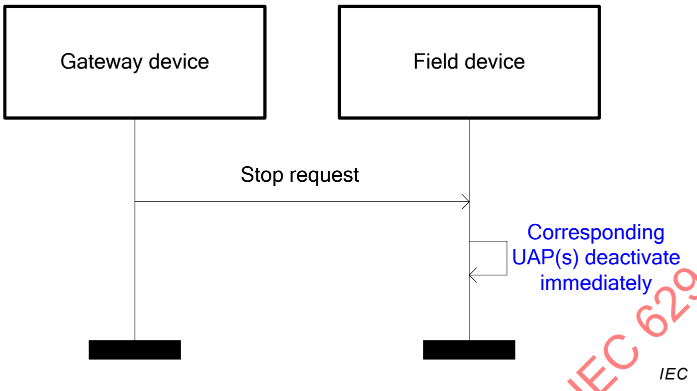


Figure 99 – Stop service process

10.7 Application sub-layer

10.7.1 Overview

Application sub-layer (ASL) provides the transparent end-to-end data transmission service for UAPs. And ASL also provides management services such as activating and deactivating P/S VCRs, and signalling alarm events to UAP.

10.7.2 ASL data service

10.7.2.1 General

ASL provides data service for UAPs in order to exchange application data between UAPs over WIA-FA network.

The ASL data service primitives include ASLDE-DATA.request, ASLDE-DATA.indication, ASLDE-DATA.response, and ASLDE-DATA.confirm.

10.7.2.2 ASLDE-DATA.request primitive

UAP shall transfer the application service request message to ASL by invoking ASLDE-DATA.request primitive. ASL shall pack an ASL request message as APDU by adding ASL header, and send it to DLL.

The definition of ASLDE-DATA.request primitive is as follows.

```
ASLDE-DATA.request(
                     DstAddr,
                     ServiceID,
                     UAP_ID,
                     Priority,
                     AsduLength,
                     Asdu
                     )
```

The parameters of ASLDE-DATA.request primitive are described in Table 84.

Table 84 – ASLDE-DATA.request primitive parameter definitions

Parameter	Data type	Valid range	Description
DstAddr	Unsigned16	0 to 65 535	Destination address of the application service request
ServiceID	Unsigned8	1 to 255	Service identifier of the application service, see Table 80
UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to

Priority	Unsigned8	0 to 255	Priority of AL user data
			The values are:
			0 = RT0;
			1 = RT1;
			2 = RT2;
			4 = NRT;
			other values are reserved.
			See 6.4.2.1 for priority details

AsduLength	Unsigned16	0 to 65 535	Length of the application service message
Asdu	Octetstring		Application service message

10.7.2.3 ASLDE-DATA.indication primitive

When ASL receives an ASL request message as APDU from DLL, it shall parse the ASL request message, and invoke ASLDE-DATA.indication primitive to transfer the application service request message to UAP.

The definition of ASLDE-DATA.indication primitive is as follows.

```
ASLDE-DATA.indication(
                                ServiceID,
                                UAP_ID,
                                AsduLength,
                                Asdu
                                )
```

The parameters of ASLDE-DATA.indication primitive are described in Table 85.

Table 85 – ASLDE-DATA.indication primitive parameter definitions

Parameter	Data type	Value Range	Description
ServiceID	Unsigned8	0 to 255	Service identifier of the application service, see Table 80
UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to
AsduLength	Unsigned16	0 to 65 535	Length of the application service message
Asdu	Octetstring		Application service message

10.7.2.4 ASLDE-DATA.response primitive

UAP shall transfer the application service response message to ASL by invoking ASLDE-DATA.response primitive. ASL shall pack an ASL response message as APDU by adding ASL header, and send it to DLL.

The definition of ASLDE-DATA.response primitive is as follows.

```
ASLDE-DATA.response(
                        ServiceID,
                        MsgType, UAP_ID,
                        AsduLength,
                        Asdu
                        )
```

Parameters of ASLDE-DATA.response are described in Table 86.

Table 86 – ASLDE-DATA.response primitive parameter definitions

Parameter	Data type	Value Range	Description
ServiceID	Unsigned8	0 to 255	Service identifier of the application service, see Table 80

MsgType	Usigned8	0 to 255	Message type of the application service. The value shall be as follows:
			0 = REQUEST;
			1 = RESPONSE_P;
			2 = RESPONSE_N;
			Others are reserved.

UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to
AsduLength	Unsigned16	0 to 65 535	Length of the application service message
Asdu	Octetstring		Application service message

10.7.2.5 ASLDE-DATA.confirm primitive

When ASL receives an ASL response message as APDU transferred from DLL, it shall parse the ASL response message, and invoke ASLDE-DATA.confirm primitive to transfer the application service response message to UAP.

The definition of ASLDE-DATA.confirm primitive is as follows.

```
ASLDE-DATA.confirm(
                      SrcAddr,
                      ServiceID,
                      MsgType,
                      UAP_ID,
                      AsduLength,
                      Asdu
                      )
```

Parameters of ASLDE-DATA.confirm are described in Table 87.

Table 87 – ASLDE-DATA.confirm primitive parameter definitions

Parameter	Data type	Value Range	Description
SrcAddr	Unsigned16	0 to 65 535	Source address of the application service response
ServiceID	Unsigned8	0 to 255	Service identifier of the application service, see Table 80

MsgType	Usigned8	0 to 255	Message type of the application service response. The value shall be as follows:
			1 = RESPONSE_P;
			2 = RESPONSE_N;
			others are reserved.

UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to
AsduLength	Unsigned16		Length of the application service message
Asdu	Octetstring		Application service message

10.7.3 ASL management service

10.7.3.1 General

The ASL management entity service access point (ASLME-SAP) provides the following primitives for management functions:

- activating and deactivating VCR services for P/S VCRs;
- signalling alarm events to UAP.

The ASL management service primitives include ASLME-VcrActive.request, ASLME-VcrDeactive.request, and ASLME_SignalEvent.indication.

10.7.3.2 ASLME-VcrActive.request primitive

UAP shall invoke ASLME-VcrActive.request primitive to activate the P/S VCR if the UAP receives a Start request from the host computer.

The definition of ASLME-VcrActive.request primitive is as follows.

```
ASLME-VcrActive.request (
                         PeerAddr,
                         UAP_ID
                         )
```

The parameters of ASLME-VcrActive.request primitive are described in Table 88.

Table 88 – ASLME-VcrActive.request primitive parameter definitions

Parameter	Data type	Valid range	Description
PeerAddr	Unsigned16	0 to 65 535	Peer address of the application service request
UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to

10.7.3.3 ASLME-VcrDeactive.request primitive

UAP shall invoke ASLME-VcrDeactive.request primitive to deactivate the P/S VCR if the UAP receives a Stop request from the host computer.

The definition of ASLME-Vcrdeactive.request primitive is as follows.

```
ASLME-VcrDeactive.request (
                           PeerAddr,
                           UAP_ID
                           )
```

The parameters of ASLME-VcrDeactive.request primitive are described in Table 89.

Table 89 – ASLME-VcrDeactive.request primitive parameter definitions

Parameter	Data type	Valid range	Description
PeerAddr	Unsigned16	0 to 65 535	Peer address of the application service request
UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to

10.7.3.4 ASLME-SignalEvent.indication primitive

If there appears or disappears an alarm event in ASL, ASL shall invoke ASLME-SignalEvent.indication primitive to signal its associated UAP.

The definition of ASLME- SignalEvent.indication primitive is as follows.

ASLME-SignalEvent.indication(
PeerAddr,
UAP_ID,
EventFlag,
AppearFlag
)

The parameters of ASLME-SignalEvent.indication primitive are described in Table 90.

Table 90 – ASLME-SignalEvent.request primitive parameter definitions

Parameter	Data type	Valid range	Description
PeerAddr	Unsigned16	0 to 65 535	Peer address of the application service request
UAP_ID	Unsigned16	0 to 65 535	Identifier of the UAP that the service references to
EventFlag	BitField16		Alarm events to be signaled to UAP. Each bit of EventFlag represents one alarm event, see Table 77 for definition.
AppearFlag	BitField16		Each bit of AppearFlag corresponds to the bit of EventFlag, and indicates whether the corresponding alarm event appears or disappears. The coding for each bit is: 0 = DISAPPEAR; 1 = APPEAR.

10.7.4 ASL message format

10.7.4.1 General message format

Each ASL message comprises the following two parts.

- ASL header , including Message control, UAP identifier, and Payload length fields;
- ASL payload with variable length.

The ASL general message format is shown in Figure 100.

ASL header			ASL payload
Message Control	UAP Identifier	PayloadLength	Payload
1 octet	2 octets	2 octets	Variable

IEC

Figure 100 – ASL general message format

10.7.4.2 ASL header

10.7.4.2.1 Message control field

10.7.4.2.1.1 General

The data type of Message control field is Unsigned8, including Service identifier and Message type subfields, shown in Figure 101.

Bits: 6 to 7	Bits: 3 to 5	Bits: 0 to 2
Message Type	Reserved	Service Identifier

IEC

Figure 101 – Format of Message control field

10.7.4.2.1.2 Service identifier subfield

The length of Service Identifier subfield is 3 bits, indicating the AL service types. Its values are shown in Table 91.

Table 91 – Service Identifier subfield definition

Bits: 0 to 2	Meaning
0b001	READ
0b010	WRITE
0b011	PUBLISH
0b100	REPORT
0b101	REPORT ACK
0b110	START
0b111	STOP

10.7.4.2.1.3 Message type subfield

The length of Message Type subfield is 2 bits, indicating the AL message types. Its values are shown in Table 92.

Table 92 – Message Type subfield definition

Bits: 6 to 7	Meaning
0b00	REQUEST
0b01	RESPONSE_P
0b10	RESPONSE_N
0b11	Reserved

10.7.4.2.2 UAPIdentifier field

The data type of UAP Identifier field is Unsigned16, and the value shall be UAP_ID of the UAP that the application service message references to. UAP_ID = 0 is used for DMAP. UAP_ID = 0xFFFF is used to indicate all UAPs on a field device or on all field devices, depending on the destination address.

10.7.4.2.3 Payload Length field

The data type of Payload Length field is Unsigned16, indicating the data length of ASL payload in octet, which excludes ASL header.

10.7.4.3 ASL Payload

Payload field includes the application service message as ASL user data with variable length. Different services define different message formats; see 10.6.2 to 10.6.6 for details.

10.7.4.4 ASL state machines

10.7.4.5 General

According to VCR endpoint role types, the ASL state machines (ASLMs) include Client state machine (AMCL), Server state machine (AMSV), Publisher state machine (AMPB), Subscriber state machine (AMSB), Report source state machine (AMRS), and Report sink state machine (AMRK).

10.7.4.6 Primitives exchanged among UAP, ASL, and DLL

10.7.4.6.1 Confirmed application service primitives

Figure 102 and Table 93 show the primitives for confirmed application services among UAP, ASL, and DLL.

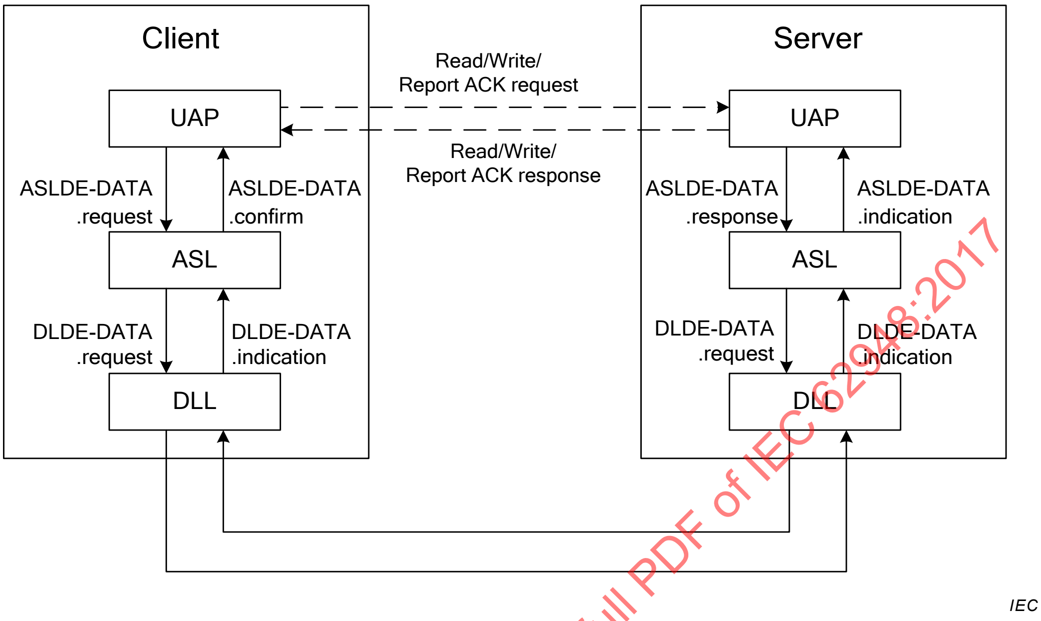


Figure 102 – Confirmed application service primitives among layers

Table 93 – Confirmed service primitives exchanged between ASL and other layers

Primitives	Source	Parameters
ASLDE-DATA.request	UAP	DstAddr, ServiceID, UAP_ID, Priority, AsduLength, Asdu
ASLDE-DATA.indication	ASL	ServiceID, UAP_ID, AsduLength, Asdu
ASLDE-DATA.response	UAP	ServiceID, UAP_ID, AsduLength, Asdu
ASLDE-DATA.confirm	ASL	SrcAddr, ServiceID, MsgType, UAP_ID, AsduLength, Asdu
DLDE-DATA.request	ASL	DstAddr, VCR_ID, DataType, Priority, PayloadLength, Payload
DLDE-DATA.indication	DLL	SrcAddr, DataType, PayloadLength, Payload

10.7.4.6.2 Unconfirmed application service primitives

Figure 103 and Table 94 show the primitives for unconfirmed application services among UAP, ASL, and DLL.

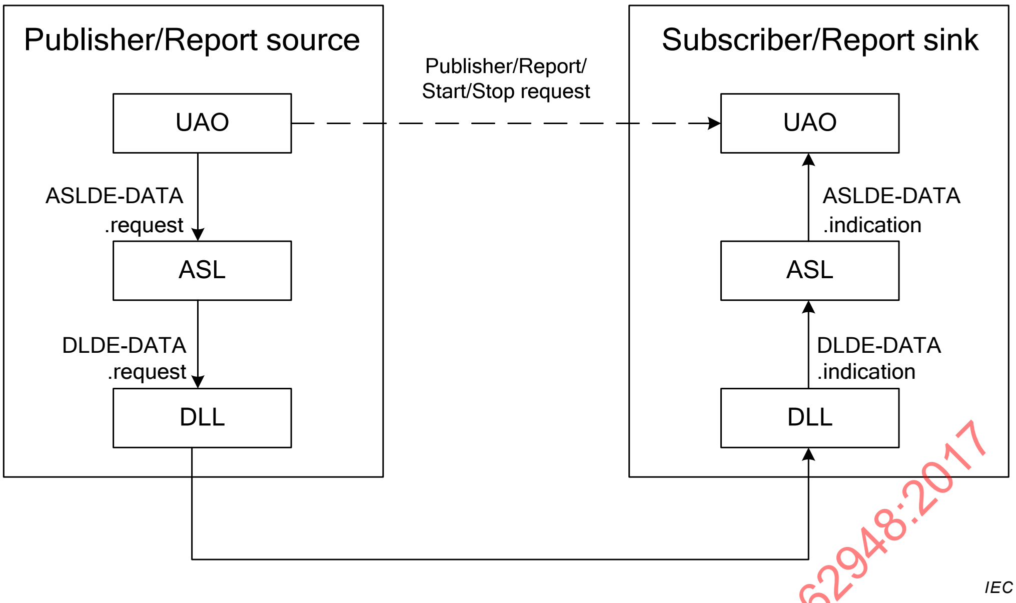


Figure 103 – Unconfirmed application service primitives among layers

Table 94 – Unconfirmed service primitives exchanged between ASL and other layers

Primitives	Source	Parameters
ASLDE-DATA.request	UAP	DstAddr, ServiceID, UAP_ID, Priority, AsduLength, Asdu
ASLDE-DATA.indication	ASL	ServiceID, UAP_ID, AsduLength, Asdu
DLDE-DATA.request	ASL	DstAddr, VCR_ID, DataType, Priority, PayloadLength, Payload
DLDE-DATA.indication	DLL	SrcAddr, DataType, PayloadLength, Payload

10.7.4.7 ASL management service primitives between ASL and UAP

Figure 104 and Table 95 show ASL management service primitives between ASL and UAP.

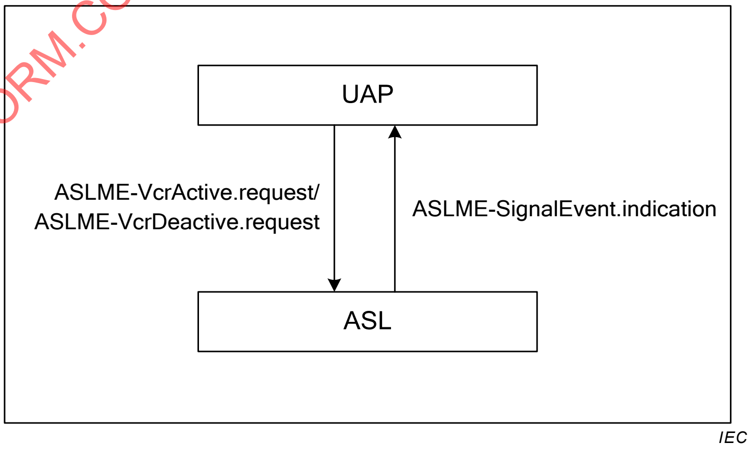


Figure 104 – ASL management service primitives between ASL and UAP

Table 95 – ASL management service primitives between ASL and UAP

Primitives	Source	Parameters
ASLME-VcrActive.request	UAP	PeerAddr, UAP_ID
ASLME-VcrDeactive.request	UAP	PeerAddr, UAP_ID
ASLME-SignalEvent.indication	ASL	PeerAddr, UAP_ID, EventFlag, AppearFlag

10.7.4.8 Client state machine

Each Client VCR endpoint shall maintain a Client state machine (AMCL). The Client state machine (AMCL) shall have the following states.

- Idle state: in this state, the Client VCR endpoint is waiting for ASLDE-DATA.request primitive to deliver a confirmed application service request message. After receiving the primitive, it shall pack the application service request message into an ASL request message by adding ASL header, and invoke DLDE-DATA.request primitive to send the message. Then, the VCR endpoint shall transit to Wait_Cnf state.
- Wait_Cnf state: in this state, the Client VCR endpoint is waiting for the ASL response message returned by the field device. It shall perform one of the following state transitions.
  - If receiving the ASL response message through DLDE-DATA.indication primitive within WatchdogTime interval, the VCR endpoint shall parse the ASL response message and deliver the application service response message to UAP by invoking ASLDE-DATA.confirm primitive. Then, the VCR endpoint shall transit to Idle state.
  - If not receiving the ASL response message within WatchdogTime interval, the VCR endpoint shall return an application service negative response message with “SERVICE_EXPIRATION” to UAP by invoking ASLDE-DATA.confirm primitive. Then, the VCR endpoint shall transit to Idle state.

WatchdogTime is monitored by a WatchdogTime timer, whose value is set to the WatchdogTime attribute of the VCR endpoint.

Figure 105 and Table 96 show the state transitions of the Client state machine.

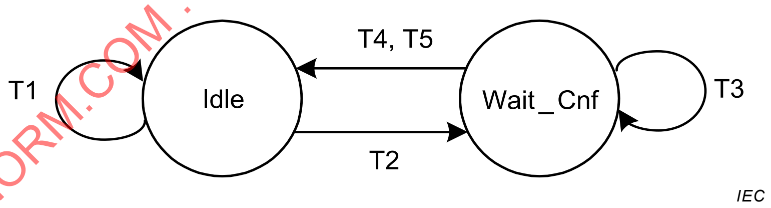


Figure 105 – State transition diagram of AMCL

Table 96 – State transition table of AMCL

#	Current state	Event or condition => action	Next state
T1	Idle	(ASLDE-DATA.request()&& ServiceID != (READ || WRITE || REPORT ACK)) || ASLDE-DATA.response() || DLDE-DATA.indication() => Ignore;	Idle

T2	Idle	ASLDE-DATA.request()&& ServiceID == (READ || WRITE || REPORT ACK)
		=>
		VCR_ID:= GetVcrID(DstAddr, CLIENT, UAP_ID);
		StartWatchdogTimer(VCR_ID);
		StoreSvrID(VCR_ID, ServiceID);
		MsgType:= REQUEST
		DLDE-DATA.request(
		DstAddr;
		VCR_ID,
		DataType:= DATA,
		Priority,
		PayloadLength:= AsduLength + 5,
		Payload:= BuildAPDU(ServiceID, MsgType, UAP_ID, AsduLength, Asdu)
		);
			Wait_Cnf

T3	Wait_Cnf	ASLDE-DATA.request() || ASLDE-DATA.response() || (DLDE-DATA.indication() &&(DataType != DATA || TakeServiceID(Payload) != (READ||WRITE||REPORT ACK) || TakeMsgType(Payload) != (RESPONSE_P || RESPONSE_N))) => Ignore;	Wait_Cnf

T4	Wait_Cnf	DLDE-DATA.indication()
		&&DataType == DATA
		&& TakeServiceID(Payload) ==(READ || WRITE || REPORT ACK)
		&&TakeMsgType(Payload) == (RESPONSE_P || RESPONSE_N)
		=>
		UAP_ID:= TakeUAPID(Payload);
		VCR_ID:= GetVcrID(SrcAddr, CLIENT, UAP_ID);
		StopWatchdogTimer(VCR_ID);
		ASLDE-DATA.confirm(
		SrcAddr,
		ServiceID:= TakeServiceID(Payload),
		MsgType:= TakeMsgType(Payload),
		UAP_ID,
		AsduLength:= PayloadLength – 5,
		Asdu:= TakeASLPayload(PayloadLength, Payload)
		);
			Idle

#	Current state	Event or condition => action	Next state

T5	Wait_Cnf	WatchdogTime r with VCR_ID expires
		=>
		ASLDE-DATA.confirm(
		SrcAddr:= GetPeerAddr(VCR_ID),
		ServiceID:= RestoreSvrID(VCR_ID),
		MsgType:= RESPONSE_N,
		UAP_ID:= GetUAPID(VCR_ID);
		AsduLength:= 2,
		Asdu:= BuildErrAsdu(SERVICE_EXPIRATION, 0)
		);
			Idle

10.7.4.9 Server state machine

Each Server VCR endpoint shall maintain a Client state machine (AMSV). The Server state machine (AMSV) shall have the following states.

- Idle state: in this state, the Server VCR endpoint is waiting for DLDE-DATA.indication primitive to deliver an ASL confirmed request message. After receiving the primitive, the VCR endpoint shall parse the ASL request message, and invoke ASLDE-DATA.indication primitive to deliver the application service request message to UAP. Then, the VCR endpoint shall transit to Wait_Rsp state.
- Wait_Rsp state: in this state, the Server VCR endpoint is waiting for the application service response message returned by UAP. It shall perform one of the following state transitions.
    - If receiving the application service response message through ASLDE-DATA.response primitive within WatchdogTime interval, the VCR endpoint shall pack the application service response message into an ASL response message by adding ASL header, and invoke DLDE-DATA.request primitive to send the ASL response message. Then, the VCR endpoint shall transit to Idle state.
    - If not receiving the application service response message within WatchdogTime interval, the VCR endpoint shall return an application service negative response with “SERVICE_EXPIRATION” to the field device. Then, the VCR endpoint shall transit to Idle state.

WatchdogTime is monitored by a WatchdogTime timer, whose value is set to the WatchdogTime attribute of the VCR endpoint. Figure 106 and Table 97 show the state transitions of the Server state machine.

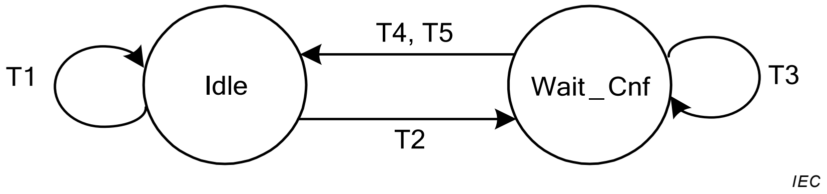


Figure 106 – State transition diagram of AMSV