INTERNATIONAL STANDARD

ISO/IEC 8802-6

ANSI/IEEE Std 802.6

First edition
1994-03-07

Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements —

Part 6:
Distributed Queue Dual Bus (DQDB) access method and physical layer specifications

Technologies de l'information — Communication de données et échange d'informations entre systèmes — Réseaux locaux et métropolitains — Exigences spécifiques —

Partie 6: Bus distribué à double queue (DQDB) et spécifications pour la couche physique

Reference number
ISO/IEC 8802-6:1994(E)
ANSI/IEEE
Std 802.6, 1994 edition

Abstract: This standard is part of a family of standards for local area networks (LANs) and metropolitan area networks (MANs) that deals with the Physical and Data Link Layers as defined by the ISO Open Systems Interconnection Reference Model. It defines a high-speed shared medium access protocol for use over a dual, counterflowing, unidirectional bus subnetwork. The Physical Layer and Distributed Queue Dual Bus (DQDB) Layer are required to support a Logical Link Control (LLC) Sublayer by means of a connectionless Medium Access Control (MAC) Sublayer service in a manner consistent with other IEEE 802 networks. Additional DQDB Layer functions are specified as a framework for other services. These additional functions will support Isochronous Service Users and Connection-Oriented Data Service Users, but their implementation is not required for conformance.

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

ISBN 1-55937-345-8

March 7, 1994 *SH16576*

ISO/IEC 8802-6 : 1994
ANSI/IEEE Std 802.6, 1994 Edition
(incorporating ANSI/IEEE Std 802.6-1990,
IEEE Std 802.6d-1993, and IEEE Std 802.6f-1993)

Information technology— Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements—

Part 6: Distributed Queue Dual Bus (DQDB) access method and physical layer specifications

Sponsor

Technical Committee on Computer Communications
of the
IEEE Computer Society

Adopted as an International Standard by the
International Organization for Standardization
and by the
International Electrotechnical Commission

Published by
The Institute of Electrical and Electronics Engineers, Inc.

American National Standard

International Standard ISO/IEC 8802-6 : 1994

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and nongovernmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 8802-6 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology.*

ISO/IEC 8802 consists of the following parts, under the general title *Information technology—Local and metropolitan area networks:*

— *Part 1: Overview and Architecture*

— *Part 2: Logical link control*

— *Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*

— *Part 4: Token-passing bus access method and physical layer specifications*

— *Part 5: Token ring access method and physical layer specifications*

— *Part 6: Distributed Queue Dual Bus (DQDB) access method and physical layer specifications*

— *Part 7: Slotted ring access method and physical layer specification*

For the purpose of assigning the organizationally unique identifiers that are used to build the 48-bit Medium Access Control (MAC) addresses, the Institute of Electrical and Electronics Engineers, Inc., USA, has been designated by the ISO and IEC Councils as the Registration Authority. Communications on this subject should be addressed to

Registration Authority for ISO/IEC 8802-6
c/o The Institute of Electrical and Electronics Engineers, Inc.
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

During the preparation of the International Standard, information was gathered on patents upon which application of the standard might depend. Relevant patents were identified as belonging to the American Telephone and Telegraph Company (AT&T) and QPSX Communications Ltd. However, ISO and IEC cannot give authoritative or comprehensive information about evidence, validity or scope of patent and like rights. The patent-holder has stated that licenses will be granted under reasonable terms and conditions. Communications on this subject should be addressed to

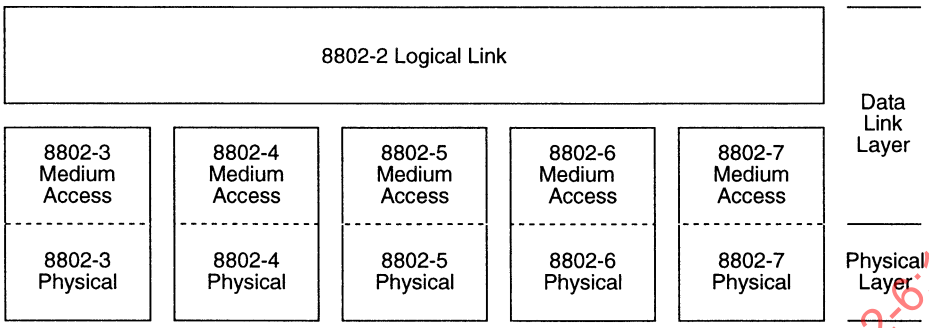
AT&T
Crawfords Corner Road
P.O. Box 3030
Holmdel, NJ 07733-3030
USA

QPSX Communications Ltd.
Perth, Western Australia

International Organization for Standardization/International Electrotechnical Commission
Case postale 56 • CH-1211 Genève 20 • Switzerland

Foreword to International Standard ISO/IEC 8802-6 : 1994

This standard is part of a family of standards for Local and Metropolitan Area Networks. The relationship between this standard and the other members of the family is shown below. (The numbers in the figure refer to ISO standard numbers.)



This family of standards deals with the Physical and Data Link layers as defined by the ISO Open Systems Interconnection Basic Reference Model (ISO 7498 : 1984). The access standards define five types of medium access technologies and associated physical media, each appropriate for particular applications or system objectives. Other types are under investigation.

The standards defining the access technologies are as follows:

a) ISO/IEC 8802-3 [ANSI/IEEE Std 802.3, 1993 Edition], a bus utilizing CSMA/CD as the access method.
b) ISO/IEC 8802-4 [ANSI/IEEE Std 802.4-1990], a bus utilizing token passing as the access method.
c) ISO/IEC 8802-5 [ANSI/IEEE Std 802.5-1992], a ring utilizing token passing as the access method.
d) ISO/IEC 8802-6 [ANSI/IEEE Std 802.6, 1994 Edition], a dual bus utilizing distributed queuing as the access method. DQDB subnetworks provide a range of telecommunications services within a metropolitan area.
e) ISO 8802-7, a ring utilizing slotted ring as the access method.

ISO 8802-2 [ANSI/IEEE Std 802.2-1989], *Logical Link Control protocol*, provides for data transfer between medium access standards and network layer protocol.

ISO/IEC 10038 [ANSI/IEEE Std 802.1D, 1993 Edition], *Media access control (MAC) bridges*, specifies an architecture and protocol for the interconnection of IEEE 802 LANs below the level of the logical link control protocol.

The reader of this document is urged to become familiar with the complete family of standards.

ANSI/IEEE Std 802.6, 1994 Edition

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE which have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least once every five years for revision or reaffirmation. When a document is more than five years old, and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

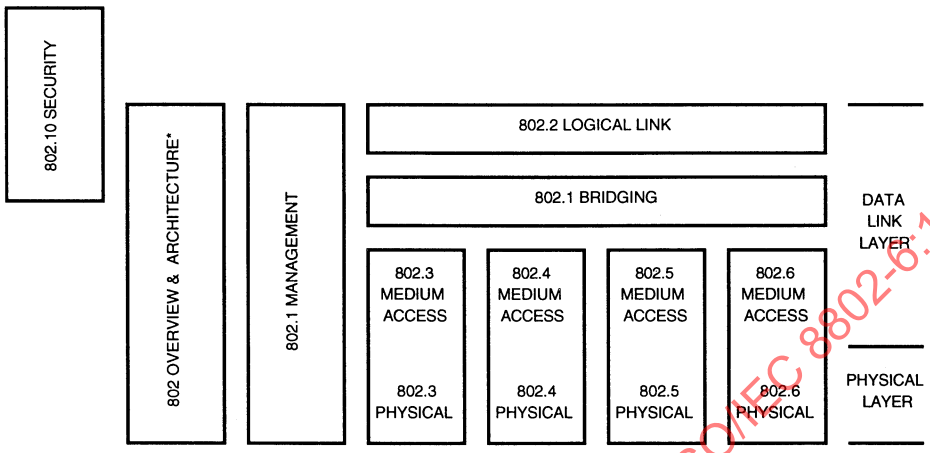
Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Introduction

(This introduction is not a part of ANSI/IEEE Std 802.6, 1994 Edition or of ISO/IEC 8802-6 : 1994.)

This standard is part of a family of standards for local and metropolitan area networks. The relationship between the standard and other members of the family is shown below. (The numbers in the figure refer to IEEE standard numbers.)



* Formerly IEEE Std 802.1A.

This family of standards deals with the Physical and Data Link layers as defined by the International Organization for Standardization (ISO) Open Systems Interconnection Basic Reference Model (ISO 7498 : 1984). The access standards define several types of medium access technologies and associated physical media, each appropriate for particular applications or system objectives. Other types are under investigation.

The standards defining these technologies are as follows:

- IEEE Std 802[1]: Overview and Architecture. This standard provides an overview to the family of IEEE 802 Standards. This standard forms part of the 802.1 scope of work.

- IEEE Std 802.1B [ISO DIS 15802-2]: LAN/MAN Management. Defines an Open Systems Interconnection (OSI) management-compatible architecture, and service and protocol elements for use in a LAN/MAN environment for performing remote management.

- ISO/IEC 10038 [ANSI/IEEE Std 802.1D]: MAC Bridging. Specifies an architecture and protocol for the interconnection of IEEE 802 LANs below the MAC service boundary.

- IEEE Std 802.1E [ISO DIS 15802-4]: System Load Protocol. Specifies a set of services and protocol for those aspects of management concerned with the loading of systems on IEEE 802 LANs.

[1]The 802 Architecture and Overview Specification, originally known as IEEE Std 802.1A, has been renumbered as IEEE Std 802. This has been done to accommodate recognition of the base standard in a family of standards. References to IEEE Std 802.1A should be considered as references to IEEE Std 802.

- ISO 8802-2 [ANSI/IEEE Std 802.2]: Logical Link Control
- ISO/IEC 8802-3 [ANSI/IEEE Std 802.3]: CSMA/CD Access Method and Physical Layer Specifications
- ISO/IEC 8802-4 [ANSI/IEEE Std 802.4]: Token Bus Access Method and Physical Layer Specifications
- ISO/IEC 8802-5 [ANSI/IEEE Std 802.5]: Token Ring Access Method and Physical Layer Specifications
- ISO/IEC 8802-6 [ANSI/IEEE Std 802.6]: Distributed Queue Dual Bus (DQDB) Access Method and Physical Layer Specifications
- IEEE Std 802.10: Interoperable Local Area Network (LAN) Security, *Currently Contains* Secure Data Exchange (SDE)

In addition to the family of standards, the following is a recommended practice for a common technology:

- IEEE Std 802.7: IEEE Recommended Practice for Broadband Local Area Networks

The reader of this standard is urged to become familiar with the complete family of standards.

Conformance test methodology

An additional standards series, identified by the number 1802, has been established to identify the conformance test methodology documents for the 802 family of standards. This makes the correspondence between the various 802 standards and their applicable conformance test requirements readily apparent. Thus the conformance test documents for 802.3 are numbered 1802.3, the conformance test documents for 802.5 will be 1802.5, and so on. Similarly, ISO will use 18802 to number conformance test standards for 8802 standards.

IEEE Std 802.6, 1994 Edition

The purpose of this standard is to lay the foundation for a set of standards to allow DQDB subnetworks to provide a range of telecommunications services within a metropolitan area. Based on this set of standards, equipment vendors will be able to build the components that will allow telecommunications services to be offered to end users within a metropolitan area.

The interconnection of DQDB subnetworks to form a metropolitan area network will be possible via a Multiport Bridge[2] or via dual-port bridges, routers, and gateways, as shown in Fig A. DQDB subnetworks can be used to provide switching, routing, and concentration of high-speed data, voice, and certain video services, as well as to interconnect LANs, hosts, workstations, and PBXs.

The user's attention is called to the possibility that compliance with this standard may require the use of inventions covered by patent rights. All enquiries should be sent to the IEEE Standards Board at the address given at the end of the introduction.

This edition of the standard defines general principles for the operation of a number of physical layers. A physical layer capable of using a DS3 transmission system (ANSI T1.102 and T1.107) is defined in this edition of the standard. It is anticipated that future editions of the standard will provide additional implementations of the physical layer to support different needs (for example, media, data rates, network operator requirements).

[2]The services of a Multiport Bridge are the subject of ongoing work under IEEE Project Authorization P802.6a, Multiple Port Bridging for Metropolitan Area Networks.

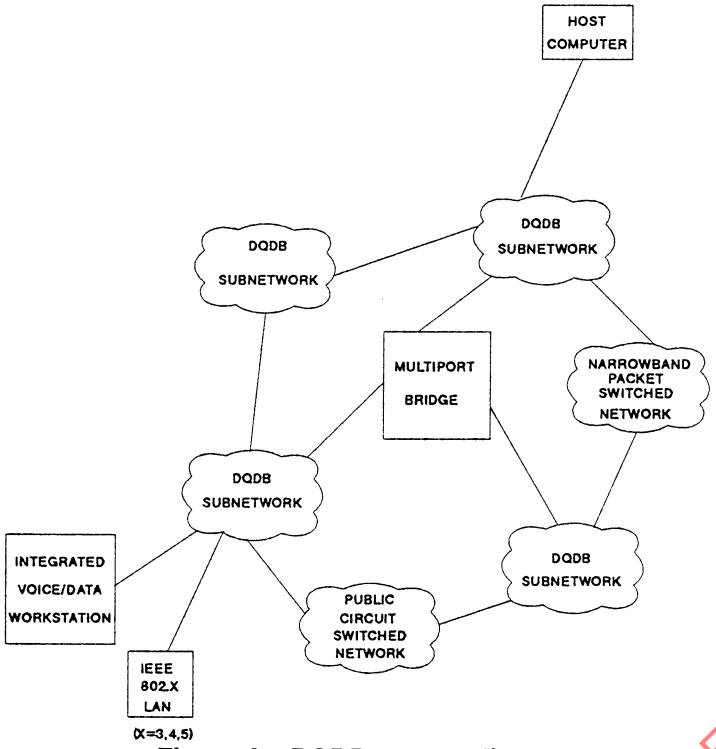


Figure A—DQDB metropolitan area network

This edition of the standard incorporates IEEE Std 802.6d-1993, *Physical Layer Convergence Procedure (PLCP) for CCITT Recommendations G.707, G.708, and G.709 SDH-based systems (155.520 Mbit/s)*, as clause 17. It also incorporates IEEE Std 802.6f-1993, *Protocol Implementation Conformance Statement (PICS) proforma*, as annex A (normative). The two standards are not available as separate publications.

Three additional IEEE 802.6 standards are currently available. These include the following:

- IEEE Std 802.6k-1992, Supplement to Media Access Control (MAC) Bridges (IEEE Std 802.1D-1990): IEEE 802.6 Distributed Queue Dual Bus (DQDB) Subnetwork of a Metropolitan Area Network (MAN)
- IEEE Std 802.6c-1993, Supplement to ISO/IEC 8802-6: Physical Layer Convergence Procedure (PLCP) for DS1-based systems (clause 12) *packaged with IEEE Std 802.6h-1993 (see below)*
- IEEE Std 802.6h-1993, Supplement to ISO/IEC 8802-6: Isochronous Service on a DQDB Subnetwork of a MAN

This standard contains state-of-the-art material. The area covered by this standard is undergoing evolution. Revisions are anticipated within the next few years to clarify existing material, to correct possible errors, and to incorporate new related material. Information on the current revision state of this and other IEEE 802 standards may be obtained from

Secretary, IEEE Standards Board
P.O. Box 1331
445 Hoes Lane
Piscataway, NJ 08855-1331, USA

IEEE 802 committee working documents are available from

IEEE Document Distribution Service
AlphaGraphics #35 Attn: P. Thrush
10201 N. 35th Avenue
Phoenix, AZ 85051, USA

Participants

When IEEE Std 802.6-1990 was approved, the IEEE 802.6 Working Group had the following voting membership:

James F. Mollenauer, *Chair*

Steven Agard
Per-Ola Andersson
Albert Azzam
Pat Baker
Kim Banker
Brian Bean
Ernest Bergmann
Terry Boston
Marc Boullet
Richard Brandwein
Richard Breault
Zee Brun
Zigmantas Budrikis
Robert Chancer
Julian Cheesman
Ben-Ren Chen
Frank Chen
Hon Wah Chin
Antonino Ciccardi
George Clapp
Ken Coit
Paul Cowell
Tracy Cox
James Dahl
Ravi Damodaram
Felix Diaz
James Doak
Subra Dravida
Arthur Drott
Cedric Druce
Gus Elmer
Peter Evans
Paul Frantz
Ingrid Fromm
Sumitra Ganguly
Raymond Gass
Ed Geiger
Sayeed Ghani
Moe Ghassami
Neville Golding
Bernard Grandjean
Peter Griffiths
Del Hanson
Takeshi Harakawa
David Harris
Chris Hemrick
Y. Jayachandra
Vijay Kapoor
Salil Karr
Jari Karttunen
Dermot Kavanagh
Paul Kellam
Olavi Keranen
Robert Klessig
Demosthenes Kostas
John Lattyak
My Le
Roger Levy
Morgan Littlewood
Jerry Mendes
Methi Methiwalla
Hamid Modarressi
Joel Morrow
Shigeo Nakagawa
Mehdi Nassehi
Robert Newman
Seppo Noppari
Stanley Ooi
David Oster
Jean Ouss
Roger Pandanda
Achilles Perdikaris
Al Pereira
Roy Perry
Douglas Place
Philip Potter
P. K. Prasanna
Russ Pretty
Vikram Punj
Harvey Rubin
Steve Russell
John Salter
Paul Sanchirico
Lars Sandstrom
John Schembri
Ray Schnitzler
Nasser Sharabianlou
George Shenoda
Shoji Soejima
Michael Spratt
David Swardstrom
John Swenson
Ahmed Tantawy
Wayne Tsou
Kent Tsuno
Harmen Van As
Surya Varanasi
Jim Wallim
Richard Weadon
Graham Williams
Tetsuya Yokotani

The working group is especially grateful for the dedicated work of Peter Evans and Sayeed Ghani, editors of IEEE Std 802.6-1990, who translated the complexities of a protocol involving many thousands of silicon gates into an accurate and usable document. Particular thanks are due also to Dan Sze, who authored several proposals in earlier stages of the MAN standard development and also served as vice chair of the committee; to Mary Kelly, whose work behind the scenes helped to make it all come together.

The following persons were on the balloting committee of IEEE Std 802.6-1990:

Bandula W. Abeysundara
William B. Adams
Jonathan Allan
Robert M. Amy
Kit Athul
William E. Ayen
Yong Myung Baeg
Ali Bahroloomi
Tim Batten
Michael Benedek
Peter Evans
Changxin Fan
John W. Fendrich
Harold C. Folts
Harvey A. Freeman
Ingrid Fromm
Eithan Froumine
Gordon Fullerton
Robert Gagliano
Isaac Ghansah
Patrick Gonia
Maris Graube
Sandor V. Halasz
Joseph L. Hammond
R. Harish
J. Scott Haugdahl
Kenneth C. Heck
Lee A. Hollaar
Marsha D. Hopwood
Anne B. Horton
Bob Jacobsen
Raj Jain
Anura P. Jayasumana

Reijo Juvonen
Richard H. Karpinski
Gary C. Kessler
Dae Young Kim
Robert W. Klessig
Jens Kolind
Peter Kornerup
Bruce R. Kravitz
Stephen B. Kruger
Thomas M. Kurihara
Anthony B. Lake
Alan L. Bridges
Richard Caasi
Mehmet U. Caglayan
Paul W. Campbell
George Carson
Brian J. Casey
George C. Chachis
Elisardo M. R. Chatruc
Kilnam Chon
Ricardo Ciciliani
Lak Ming Lam
Michael Lawler
John E. Lecky
Jaiyong Lee
Lewis E. Leinenweber
F. C. Lim
Randolph S. Little
M. T. Liu
William D. Livingston
Mauro Lolli
Donald C. Loughry
Gottfried W. R. Luderer
Peter Martini
Richard H. Miller
David S. Millman
C. B. M. Mishra
Wen Hsien Lim Moh
John E. Montague
Kinji Mori
Gerald Moseley
Ruth Nelson
Arne A. Nilsson
Donal O'Mahony
Charles Oestereicher
Jeffrey L. Paige
Andreas Pfitzmann
Rafat Pirzada
Urdo W. Pooch
Vikram Prabhu
Andris Putnins
Sudha Ram
Thad L. D. Regulinski
John P. Riganati
Michael H. Coden
Robert Crowder
Jose A. Cueto
Mark S. Day
Ashwani K. Dhawan
Michel Diaz
Mitchell G. Duncan
John E. Emrich
Philip H. Enslow
Richard G. Estock
Gary S. Robinson
Philip T. Robinson
Edouard Y. Rocher
Floyd E. Ross
Victor Rozentouler
Chiseki Sagawa
Mark S. Sanders
Julio Gonzalez Sanz
Ambatipudi R. Sastry
Norman Schneidewind
Jeffrey R. Schwab
Arthur J. Scialabba
Donald A. Sheppard
Ing. Alex. Soceanu
Charles Spurgeon
Fred J. Strauss
Efstathios D. Sykas
Daniel Sze
Nhi P. Ta
Hao Tang
Ahmed N. Tantawy
James N. Thomas
Robert Tripi
A. von Mayrhauser
James T. Vorhies
Donald F. Weir
Alan J. Weissberger
Raymond Wenig
Earl J. Whitaker
Paul A. Willis
George B. Wright
Oren Yuen
William H. Yundt
Wei Zhao

When the IEEE Standards Board approved IEEE Std 802.6-1990 on December 6, 1990, it had the following membership:

Marco W. Migliaro, *Chair* **Donald C. Loughry,** *Vice Chair*
Andrew G. Salem, *Secretary*

Dennis Bodson
Paul L. Borrill
Fletcher J. Buckley
Allen L. Clapp
Stephen R. Dillon
Donald C. Fleckenstein
Jay Forster*
Thomas L. Hannan
Kenneth D. Hendrix
John W. Horch
Joseph L. Koepfinger*
Irving Kolodny
Michael A. Lawler
Donald C. Loughry
John E. May, Jr.
Lawrence V. McCall
L. Bruce McClung
Donald T. Michael*
Stig Nilsson
Roy T. Oishi
Gary S. Robinson
Terrance R. Whittemore
Donald W. Zipse

*Member Emeritus

Paula M. Kelty
IEEE Standards Project Editor

IEEE Std 802.6-1990 was approved by the American National Standards Institute (ANSI) on May 30, 1991.

When IEEE Std 802.6d-1993 and IEEE Std 802.6f-1993 were approved, the IEEE 802.6 Working Group had the following voting membership:

James F. Mollenauer, *Chair*

Steve Agard
Sven Akerlund
Al Azzam
Pat Baker
Kim Banker
Terry Boston
Doug Brady
Rich Brandwein
Richard Breault
Chuck Brill
Zig Budrikis
Graham Campbell
Hon Wah Chin
Tony Ciccardi
George Clapp
Ken Coit
Paul Cowell
Tracy Cox
Jim Dahl
Ravi Damodaram
Gus Elmer
Peter Evans
Mark Fine
Paul Franz
Ingrid Fromm
Raymond Gass
Sumitra Ganguly
Ed Geiger
Sayeed Ghani
Suvankar Ghosh
Jay Gill
Neville Golding
Bernard Grandjean
Peter Griffiths
Takeshi Harakawa
Hossam Hassanein
Chris Hemrick
Gunter Horn
Hajime Inoue
Y. Jayachandra
Vijay Kapoor
Sal Karr
Jari Karttunen
Dermot Kavanagh
Olavi Keranen
Bob Klessig
Demos Kostas
Roger Levy
Morgan Littlewood
Peter Martini
Jerry Mendes
Methi Methiwalla
Hamid Modaressi
Joel Morrow
Dave Oster
Jean Ouss
Al Pereira
Egidio Perretti
Roy Perry
Doug Place
Vikram Punj
Steve Russell
John Salter
Ray Schnitzler
Nasser Sharabianlou
George Shenoda
Jean-Louis Simon
Shoji Soejima
Michael Spratt
David Stacy
Dave Swardstrom
John Swenson
Laszlo Szerenyi
Ahmed Tantawy
Carlos Tomaszewski
Wayne Tsou
Kent Tsuno
Surya Varanasi
Jim Wallin
George Wayne
Dick Weadon
Al Weissberger
Graham Williams
Tetsuya Yokotani

The following persons were on the balloting committee that voted on IEEE Std 802.6d-1993 and IEEE Std 802.6f-1993:

Robert M. Amy
Per-Ola Andersson
Michael Benedek
Champa Bhushan
Paul W. Campbell
George C. Chachis
Michael H. Coden
Robert Crowder
John E. Emrich
Peter Evans
John W. Fendrich
Harvey A. Freeman
Ingrid Fromm
Isaac Ghansah
Julio Gonzalez-Sanz
Kenneth C. Heck
Raj Jain
Anura P. Jayasumana
Reijo Juvonen
Gary C. Kessler
Dae Young Kim
Robert W. Klessig
Demosthenes Kostas
Bruce Kravitz
Jai Yong Lee
Randolph S. Little
Ming T. Liu
Donald C. Loughry
Gottfried W. R. Luderer
Peter Martini
David S. Millman
James Mollenauer
Kinji Mori
Donal O'Mahony
Udo W. Pooch
Vikram Punj
Andris Putnins
Thad L. D. Regulinski
Gary S. Robinson
Edouard Y. Rocher
Victor Rozentouler
Norman Schneidewind
Efstathiois D. Sykas
Ahmed N. Tantawy
Robert Tripi
James T. Vorhies
Alan J. Weissberger
Paul A. Willis
Oren Yuen

The final conditions for approval of IEEE Std 802.6d-1993 and IEEE Std 802.6f-1993 were met on July 12, 1993. These standards were conditionally approved by the IEEE Standards Board on June 17, 1993, with the following membership:

Wallace S. Read, *Chair* **Donald C. Loughry,** *Vice Chair*

Andrew G. Salem, *Secretary*

Gilles A. Baril	Jim Isaak	Don T. Michael*
José A. Berrios de la Paz	Ben C. Johnson	Marco W. Migliaro
Clyde R. Camp	Walter J. Karplus	L. John Rankine
Donald C. Fleckenstein	Lorraine C. Kevra	Arthur K. Reilly
Jay Forster*	E. G. "Al" Kiener	Ronald H. Reimer
David F. Franklin	Ivor N. Knight	Gary S. Robinson
Ramiro Garcia	Joseph L. Koepfinger*	Leonard L. Tripp
Donald N. Heirman	D. N. "Jim" Logothetis	Donald W. Zipse

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
James Beall
Richard B. Engelman
David E. Soffrin
Stanley I. Warshaw

Kristin M. Dittmann
Adam Sicker
IEEE Standards Project Editors

Peter Evans was the first technical editor of the International Standard (ISO/IEC 8802-6 : 1994). Ingrid Fromm succeeded him in 1993 and saw the document through publication. Kristin Dittmann was the IEEE Standards Project Editor.

Contents

CLAUSE PAGE

Information technology—
Telecommunications and information exchange between systems—
Local and metropolitan area networks—
Specific requirements—

Part 6: Distributed Queue Dual Bus (DQDB) access method and physical layer specifications

1. Overview

1.1 Scope

This part of ISO/IEC 8802 for a Distributed Queue Dual Bus (DQDB) subnetwork of a metropolitan area network (MAN) defines a high-speed shared medium access protocol for use over a dual, counter-flowing, unidirectional bus subnetwork. This International Standard specifies the Physical Layer and DQDB Layer required to support

— *Logical Link Control (LLC) Sublayer,* by a connectionless Medium Access Control (MAC) Sublayer service provided to support an LLC Sublayer in a manner consistent with other parts of this ISO/IEC 8802 series of International Standards.

This part of ISO/IEC 8802 also specifies additional DQDB Layer functions as a framework for other services. In this edition of the standard, they are provided for completeness as part of the overall architecture; their implementation is not required for conformance.

These additional functions will support

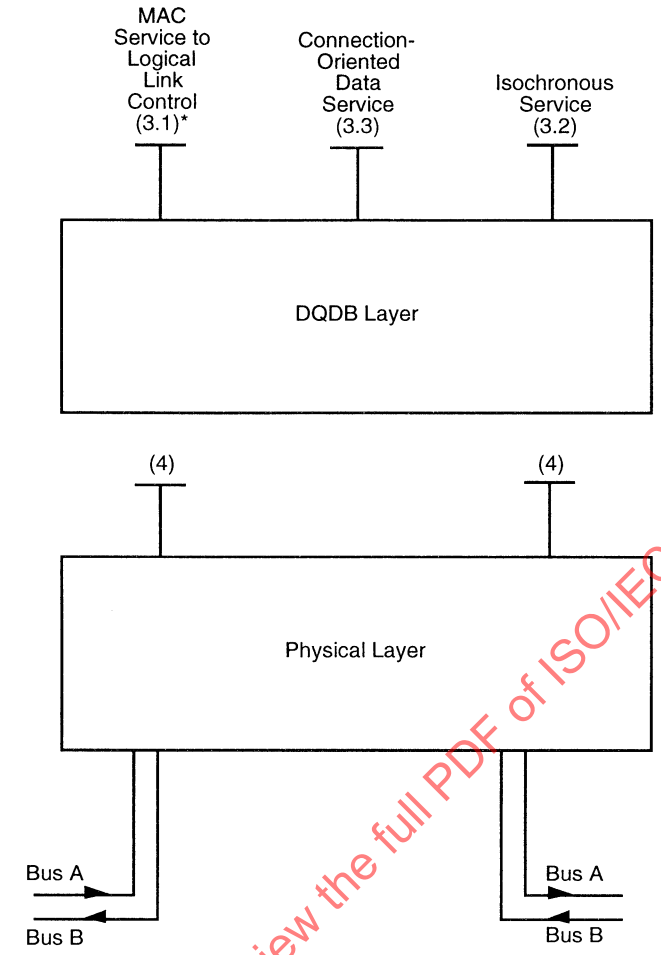
— *Isochronous Service Users (ISUs),* by a connection-oriented service, which may be used to transport isochronous data; for example, conventional digitized voice.
— *Connection-Oriented Data Service Users,* by an asynchronous, connection-oriented service, which may be used to transport bursty data; for example, signaling.

The scope of this part of ISO/IEC 8802 is shown diagrammatically in figure 1-1.

The DQDB Layer supports these services by employing the following access methods:

— A *Queued Arbitrated access method*, with three priority queues for medium access arbitration and fixed-length slots for data transfer. Each priority level provides distributed queue access for the support of connectionless MAC service and connection-oriented data service.
— A *Pre-Arbitrated access method*, which uses assigned octet positions in particular slots for the transfer of individual octets of data. This access method supports isochronous connection-oriented services.

The Queued Arbitrated access method uses fixed-length slots for data transfer. This transfer mechanism is enhanced by a MAC convergence function to the service expected by the LLC Sublayer. The MAC convergence function is defined in this part of ISO/IEC 8802.



*Numbers in parentheses refer to clauses and subclauses of this part of ISO/IEC 8802 where the services are defined.

Figure 1-1—Scope of this part of ISO/IEC 8802

This part of ISO/IEC 8802 defines the DQDB Layer functions to allow access to the medium to read and write data to support the isochronous service and the connection-oriented data service. The signaling procedures for establishing, maintaining, and clearing a connection are outside the scope of this part of ISO/IEC 8802.[1]

The Physical Layer is defined to allow the use of different underlying transmission systems. This International Standard supports a DS3 interface operating at 44.736 Mbit/s, as specified in ANSI T1.102[2] and T1.107. It also supports transmission systems based on CCITT Recommendation G.703 operating at 2.048 Mbit/s, 34.368 Mbit/s, and 139.264 Mbit/s, and Synchronous Digital Hierarchy (SDH) transmission systems operating at 155.520 Mbit/s, as specified in CCITT Recommendations G.707, G.708, and G.709. Other appropriate media and transmission rates may be considered for standardization in the future.

[1] Annex B gives a tutorial description of the sequence of events that are required to support complete isochronous communication capability.

[2] Information on references can be found in 1.3.

Each transmission system may need to be enhanced to provide the Physical Layer service that is required by the DQDB Layer. This enhancement function, if required, is provided by a suitable Physical Layer Convergence Procedure (PLCP). This part of ISO/IEC 8802 will define a PLCP for each transmission system that is supported.

1.2 Applicability

DQDB subnetworks are intended to support a range of service types including statistical data traffic. Subnetworks should therefore be dimensioned by the network provider so as to ensure that periods of expected peak demands do not cause excessive delay to the particular services that are intended to be supported.

During the periods of overload, which may still occur, a DQDB subnetwork will continue to operate at close to maximum throughput, but some service degradation must necessarily occur. To ensure manageable degradation of the services supported by a DQDB subnetwork, three mechanisms have been provided:

— Pre-Arbitrated access
— Queued Arbitrated access with priority
— Bandwidth balancing

A description of these mechanisms is provided in 2.1.2, and their areas of application are outlined in 2.1.4.

1.3 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8802. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 8802 are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ANSI T1.102-1987, American National Standard for Telecommunication—Digital Hierarchy—Electrical Interfaces.[3]

ANSI T1.107-1988, American National Standard for Telecommunications—Digital Hierarchy—Formats Specifications.

ANSI T1.107a-1990, American National Standard for Telecommunications—Digital Hierarchy—Supplement to Formats Specifications (DS3 Format Applications).

CCITT Recommendation E.164 (1988), Numbering plan for the ISDN era. In vol. II, fascicle II.2 of the *CCITT Blue Books*—Telephone network and ISDN—Operation, numbering, routing and mobile service.[4]

CCITT Recommendation G.703 (1988), Physical/Electrical characteristics of hierarchical digital interfaces. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

CCITT Recommendation G.704 (1988), Synchronous frame structures used at primary and secondary hierarchical levels. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

[3] ANSI publications are available from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

[4] CCITT Recommendations are available from the Telecommunication Standardization Bureau of the International Telecommunication Union, Place des Nations, CH-1211, Geneva 20, Switzerland.

CCITT Recommendation G.707 (1988), Synchronous digital hierarchy bit rates. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

CCITT Recommendation G.708 (1988), Network node interface for the synchronous digital hierarchy. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

CCITT Recommendation G.709 (1988), Synchronous multiplexing structure. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

CCITT Recommendation G.751 (1988), Digital multiplexing equipments operating at the third order bit rate of 34 368 kbit/s and the fourth order bit rate of 139 264 kbit/s and the fourth order bit rate of 139 264 kbit/s and using positive justification. In vol. III, fascicle III.4 of the *CCITT Blue Books*—General aspects of digital transmission systems; terminal equipments.

CCITT Recommendation G.783 (1990), Characteristics of Synchronous Digital Hierarchy (SDH) multiplexing equipment functional blocks.

CCITT Recommendation I.432 (1990), B-ISDN User-Network Interface—physical layer specification.

ISO 2382-25 : 1992, Information technology—Vocabulary—Part 25: Local area networks.[5]

ISO DIS 2382-26...,[6] Information technology—Vocabulary—Part 26: OSI architecture.

ISO 7498 : 1984, Information processing systems—Open Systems Interconnection—Basic Reference Model.

ISO/TR 8509 : 1987, Information processing systems—Open Systems Interconnection—Service conventions.

ISO 8802-2 : 1989 [ANSI/IEEE Std 802.2-1989], Information processing systems—Local area networks—Part 2: Logical link control.

ISO/IEC 9595 : 1990, Information processing systems—Open Systems Interconnection—Common management information service definition.

ISO/IEC 9596 : 1990, Information processing systems—Open Systems Interconnection—Common management information protocol specification.

ISO/IEC 10039 : 1991, Information technology—Open Systems Interconnection—Local area networks—Medium Access Control (MAC) service definition.

[5]ISO and ISO/IEC publications are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/Suisse. ISO publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

[6]Current at state of Draft International Standard.

1.4 Definitions

This clause defines the terms used in this part of ISO/IEC 8802. Words in *italics* indicate terms that are defined elsewhere in the list of definitions.

1.4.1 Access Control Field (ACF): The *protocol control information* in a *slot*, which is used to support the *access control function*.

1.4.2 access control function: The generic name for the *Queued Arbitrated (QA) Access* and *Pre-Arbitrated (PA) Access functions* in the *DQDB Layer* that control access to the medium in this part of ISO/IEC 8802.

1.4.3 access unit (AU): The functional unit in a *node* that performs the *DQDB Layer* functions to control access to both *buses*. Access units attach to each bus via a *write* connection and a *read* tap placed *upstream* of the write connection.

1.4.4 address: An identifier that tells where a *service access point (SAP)* may be found (ISO 7498).

1.4.5 address field: The part of a *protocol data unit (PDU)* that contains an *address* that identifies one or more addressable entities. (The address may be a single-source address, single-destination address, or multiple-destination address [*multicast*].)

1.4.6 bandwidth balancing mechanism: A procedure to facilitate effective sharing of the bandwidth, whereby a *node* occasionally skips the use of *empty Queued Arbitrated (QA) slots*.

1.4.7 bridge: A functional unit that interconnects two subnetworks that use a single *Logical Link Control (LLC) procedure* but may use different *Medium Access Control (MAC) procedures*. *Local area networks (LANs)* and *metropolitan area networks (MANs)* are examples of the subnetworks that a bridge may interconnect.

1.4.8 broadcast address: A predefined destination *address* that denotes the set of all *service access points (SAPs)* within a given *layer*.

1.4.9 bus: The concatenation of the *transmission links* between *nodes* and the data path within nodes that provides unidirectional transport of the digital bit stream from the *Head of Bus function* past the *access unit (AU)* of each node to the end of bus.

NOTE—This differs from the bidirectional bus as used in ISO/IEC 8802-3 (Carrier Sense Multiple Access with Collision Detection [CSMA/CD] and ISO/IEC 8802-4 [Token Bus]).

1.4.10 busy slot: A *slot* that contains information and is not available for *Queued Arbitrated (QA) access*.

1.4.11 Configuration Control function: The function that ensures that the resources of all *nodes* of a DQDB *subnetwork* are configured into a correct Dual Bus topology. The resources that are managed are the *Head of Bus function*, the *External Timing Source function*, and the *Default Slot Generator function*.

1.4.12 connection: An association established by a *layer* between two or more users of the layer service for the transfer of information (ISO 7498).

1.4.13 convergence function: A function or procedure that provides sufficient additional services to enable a *layer* or *sublayer* to provide the services expected by a particular higher layer user. (For example, the MAC Convergence Function enables the capabilitites of the *Queued Arbitrated access function* to be enhanced to provide the *Medium Access Control (MAC) Sublayer* service to the *Logical Link Control Sublayer*.)

1.4.14 Data Link Layer: In Open Systems Interconnection (OSI) architecture, the *layer* that provides services to transfer data over a *transmission link* between open systems.

In the ISO/IEC series of *local area network (LAN)* standards, the Data Link Layer is formed by the *Logical Link Control (LLC) Sublayer* and the *Medium Access Control (MAC) Sublayer.*

Using this part of ISO/IEC 8802, the Data Link Layer is formed by the operation of the LLC Sublayer (ISO 8802-2) over the *Medium Access Control (MAC)* Sublayer service offered by the *DQDB Layer.*

1.4.15 Default Slot Generator function: The function that defines the identity (i.e., Bus A or Bus B) for each *bus* of a *Dual Bus subnetwork*. Additionally, the function that provides *Head of Bus functions* for both Bus A and Bus B in a *looped Dual Bus* subnetwork. (If the looped Dual Bus subnetwork is reconfigured to an *open Dual Bus* subnetwork due to *transmission link* faults, then the Head of Bus functions for either one or both buses are assigned to other nodes for the duration of the fault.[7])

1.4.16 Derived MAC Protocol Data Unit (DMPDU): The *Protocol Data Units (PDUs)* of a length of 48 octets formed by the addition of *protocol control information* (including *message identifier* and error protection information) to each of the 44-octet *segmentation units* created from the *segmentation* of an *Initial MAC Protocol Data Unit (IMPDU)*. Each DMPDU is carried as the payload of a *Queued Arbitrated (QA) segment*.

1.4.17 Distributed Queue: The *Medium Access Control (MAC) procedure* in this part of ISO/IEC 8802 for *Queued Arbitrated (QA) access*.

1.4.18 downstream: The direction of data flow along a *bus*, i.e., away from the *Head of Bus function*.

1.4.19 DQDB Layer: The *sublayer* in this part of ISO/IEC 8802 that uses the services of the *Physical Layer* to provide the following:

— *Medium Access Control (MAC) Sublayer* service to the *Logical Link Control (LLC) Sublayer,* and
— *isochronous* service, and
— connection-oriented data service.

1.4.20 Dual Bus: A pair of *buses* carrying digital bit streams flowing in opposite directions. One bus is referred to as Bus A and the other bus as Bus B.

1.4.21 empty Queued Arbitrated (QA) slot: A *Queued Arbitrated (QA) slot* that was designated by the *Head of Bus function* as being available for transfer of a *QA segment*, and that does not contain a QA segment.

1.4.22 External Timing Source function: The function of providing the primary point of synchronization of the DQDB *subnetwork* to some external timing reference, for example, that provided by a public network operator.

1.4.23 gateway: A functional unit that interconnects a *local area network (LAN)* with another network having different higher layer protocols.

1.4.24 group address: A predefined destination *address* that denotes a set of selected *service access points (SAPs)* from the *Medium Access Control (MAC) Sublayer* service offered by the *DQDB Layer* to the *Logical Link Control (LLC) Sublayer.*

[7] Note that all DQDB *subnetworks*, whether in an *open* or *looped Dual Bus* configuration, contain exactly one *node* with an active Default Slot Generator function at network initialization. This node is nominated prior to network initialization. In an open Dual Bus subnetwork, the Default Slot Generator function may be activated at any single node having this capability.

1.4.25 Head of Bus function: The function that generates *empty Queued Arbitrated (QA) slots*, *Pre-Arbitrated (PA) slots*, and *management information octets* at the point on each bus where data flow starts. The Head of Bus function also inserts the *virtual channel identifier* in the *PA segment header* of PA slots.

1.4.26 individual address: An *address* that identifies a single source or destination *service access point*.

1.4.27 Initial MAC Protocol Data Unit (IMPDU): A *protocol data unit (PDU)* formed in the *DQDB Layer* by the addition of *protocol control information* (including address information) to a *MAC Service Data Unit* received from the *Logical Link Control (LLC) Sublayer*. The IMPDU is segmented into 44-octet *segmentation units* for transfer in *Derived MAC Protocol Data Units (DMPDUs)*.

1.4.28 island: An operating part of a DQDB *subnetwork* that is isolated from the node containing the *default slot generator function*.

1.4.29 isochronous: The time characteristic of an event or signal recurring at known, periodic time intervals.

1.4.30 isochronous service octet: A single *octet* of data passed *isochronously* between the *DQDB Layer* and the *Isochronous Service User (ISU)*.

1.4.31 Isochronous Service User (ISU): The entity that uses the *isochronous* service provided by the *DQDB Layer* to transfer *isochronous service octets* over an established isochronous *connection*.

1.4.32 layer: A subdivision of the Open Systems Interconnection (OSI) architecture, constituted by *subsystems* of the same rank (ISO 7498).

1.4.33 layer management: Functions related to the administration of a given Open Systems Interconnection (OSI) *layer*. These functions are performed in the layer itself according to the protocol of the layer and partly performed as a subset of *network management* or *systems management*.

1.4.34 Layer Management Entity (LME): The entity in a *layer* that performs local management of a layer. The LME provides information about the layer, effects control over it, and indicates the occurrence of certain events within it.

1.4.35 Layer Management Interface (LMI): The service interface provided by the *Layer Management Entity (LME)* to the *Network Management Process (NMP)*.

1.4.36 local area network (LAN): A non-public data network in which serial transmission is used without store and forward techniques for direct data communication among data stations located on the user's premises.

1.4.37 Logical Link Control (LLC) procedure: In a *local area network (LAN)* or a metropolitan area network (MAN), the part of the protocol that governs the assembling of *Data Link Layer* frames and their exchange between data stations independently of how the *transmission medium* is shared.

1.4.38 Logical Link Control (LLC) Sublayer: In a *local area network (LAN)* or metropolitan area network (MAN), that part of the *Data Link Layer* that supports medium-independent data link functions, and uses the *Medium Access Control (MAC) Sublayer* service to provide services to the *Network Layer*.

1.4.39 looped Dual Bus: A DQDB *subnetwork* with the *Head of Bus functions* for both Bus A and Bus B collocated.

1.4.40 MAC address: An address that identifies a particular *Medium Access Control (MAC) Sublayer service access point (SAP)*.

1.4.41 MAC Service Data Unit (MSDU): The user data unit received in an MA-UNITDATA request for transfer by the *Medium Access Control (MAC) Sublayer.*

1.4.42 management information octets: *DQDB Layer Protocol Data Units (PDUs)* used to carry DQDB Layer Management Protocol information between peer *DQDB Layer Management Entities (LMEs).*

1.4.43 Medium Access Control (MAC) procedure: In a *local area network (LAN)* or metropolitan area network (MAN), the part of the protocol that governs access to the *transmission medium* independently of the physical characteristics of the medium, but taking into account the topological aspects of the *subnetwork*, in order to enable the exchange of data between *nodes*.

The MAC procedures include framing, error protection, and acquiring the right to use the underlying transmission medium.

1.4.44 Medium Access Control (MAC) Sublayer: In a *local area network (LAN)*, the part of the *Data Link Layer* that supports topology-dependent functions and uses the services of the *Physical Layer* to provide service to the *Logical Link Control (LLC) Sublayer* defined in ISO/IEC 10039.

In this part of ISO/IEC 8802, the combined set of functions in the *DQDB Layer* that support the MAC Sublayer service to the *Logical Link Control (LLC) Sublayer.*

1.4.45 message identifier: An identifier used to identify *Derived MAC Protocol Data Units (DMPDUs)* derived from the same *Initial MAC Protocol Data Unit (IMPDU).*

1.4.46 metropolitan area network (MAN): A network for connecting a group of individual stations and networks [for example, *local area networks (LANS)*] located in the same urban area.

NOTE—A MAN generally operates at a higher speed than the networks interconnected, crosses network administrative boundaries, may be subject to some form of regulation, and supports several access methods.

1.4.47 MID (Message Identifier) page: A set of one *message identifier* value.

1.4.48 multicast address: See *group address.*

1.4.49 multiport bridge: A *bridge* that interconnects two or more DQDB *subnetworks.*

1.4.50 Network Layer: In Open Systems Interconnection (OSI) architecture, the *layer* that provides services to establish a path between open systems with a predictable quality of service.

1.4.51 network management: Within this series of standards, the functions related to the management of *Data Link Layer* and *Physical Layer* resources and their status across an IEEE 802 *local area network (LAN)* or metropolitan area network (MAN).

1.4.52 Network Management Process (NMP): The entity that provides access to *network management* functions on behalf of the user of the network management services. In order to perform this function, NMPs may intercommunicate in a peer-to-peer manner and may use the services of NMPs in other *nodes* via a network management protocol. The NMP at a node is the user of the service provided at the *Layer Management Interface (LMI).*

1.4.53 node: A device that consists of an *access unit (AU)* and a single point of attachment of the access unit to each *bus* of a DQDB *subnetwork* for the purpose of transmitting and receiving data on that subnetwork. Adjacent nodes are connected by a *transmission link.*

1.4.54 octet: A group of eight adjacent bits.

1.4.55 offset: The *octet* position relative to the start of a *Pre-Arbitrated (PA) segment* used to carry an *isochronous service octet* for a particular *Isochronous Service User (ISU)*.

1.4.56 open Dual Bus: A DQDB *subnetwork* with the *Head of Bus function* for Bus A and the Head of Bus function for Bus B at different *node*s.

1.4.57 Physical Layer: In this part of ISO/IEC 8802, the subdivision that provides the protocol to allow transfer of *slot octet*s, *management information octets*, and *DQDB Layer* timing information over the *transmission link* between *DQDB Layer subsystem*s at adjacent *nodes*. The Physical Layer provides the service to the *DQDB Layer*.

1.4.58 Physical Layer Convergence Procedure (PLCP): In this part of ISO/IEC 8802, the part of the *Physical Layer* that supports the transfer of *slot octet*s, *management information octets*, and *DQDB Layer* timing information in a manner that adapts the capabilities of the *transmission system* to the service expected by the DQDB Layer.

1.4.59 pipelining: The function of forwarding in sequence some or all of the Beginning of Message (BOM) and Continuation of Message (COM) *Derived MAC Protocol Data Unit*s (*DMPDU*s) before receipt of the End of Message (EOM) DMPDU.

1.4.60 Pre-Arbitrated (PA) Access function: The *access control function* in this part of ISO/IEC 8802 that uses assigned *offset*s in *Pre-Arbitrated (PA) slot*s for the transfer of *isochronous service octet*s.

1.4.61 Pre-Arbitrated (PA) segment: A multiuser *segment* transferred using *Pre-Arbitrated Access (PA) functions*. The payload of the *PA segment* contains *isochronous service octets* from zero or more *Isochronous Service User*s *(ISU*s*)*.

1.4.62 Pre-Arbitrated (PA) slot: A *slot* that is dedicated by the *Head of Bus function* for transfer of *isochronous service octet*s in the payload of a *PA segment*.

1.4.63 protocol control information: Information exchanged between entities to coordinate their joint operation.

1.4.64 Protocol Data Unit (PDU): Information that is delivered as a unit between peer entities of a *local area network (LAN)* or a *metropolitan area network (MAN)* and that contains control information, address information, and may contain user data.

1.4.65 Queued Arbitrated (QA) Access function: The *access control function* in this part of ISO/IEC 8802 that uses the *Distributed Queue* to access *empty Queued Arbitrated (QA) slot*s for the transfer of *QA segment*s.

1.4.66 Queued Arbitrated (QA) segment: A *segment* transferred using *Queued Arbitrated (QA) Access functions*.

1.4.67 Queued Arbitrated (QA) slot: A *slot* that is used for the transfer of a *QA segment*.

1.4.68 read: The process of an *access unit (AU)* copying bits of a data stream as they pass on the *bus*.

1.4.69 reassembly: The function in the *DQDB Layer* that provides for the reconstruction of an *Initial MAC Protocol Data Unit (IMPDU)*. Reassembly is performed by concatenating the *segmentation unit*s received in *Derived MAC Protocol Data Unit*s *(DMPDU*s*)*. This is the inverse process to segmentation.

1.4.70 reconfiguration: The process by which the *Configuration Control function* activates and deactivates resources of a DQDB *subnetwork* to take account of a change in the operational status of a cluster, *node,* or *transmission link* in the subnetwork.

1.4.71 router: A functional unit that interconnects two computer networks that use a single *Network Layer* procedure but may use different *Data Link Layer* and *Physical Layer* procedures.

1.4.72 segment: The *protocol data unit (PDU)* of 52 octets transferred between peer *DQDB Layer* entities as the information payload of a *slot.* It contains a *segment header* of 4 octets and a *segment payload* of 48 octets. There are two types of segments: *Pre-Arbitrated (PA) segments* and *Queued Arbitrated (QA) segments*.

1.4.73 segment header: The *protocol control information* in a *segment*.

1.4.74 segment payload: The unit of data carried by a *segment.*

1.4.75 segmentation: The function in the *DQDB Layer* that fragments a variable length *Initial MAC Protocol Data Unit (IMPDU)* into fixed-length *segmentation units* for transfer in *Derived MAC Protocol Data Units (DMPDUs)* (cf., *reassembly*).

1.4.76 segmentation unit: The fixed-length data units of 44 octets formed by the fragmentation of an *Initial MAC Protocol Data Unit (IMPDU).*

1.4.77 service access point (SAP): The point at which services are provided by one *layer* (or *sublayer*) to the layer (or sublayer) immediately above it (ISO 7498).

1.4.78 service data unit (SDU): Information that is delivered as a unit between peer *service access points (SAPs).*

1.4.79 service primitive; primitive: An abstract, implementation-independent interaction between a service user and the service provider.

1.4.80 slot: The *protocol data unit (PDU)* of 53 octets used to transfer *segments*. It contains a segment of 52 octets and a 1 octet *Access Control Field (ACF)*. There are two type of slots: *Pre-Arbitrated (PA) slots* and *Queued Arbitrated (QA) slots*.

1.4.81 sublayer: A subdivision of a *layer* in the Open System Interconnection (OSI) reference model.

1.4.82 subnetwork: In this part of ISO/IEC 8802, a functional unit comprised of a single *Dual Bus* pair and those *access units (AUs)* attached to it. Subnetworks are physically formed by connecting adjacent *nodes* with *transmission links*.

1.4.83 subnetwork configuration: The topological arrangement of *nodes* to form a *subnetwork*. In normal operation a DQDB subnetwork can have one of two configurations, *open Dual Bus* or *looped Dual Bus*.

1.4.84 subsystem: An element in a hierarchical division of an open system that interacts directly only with elements in the next higher division or the next lower division of that open system (ISO 7498).

1.4.85 systems management: Functions in the application *layer* related to the management of various Open Systems Interconnection (OSI) resources and their status across all layers of the OSI architecture.

1.4.86 transmission link: The physical unit of a DQDB *subnetwork* that provides the transmission connection between adjacent *nodes*. Each transmission link accommodates both *buses* of the *Dual Bus* pair between the adjacent nodes.

1.4.87 transmission medium: The material on which information signals may be carried; e.g., optical fiber, coaxial cable, and twisted-wire pairs.

1.4.88 transmission system: The interface and *transmission medium* through which peer *Physical Layer* entities transfer bits.

1.4.89 upstream: The direction along a *bus* that is towards the *Head of Bus function*. This is opposite to the direction of data flow along a bus.

1.4.90 Virtual Channel Identifier (VCI): A label that is used to distinguish between the different virtual channels. A virtual channel is a logical association between entities that enables unidirectional transfer of *segments* between the entities. In the context of this part of ISO/IEC 8802, the VCI label can be used to allow a transmitter to distinguish between different outgoing *protocol data units (PDUs)*, and is used to allow a receiver to determine whether to receive an incoming segment as well as to distinguish between incoming PDUs.

1.4.91 write: The process of an *access unit (AU)* sending data downstream on a bus by logically ORing its outgoing data with the data pattern (normally all zeros) arriving from *upstream* on that bus.

1.5 Abbreviations and acronyms

ACF	Access Control Field
ANSI	American National Standards Institute
AU	Access Unit
BAsize	Buffer Allocation size
BCD	Binary Coded Decimal
BEtag	Beginning-End tag
BIF	Bus Identification Field
BOM	Beginning Of Message
BWBM	BandWidth Balancing Machine
BWB_CNTR	BandWidth Balancing CouNTeR
BWB_MOD	BandWidth Balancing MODulus
CC	Configuration Control (function)
CC_1	Configuration Control type 1 function
CC_2	Configuration Control type 2 function
CC_D	Default Configuration Control function
CCITT	the International Telegraph and Telephone Consultative Committee
CD	CountDown (counter)
CE	Connection Endpoint
CIB	CRC32 Indicator Bit
COCF	Connection-Oriented Convergence Function
COM	Continuation Of Message
CMIP	Common Management Information Protocol
CMIS	Common Management Information Service
CRC	Cyclic Redundancy Check
CRC32	32-bit Cyclic Redundancy Check
DA	Destination Address
DIS	Draft International Standard
DMPDU	Derived MAC Protocol Data Unit
DQDB	Distributed Queue Dual Bus
DQSM	Distributed Queue State Machine
DSG	Default Slot Generator
DSGS	Default Slot Generator Subfield

EOM	End Of Message
ETS	External Timing Source
ETSS	External Timing Source Subfield
FEBE	Far End Block Error
FERF	Far End Receive Failure
GPSM	Get Page State Machine
HCS	(segment) Header Check Sequence
HEL	Header Extension Length
HOB	Head of Bus
HOBS	Head of Bus Subfield
HOB_A	Head of Bus A
HOB_B	Head of Bus B
ICF	Isochronous Convergence Function
IEC	International Electrotechnical Commission
IMPDU	Initial MAC Protocol Data Unit
ISDN	Integrated Services Digital Network
ISDU	Isochronous Service Data Unit
ISO	International Organization for Standardization
ISU	Isochronous Service User
kbit/s	Kibobits per second
KPSM	Keep Page State Machine
LAN	Local Area Network
LLC	Logical Link Control
LME	Layer Management Entity
LMI	Layer Management Interface
LSS	Link Status Signal
MAC	Medium Access Control
MAN	Metropolitan Area Network
Mbit/s	Megabits per second
MCF	MAC Convergence Function
MCP	MAC Convergence Protocol
MID	Message IDentifier
MPA	MID Page Allocation
MPAF	MID Page Allocation Field
MSAP	MAC Service Access Point
MSDU	MAC Service Data Unit
NMP	Network Management Process
OSI	Open Systems Interconnection
PA	Pre-Arbitrated
PCC	Page Counter Control (subfield)
PCI	Protocol Control Information
PCM	Page Counter Modulus (subfield)
PCSM	Page Counter State Machine
PDU	Protocol Data Unit
Ph-SAP	Physical layer Service Access Point
PI	Protocol Identification
PL	PAD Length
PLCP	Physical Layer Convergence Procedure
PLCSM	Physical Layer Connection State Machine
POH (SDH)	Path Overhead
POI	Path Overhead Identifier (octets)
PR	Page Reservation (subfield)
PSR	Previous Segment Received
QA	Queued Arbitrated

QOS	Quality Of Service
REQ	REQuest
RIT	Reassembly IMPDU Timer
RQ	ReQuest (counter)
RQM	REQ Queue Machine
RSM	Reassembly State Machine
SA	Source Address
SAP	Service Access Point
SDH	Synchronous Digital Hierarchy
SDU	Service Data Unit
SG_1	Slot Generator type 1 function
SG_2	Slot Generator type 2 function
SG_D	Default Slot Generator function
SNCF	SubNetwork Configuration Field
SSM	Single Segment Message
VCI	Virtual Channel Identifier

1.6 Conformance

An implementation may be called conformant with this part of ISO/IEC 8802 if all mandatory functions in this part of ISO/IEC 8802 are implemented according to the respective mandatory specification.

Additional optional features need not be implemented. However, if they are implemented, they should conform with the specification in this part of ISO/IEC 8802, if applicable.

By way of example, support for 48-bit universally administered MAC Service Access Point (MSAP) addresses is mandatory. However, if 60-bit MSAP addresses are implemented to carry numbers allocated according to CCITT Recommendation E.164, the coding shall be as specified in this part of ISO/IEC 8802.

1.7 Notation

1.7.1 Service specification method and notation

This clause covers the method of specification for the services required of the DQDB Layer by the LLC Sublayer, by the ISUs, and by the Connection-Oriented Data Service Users. It also covers the method of specification for the services required of the Physical Layer by the DQDB Layer.

In general, the services of an (N)-layer (or sublayer) are the capabilities that it offers to an (N)-user in the next higher layer (or sublayer). In order to provide its service, a layer (or sublayer) builds its functions on the services it requires from the next lower layer (or sublayer). Figure 1-2 illustrates the layer service model.

Services are specified by describing the information flow between the (N)-user and the (N)-layer (or Sublayer). This information flow is modeled by discrete, instantaneous events, which characterize the provision of a service. Each event consists of passing a service primitive from one layer (or Sublayer) to the other through the (N)-layer (or sublayer) service access point (SAP) associated with an (N)-user. Service primitives convey the information required in providing a particular service. These service primitives are an abstraction in that they specify only the service provided rather than the means by which the service is provided. This definition of service is independent of any particular interface implementation.

Services are specified by describing the service primitives and parameters that characterize each service. A service may have one or more related primitives that constitute the activity that is related to the particular service. Each service primitive may have zero or more parameters that convey the information required to provide the service.

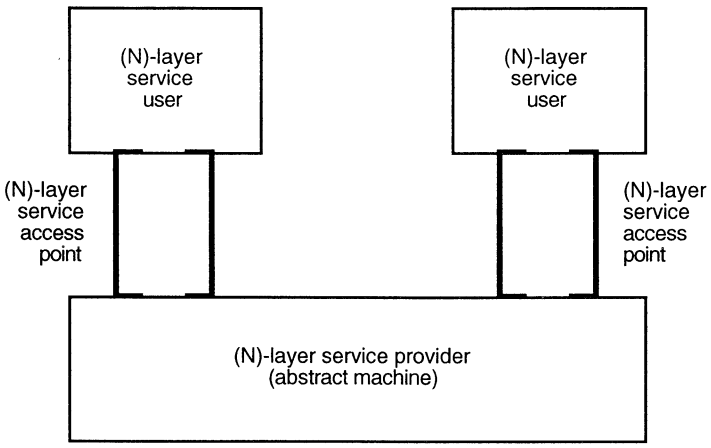


Figure 1-2—Layer service model

Primitives are of four generic types:

a) REQUEST. The request primitive is passed from the (N)-user to the (N)-layer (or sublayer) to request that a service be initiated.

b) INDICATION. The indication primitive is passed from the (N)-layer (or sublayer) to the (N)-user to indicate an internal (N)-layer (or sublayer) event that is significant to the (N)-user. This event may be logically related to a remote service request, or may be caused by an event internal to the (N)-layer (or sublayer).

c) RESPONSE. The response primitive is passed from the (N)-user to the (N)-layer (or sublayer) to complete a procedure previously invoked by an indication primitive.

d) CONFIRM. The confirm primitive is passed from the (N)-layer (or sublayer) to the (N)-user to convey the results of one or more associated previous service request(s).

This part of ISO/IEC 8802 only requires the use of request and indication primitives. Possible relationships among primitive types are illustrated by the time sequence diagrams shown in figure 1-3. The figure also indicates the time relationship of the primitive types. The key of figure 1-3 depicts the time sequence framework used to illustrate the interactions between the two correspondent service users and their associated service provider. Primitive types occurring earlier in time and connected by dotted lines are the logical antecedents of subsequent primitive types. If there is no specific relationship between events, in that it is impossible to predict which will occur first, but both must occur within a finite period of time, then a tilde (~) is used.

This part of ISO/IEC 8802 uses the conventions for naming service primitives contained in annex A of ISO/TR 8509.

1.7.2 State machine notation

The operation of a protocol can be described by subdividing the protocol into a number of interrelated functions. The operation of a function can be described by state machines. Each machine represents the domain of a function and consists of a group of connected, mutually exclusive states. Only one state of a function is active at any given time.

Each state that the function can assume is represented by a vertical line. All permissible transitions between states of a function are represented by arrows from one state to the next state. The condition that causes a change from one state to another is shown above the transition arrow. The affected signals and flags are shown below the transition arrow.

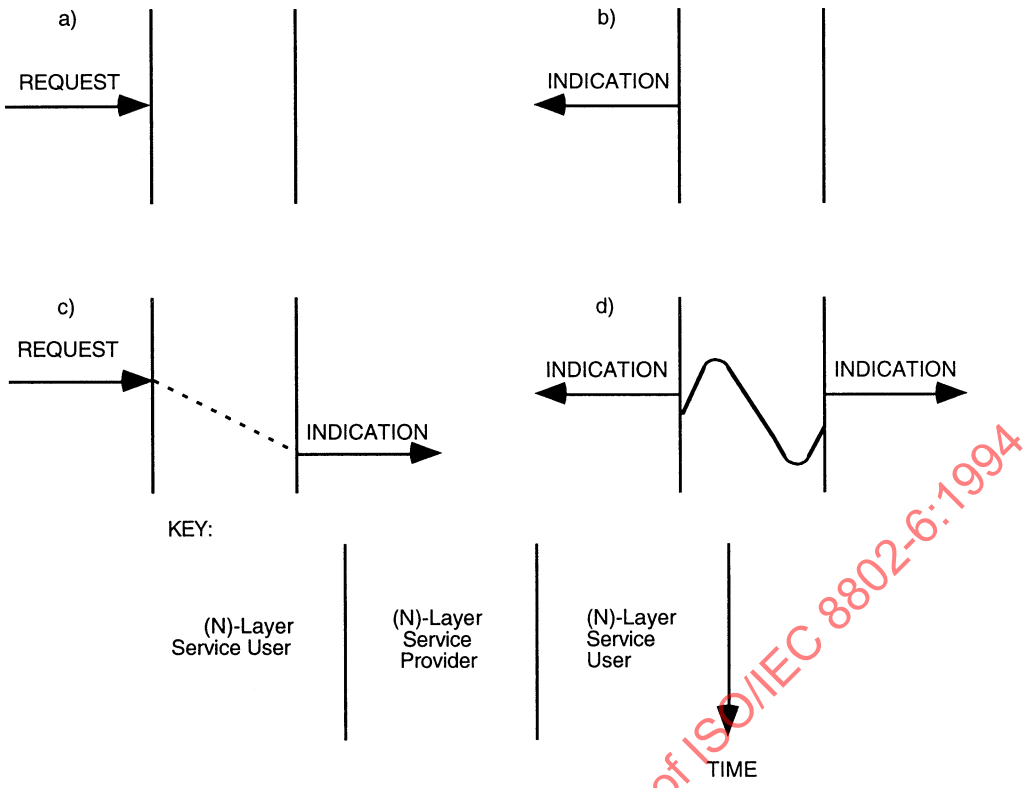


Figure 1-4—Time sequence diagram

Signals and flags refer to the form by which one functional block communicates with another. Both terms arise from the use of the state machines as a descriptive method in this part of ISO/IEC 8802. Signals are transient communications (in theory, they are of infinitesimal time width) that occur during a change of state. Flags are steady, nontransient indications that remain constant until explicitly changed.

1.8 Organization of this part of ISO/IEC 8802

Clause 2 contains an overview of the operation of a DQDB subnetwork and the communication services supported. It also provides an introduction to the functional architecture of a DQDB node.

Clause 3 contains the service definition for the DQDB Layer, which gives a formal definition for the MAC Sublayer service offered to LLC and for the isochronous service, and a statement on the requirements for the connection-oriented data service.

Clause 4 contains the definition of the Physical Layer service to the DQDB Layer.

Clause 5 gives a rigorous description of the internal operation of a DQDB node, with forward references to clauses 6, 7, 8, 9, and 10. Clause 6 describes the DQDB Layer PDU formats (with the exception of the management information octets, which are described in clause 10). Clause 7 describes the facilities required for operation of the DQDB Layer, such as timers, counters, parameters, and status indicators. Clause 8 specifies the state machines that describe peer-to-peer communication by DQDB Layer entities.

DQDB Layer Management is described in clauses 9 and 10. Clause 9 describes the services offered at the DQDB LMI to the NMP. This interface can support remote management of a DQDB node using management protocols such as those specified in ISO 9595 (CMIS) and ISO 9596 (CMIP). Clause 10 describes the

operation of the DQDB Layer Management Protocol, which is the mechanism by which DQDB Layer Management Entities communicate to perform layer-wide management of DQDB Layer resources.

Clause 11 describes the principles of operation of the Physical Layer. Clause 13 describes the operation of the Physical Layer Convergence Procedure (PLCP) required to support a DS3 rate (as specified in ANSI T1.102 and T1.107). Subsequent clauses describe the operation of the PLCPs required to support transmission systems based on CCITT Recommendation G.703 and operating at 2.048 Mbit/s (clause 14), 34.368 Mbit/s (clause 15), and 139.264 Mbit/s (clause 16). Clause 17 describes the operation of the PLCP required to support SDH transmission systems operating at 155.520 Mbit/s, as specified in CCITT Recommendations G.707, G.708, and G.709.

2. Overview

This clause is intended as an introduction to the DQDB access mechanism and is not intended as a rigorous definition of the protocol. The first part of this clause, 2.1, provides an overview of the operation of a DQDB subnetwork. The second part, 2.2, provides a focused view describing the functional architecture of a DQDB node. Rigorous descriptions of the node functional operation, and of the DQDB Layer protocols, are given in clause 5, and clauses 8 and 10, respectively.

2.1 DQDB subnetwork

The purpose of a MAN is to provide integrated services, such as data, voice, and video, over a large geographical area. This part of ISO/IEC 8802 describes the DQDB subnetwork that can be used as a component part of a MAN. Typically, a MAN would consist of interconnected DQDB subnetworks. The interconnection of subnetworks could be via bridges, routers, or gateways, but this is not covered in this International Standard. MANs can be interconnected by suitable means to form a wide area network.

The DQDB subnetwork is a distributed multiaccess network that supports integrated communications. In particular, the DQDB subnetwork supports connectionless data transfer, connection-oriented data transfer, and isochronous communications (e.g., voice). The various communication transfer functions share flexibly the total capacity of the subnetwork.

In the first part of this clause, 2.1.1, there is a description of the Dual Bus architecture and the format by which nodes communicate. The second part of the clause, 2.1.2, describes the control of access to the Dual Bus. This includes Distributed Queue access control for non-isochronous communications and Pre-Arbitrated access control for isochronous communications.

The third part of this clause, 2.1.3, describes the functions specified in addition to the access control mechanism to support the DQDB Layer services. For example, it describes how LLC is supported by the DQDB subnetwork.

The fourth part of this clause, 2.1.4, provides guidance to network administrators in the application of the standard.

The final part of this clause, 2.1.5, describes the Configuration Control function, which is important in subnetwork start-up and in subnetwork reconfiguration in the case of bus faults.

2.1.1 Dual Bus architecture

The Dual Bus architecture of a DQDB subnetwork consists of two unidirectional buses and a multiplicity of nodes along the buses, as shown in figure 2-1. The buses, denoted Bus A and Bus B, support communications in opposite directions allowing full duplex communications between any pair of nodes on the subnetwork. Therefore, it is desirable for a node to know which bus to use to communicate to another node. Annex C provides information as to how this can be done for connectionless services.

As both buses are operational at all times, the capacity of the subnetwork is up to twice the capacity of a single bus. The operation of the two buses in the transfer of data is independent.

The data on each bus are formatted into either DQDB Layer Management information octets or fixed-length slots, this format being generated by the node at the head of each bus. Hence, there must be one node at the Head of Bus A and one node at the Head of Bus B. The management information octets are used to maintain the operational integrity of the subnetwork. The slots are used to carry data between the nodes. That is, nodes may write into slots under the control of the access protocol. All data flow terminates at the end of the bus.

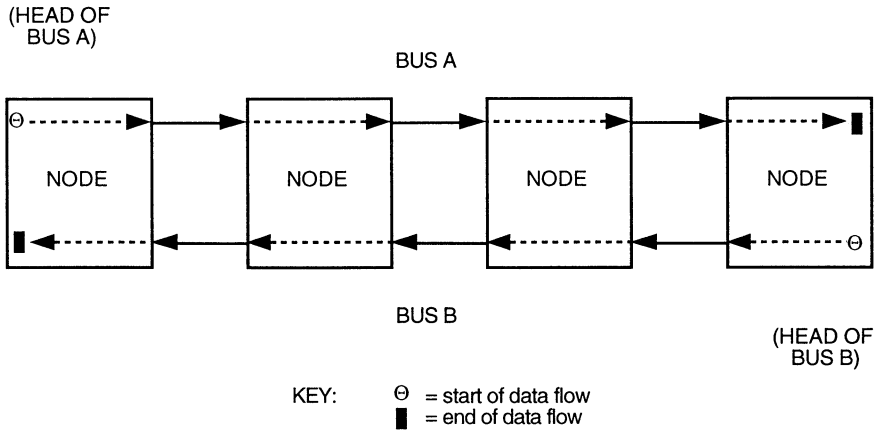


Figure 2-1—Dual Bus architecture

The nodes on the subnetwork consist of an access unit and the attachment of the access unit to the two buses. The access unit performs the node's DQDB Layer functions and attaches to a bus via one read and one write connection. The writing of data to the bus is a logical OR of the data from upstream with the data from the access unit. The read connection is placed logically ahead of the write connection and allows all data to be copied from the bus unaffected by the unit's own writing.

The read and write functions can be implemented using either passive or active techniques.[8] An example implementation of the AU connection is shown in figure 2-2. With this connection arrangement, an AU can copy data that passes on the bus, but can never remove it, and only alter data when it is permitted by the access protocol.

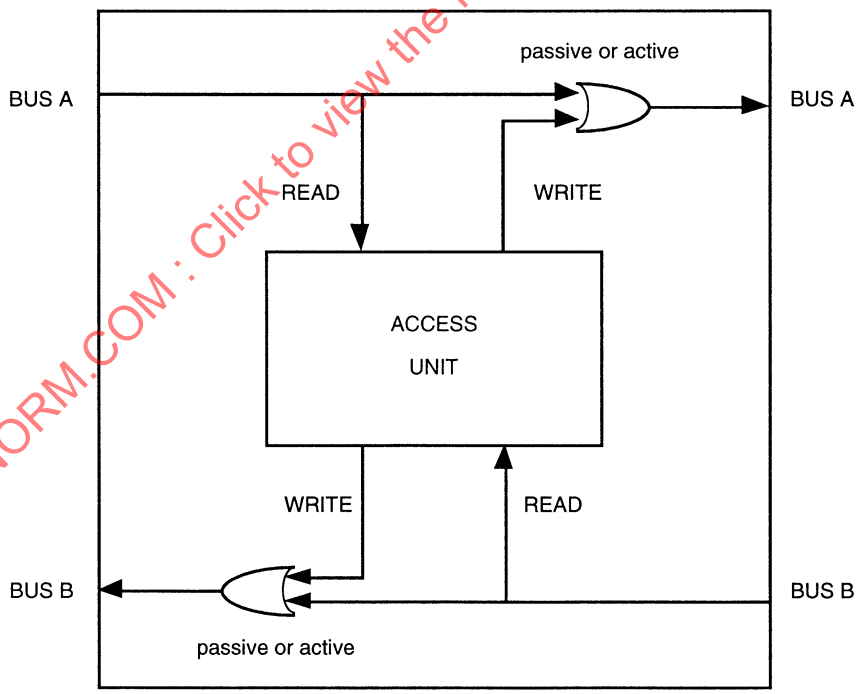


Figure 2-2—Example access unit attachment

[8] A passive optical implementation would limit the number of nodes on the bus according to the optical power budget.

With the exception of AUs performing the Head of Bus functions, the operation of the bus is only loosely coupled to the operation of each AU; an AU can fail (provided that it does not destructively write to the bus), or even be removed from the subnetwork, with no consequence on the operation of the rest of the subnetwork.

Under normal conditions there is a single source of slot timing for the subnetwork, which is used to ensure that all nodes in the subnetwork transfer data (i.e., slots and management information octets) at the same average rate. This is necessary to ensure stable operation of the Distributed Queue access mechanism (see annex D) and to ensure that isochronous communications can be supported without slips being introduced by different timing sources at the two communicating entities.

The timing source can be traceable to an external timing source, such as that provided by a public telecommunications network. Any node or nodes can be designated by network operations procedures to provide the External Timing Source function. If the subnetwork is supporting certain isochronous services and is connected to a public network, then the external timing source may be required. However, in the absence of an external timing source, one node with the capability specified below will be selected by subnetwork configuration control procedures to be the source of subnetwork timing. In this case, the selected node must have available and operate a local clock that has a nominal period of 125 µs.

2.1.2 Access control to the Dual Bus subnetwork

The DQDB Layer protocols and protocol data units allow a DQDB subnetwork to be operated at a variety of speed and distance combinations. This permits a wide range of transmission systems to be used in the underlying Physical Layer.

The DQDB Layer provides two modes of access control to the Dual Bus. These are Queued Arbitrated (QA) and Pre-Arbitrated (PA), which use QA and PA slots for access respectively. Each slot contains an ACF and a segment, which forms the payload of the slot.

Queued Arbitrated access is controlled by the Distributed Queueing protocol and would be used typically to provide non-isochronous services. Pre-Arbitrated access would be used typically to provide isochronous services. The Distributed Queue and Pre-Arbitrated access protocols are described in the following subclauses.

2.1.2.1 The Distributed Queue access protocol

Distributed Queueing is a media access protocol that controls the access to the payload of QA slots on the DQDB bus. The fixed-length payload of a QA slot is called a QA segment.

The protocol provides deterministic access control for services that are typically bursty in nature. In particular, the protocol can enable all of the payload bandwidth to be used and the average slot access delay approximates that of a perfect scheduler up to high levels of subnetwork loading.

The operation of the protocol is based on two control fields: the BUSY bit, which indicates whether or not a slot is used, and the REQUEST field, which is used to indicate when a segment has been queued for access. Each node, by counting the number of requests it receives and unused slots that pass, can determine the number of segments queued (i.e., in line) ahead of it. This counting operation establishes a single queue across the subnetwork of segments queued for access to each bus.

With such queued access, levels of priority can be established by operating a number of queues, one for each level. Within each level the performance characteristics described above are maintained. Segments will gain access as soon as capacity becomes available, but priority is always given to segments in higher level queues. This part of ISO/IEC 8802 specifies three levels of priority.

The operation of Distributed Queueing is fundamentally different from other MAC protocols. In Distributed Queueing, information that explicitly indicates the queueing state of the subnetwork is kept in the nodes. Hence, when a node has data to transmit, the node need not first derive information from the subnetwork to tell it when it can gain access.

With Distributed Queueing, a current state record that holds the number of segments awaiting access to the bus is kept in every node. When a node has a segment for transmission, it uses this count to determine its position in the distributed queue. If no segments are waiting, permission for access is immediate, otherwise deference is given only to those segments already queued.

To facilitate effective sharing of the bandwidth, the Distributed Queue protocol includes a bandwidth balancing mechanism to occasionally skip the use of empty QA slots.

2.1.2.1.1 The basic Distributed Queueing algorithm

The operation of the basic Distributed Queueing algorithm for access to Bus A is illustrated in figure 2-3. In this case, Bus A is the forward bus and Bus B is the reverse bus. An identical, but independent, arrangement applies for access to the opposite bus, Bus B.

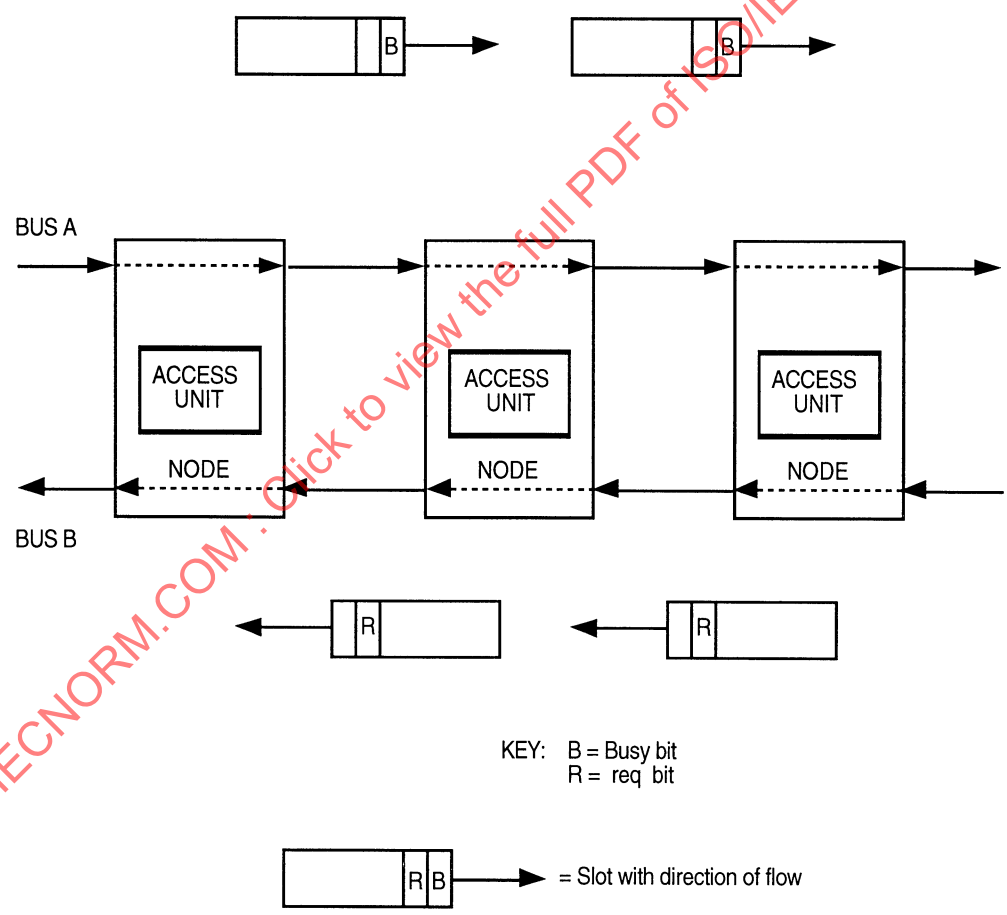


Figure 2-3—Queue formation on Bus A

Each slot on the buses (whether a QA or PA slot) contains an ACF that includes a BUSY bit and a REQUEST field of three Request bits (henceforth referred to as REQ bits), one for each priority level. The

BUSY bit indicates whether or not the slot is used. The REQ bits are to signal when a QA segment has been queued on the reverse bus.

The operation of the Distributed Queueing algorithm within a single priority level is described first. The description of the multiple priority scheme follows, as does a description of the bandwidth balancing mechanism.

When an AU has a QA segment for transmission on the forward bus it will cause a single REQ to be sent on the reverse bus. This REQ will eventually be written into the next free REQ bit of the required priority on the reverse bus. The bit once written will pass to all upstream AUs, where upstream is defined in relation to data flow on the forward bus, in this case Bus A. This REQ bit serves as an indicator to the upstream AUs that an additional QA segment is now queued for access. For each AU, the Distributed Queueing algorithm allows, at most, one QA segment per priority level, to be queued for access to each bus.

Each AU keeps track of the number of QA segments queued downstream from itself for access to the forward bus by counting the REQ bits as they pass on the reverse bus, as shown in figure 2-4a). The request (RQ) counter is incremented for each REQ passing on the reverse bus. For a node that is not queued to send, one REQ in the RQ counter is cancelled each time an empty slot passes on the forward bus. This is done since the empty slot that passes the AU will be used by one of the downstream queued QA segments. Hence, with these two actions, the RQ counter keeps a record of the number of segments queued downstream.

In addition to issuing the REQ for the reverse bus, an AU with a QA segment to send will transfer the current value of the RQ counter to another counter, the countdown (CD) counter, as shown in figure 2-4b), with the RQ counter then being reset to zero. This action loads the CD counter with the number of downstream segments queued ahead of it. This, along with the issuing of the REQ for the AU's segment, effectively places the QA segment in the distributed queue. The distributed queue at a given level of priority approximates a first-in-first-out (FIFO) queue of the QA segments at the heads of the local queues in each node.

To ensure that the segments registered in the CD counter gain access before the newly queued segment in the given AU, the CD counter is decremented for every empty slot that passes on the forward bus. This operation is shown in figure 2-4b). The given AU can transmit its QA segment in an empty slot provided the CD count is zero. For this single priority description, this is equivalent to claiming the first free slot after the CD count reaches zero, which ensures that no downstream segment that queued after the given segment can access out of order.

During the time that the AU is waiting for access for its segment, any new REQs received from the reverse bus are added to the RQ counter, as shown in figure 2-4b). Hence, the RQ counter still tracks the number of segments newly queued downstream and the count will be correct for the next QA segment access.

It should be noted that the control of the access of the QA segment to the forward bus is determined solely by the values of the counters. Access is not inhibited if the value of the CD counter is zero but the REQ bit associated with the QA segment has not yet been written to the reverse bus. That is, the operation of writing REQ bits and sending QA segments is independent.

Thus, with the use of two counters in each AU, one counting unsatisfied access requests and the other counting down before access, a FIFO queue is established for access to the forward bus. The queue formation is also such that, except for bandwidth balancing considerations (described later in this clause), a slot is never wasted on the subnetwork if there is a segment queued for it. This is guaranteed since the CD count in AUs with segments queued represents the number of downstream segments ahead in the queue. Since at any point in time[9] one segment must have queued first, then at least one AU is guaranteed to have a CD count of zero. It is the most upstream of these that will gain access.

[9] More correctly, at the same point in time normalized for propagation delay of the slot along the bus.

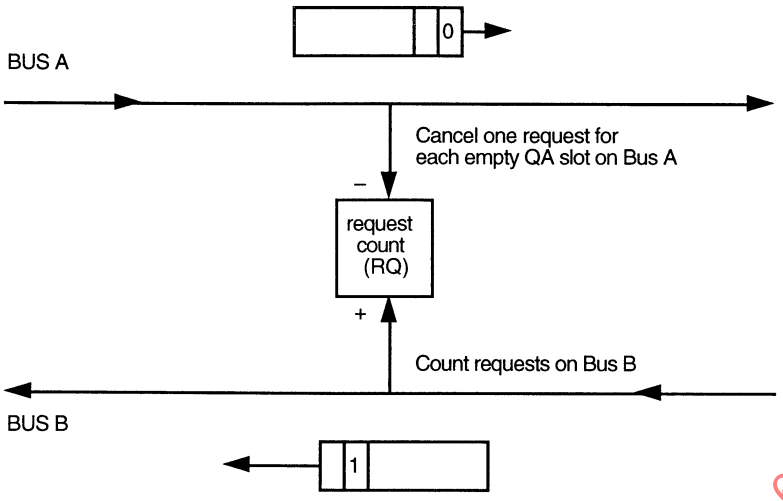


Figure 2-4a)—Node not queued to send (Bus A)

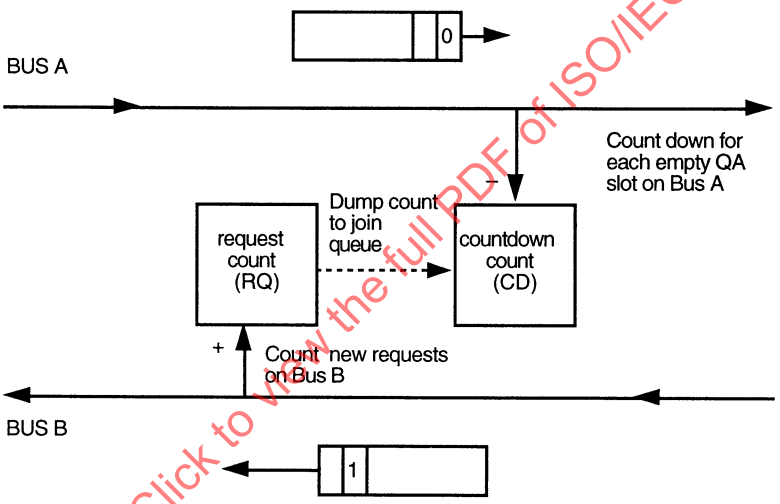


Figure 2-4b)—Node queued to send (Bus A)

As an example of the operation of Distributed Queueing, consider figure 2-5, which shows a subnetwork consisting of five AUs, three of which will queue for access. Assume that each request counter, the value of which is indicated by RQ at the AU, and each CD counter, the value of which is indicated by CD at the AU, are all initially zero and that all slots passing on Bus A are used. AUs 5, 2, and 3 queue in that order, using the operations described above. The counters in the final state can be interpreted as follows:

AU 1: There are three AUs downstream on Bus A that have QA segments queued for access.

AU 2: There is one downstream AU with a QA segment queued for access before AU 2, as represented by the one in the CD counter. There is also one downstream AU with a QA segment queued for access after AU 2, as represented by the one in the RQ counter.

AU 3: There is one downstream AU with a QA segment queued for access before AU 3, as represented by the one in the CD counter, and no downstream AU with a QA segment queued after AU 3, as represented by the zero in the RQ counter. (Note that the REQ from AU 2 is not registered at AU 3.)

AU 4: There is one downstream AU with a QA segment queued for access.

AU 5: There are no downstream AUs from AU 5; hence, both the RQ and CD counts are zero.

Now consider figure 2-6, which shows the AUs gaining access. Assume no further requests are received on the reverse bus and that empty QA slots pass on bus A. The AUs then gain access in the order 5, 2, then 3, which is the order of queueing. Note that after AU 5 gains access both AU 2 and AU 3 have a CD count of zero. AU 2 will gain access first as it is the most upstream.

2.1.2.1.2 Priority Distributed Queueing

The Distributed Queueing protocol supports the assignment of priority to QA segment access. However, all connectionless data segments must be sent at the lowest priority, priority level 0 (see 7.3.3). Future use of priorities for other services is under study. For upward compatibility, Distributed Queues for each level of priority are required in this part of ISO/IEC 8802.

The operation of separate distributed queues for each level of priority is achieved by using a separate REQ bit on the reverse bus for each level of priority and separate RQ and CD counters for each priority level. The counters operate similarly to the single priority case, except that account must be taken of REQs at higher levels.

That is, for an AU that does not have a QA segment queued at a particular priority level, the RQ counter operating at that level will count REQs at the same *and higher* priority levels. Thus, the RQ counter records all queued segments at equal and higher priorities.

If the AU does have a QA segment queued at a particular priority level, then the RQ counter operating at that level will only count REQs at the same priority level. However, the operation of the CD counter at that priority level is slightly altered. The CD counter will, in addition to counting down the received empty slots, increment for REQs received at higher priority levels. This allows the higher priority segments to claim access ahead of already queued segments.

It should be noted that decrementing of the RQ counters and CD counters at all priority levels occurs when an empty QA slot is received at the function within the node operating the Distributed Queue, not when it is sent by this function. This ensures that the correct counter values are maintained if the highest priority segment queued by the AU gains access to the empty slot and the AU marks the slot as busy.

The intent of the priority scheme is for the access performance of the highest priority traffic to be unaffected by lower priority traffic. Such a feature is very important in network signaling. However, it should be noted that priority should not be relied upon for all subnetwork distance, speed, and loading conditions. (See 2.1.4.)

This part of ISO/IEC 8802 specifies three levels of access priority. The definition of how these levels are used is a matter for the operator of the particular DQDB subnetwork, although it is likely that the highest level would be used for network management and signaling. The DQDB Layer assumes that the user of the QA access requests the appropriate priority level for a particular segment transfer.

2.1.2.1.3 Bandwidth balancing

To facilitate effective sharing of the bandwidth, the Distributed Queue access protocol includes a bandwidth balancing mechanism to occasionally skip the use of empty QA slots. To support this function, the Distributed Queueing function in the node operates a bandwidth balancing counter for each bus. There is also a system parameter called the Bandwidth Balancing Modulus, denoted BWB_MOD. If BWB_MOD is set to zero, the bandwidth balancing mechanism is disabled. If BWB_MOD is set to a value greater than zero, it indicates when the value of the bandwidth balancing counter should roll over to zero, as explained below.

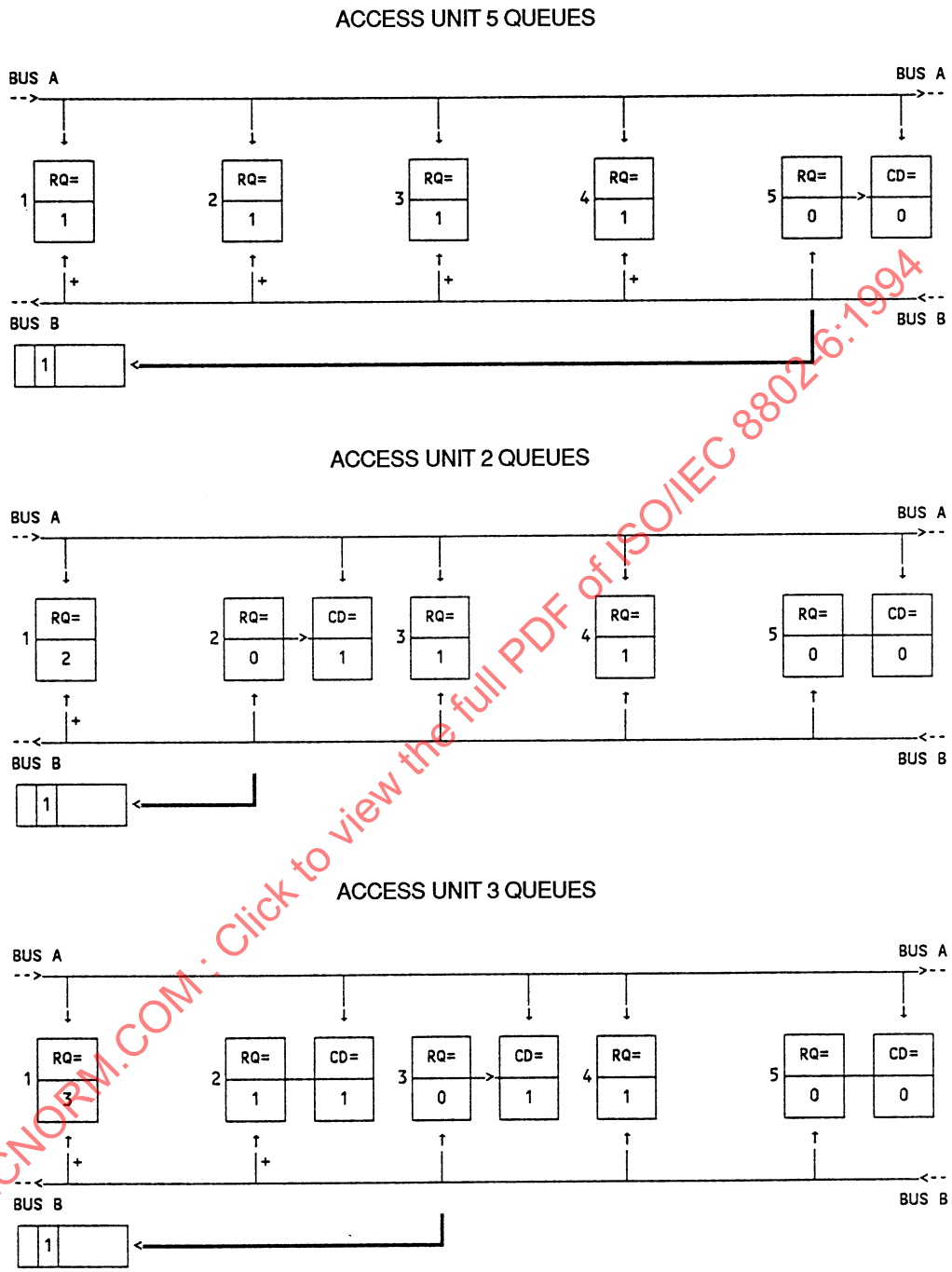


Figure 2-5—Queueing for access (Bus A)

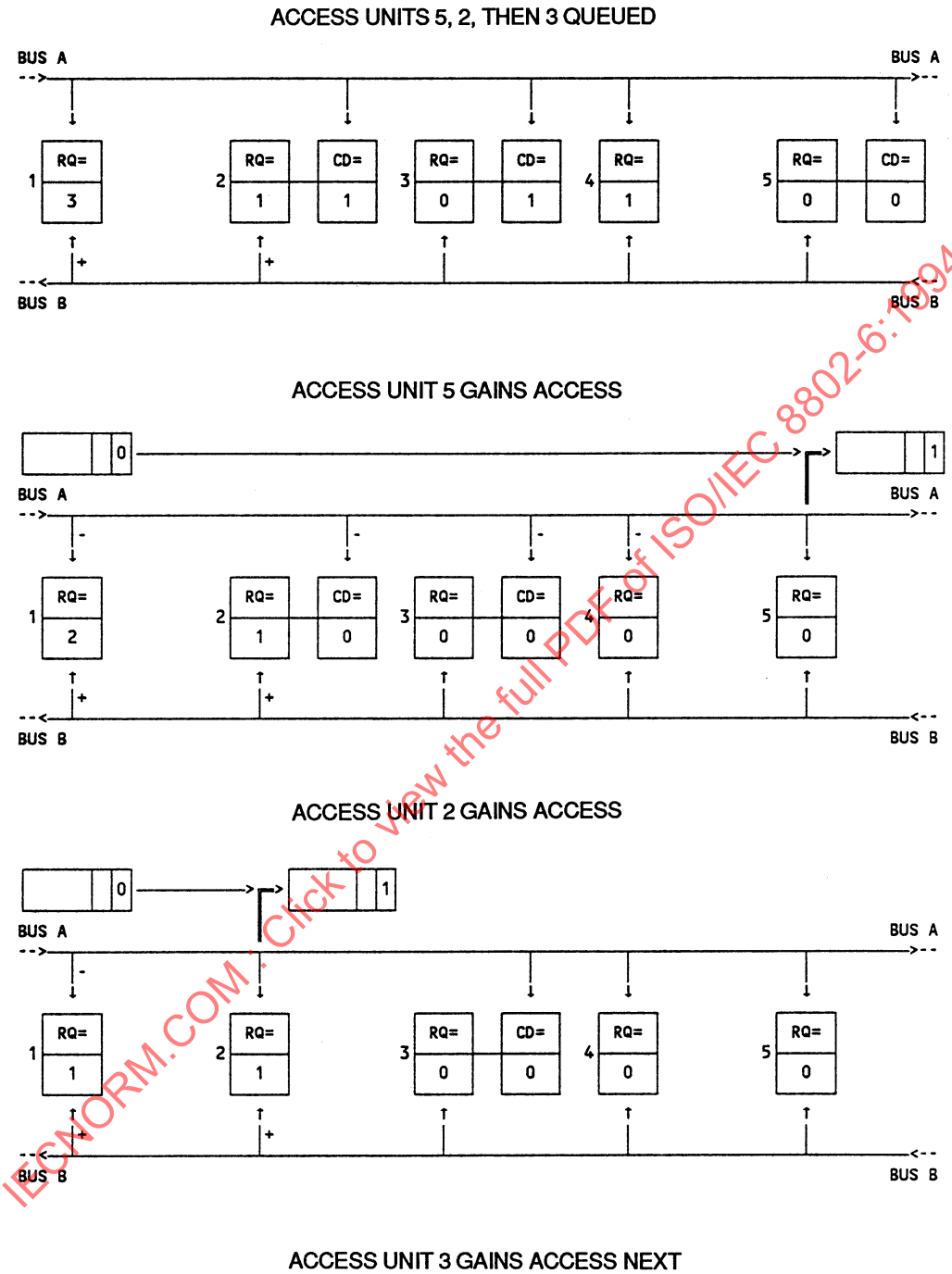


Figure 2-6—Gaining access (Bus A)

Whenever the node transmits a QA segment on one of the buses, the bandwidth balancing counter for that bus is incremented by one, provided that the counter does not have a value of (BWB_MOD – 1). If the counter does have a value of (BWB_MOD – 1) when the QA segment is transmitted, the bandwidth balancing counter is reset to zero instead.

When the bandwidth balancing counter for a bus is reset to zero, this is an indication that the node should skip the use of one empty QA slot on that bus. To make that happen, the Distributed Queueing function increments the request counters for that bus for all priority levels for which no QA segment is queued, and increments the countdown counters for that bus for all priority levels for which a QA segment is queued.

NOTES

1—Resetting the bandwidth balancing counter for a bus has no effect on the queue of requests waiting to be sent on the opposite bus.

2—If the value of BWB_MOD is one, the bandwidth balancing counter always has the value zero. This means that every time the node transmits a QA segment on the bus, the request or countdown counter corresponding to each priority level for that bus is incremented, as described above.

2.1.2.2 Pre-Arbitrated access control

Pre-Arbitrated (PA) slot access will be used typically to provide for transfer of isochronous service octets. The access to PA slots and the use of PA segment payloads differ from that for QA access. The access differs in that PA slots are designated by the node at the head of bus and that more than one AU may share access to the slot. A PA segment payload consists of a number of octets, each of which can be used by a different AU. Therefore, an AU may write zero, one or more isochronous service octets into designated positions of a PA segment payload. The AU is notified of the offsets of these octet positions relative to the start of the PA segment payload via the DQDB Layer Management procedures.

The designation of slots to PA access and the marking of the PA slot and PA segment header is controlled by functions in the node at the head of the bus. In particular, the Head of Bus function must write the Virtual Channel Identifier (VCI) field into PA slots. The Head of Bus function must also ensure that the PA slots for each VCI value are provided in a periodic manner on the bus to guarantee that sufficient bandwidth is available for the Isochronous Service Users (ISUs).

The access to the PA slots by an AU commences by examining the VCI. For each VCI value that the AU must access, the AU will have a table that indicates which octet offsets within the slot the AU should use for reading and writing. The AU will write isochronous service octets into those write positions and will read from positions the table has marked for reading. The PA slot is ignored if the VCI is not one in use by the AU.

2.1.3 Provision of DQDB Layer services

The Queued Arbitrated and Pre-Arbitrated functions of the DQDB Layer provide access control to the Dual Buses. These access control functions are used by a range of convergence functions to create the DQDB Layer services shown in figure 2-7 and defined formally in clause 3. This part of ISO/IEC 8802 specifies the convergence function for the provision of the ISO/IEC 10039 MAC Sublayer service to the LLC Sublayer. Also specified is a general isochronous service. Future services, such as a connection-oriented data service, are the subject of further study.

At no time do two convergence functions simultaneously use the same VCI at a given node. Hence, if a node is to receive a particular segment, the VCI can be used to direct the segment to the appropriate convergence function at the node. An overview of the convergence functions specified in this part of ISO/IEC 8802 is given in the following subclauses.

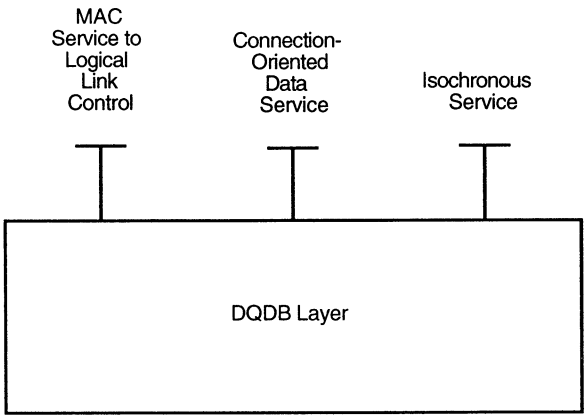


Figure 2-7—DQDB Layer services

2.1.3.1 Provision of MAC service to LLC

The provision of MAC service to LLC consists of the segmentation of the MAC Service Data Unit (MSDU) at the source into fixed-length units and the transfer of these fixed-length units to the destination, which reassembles them into the MSDU.

The segmentation process follows the formation of an Initial MAC Protocol Data Unit (IMPDU) by the addition of an IMPDU header,[10] an optional Header Extension, an optional 32-bit CRC, a Common PDU trailer,[11] and a variable-length PAD field to the MSDU. The PAD field ensures that all of the fields added to the MSDU are 32-bit aligned. The IMPDU is fragmented into fixed-length segmentation units, as shown in figure 2-8, for transfer in QA segment payloads. There may be padding of the IMPDU with trailing zero octets to ensure complete filling of the last segmentation unit.

All segment payloads that support the MAC service are called Derived MAC Protocol Data Units (DMPDUs) and consist of a header field and a trailer field[12] along with the segmentation unit, as shown in figure 2-9. The DMPDU header field consists of three subfields. The first is the Segment Type subfield. The second is the Sequence Number subfield. The third subfield is the Message Identifier (MID). The DMPDU trailer field consists of two subfields.

The first is the Payload Length subfield. The second subfield is the Payload CRC.

The MID is used to provide the logical linking between segmentation units derived from the same IMPDU, and should be unique on a subnetwork while the IMPDU is being transferred. Each AU will have at least one unique MID. The allocation of the MID numbers is controlled by the MID page allocation scheme, which is a distributed method for claiming and keeping MID values that are unique across the whole subnetwork.[13]

[10] The IMPDU header is formed by the concatenation of two types of header information. The first four octets of the 24-octet IMPDU header are called the Common PDU header. The remaining 20 octets of the IMPDU header are called the MAC Convergence Protocol (MCP) header. The MCP header is specific to the transfer of a MSDU by the DQDB Layer, and is not used to support the connection-oriented data service.

[11] The Common PDU header and the Common PDU trailer are intended to support a common segmentation and reassembly procedure across all of the bursty data services that can be provided by the DQDB Layer, such as the MAC service and the connection-oriented data service. The format of the Common PDU header and the Common PDU trailer is the same for both of these services, although the exact use of the fields differs.

[12] As with the Common PDU header and Common PDU trailer for the IMPDU, the DMPDU header and DMPDU trailer are also intended to support the common segmentation and reassembly procedure across all of the bursty data services that can be provided by the DQDB Layer. Hence, the QA segment payloads used to support all of these services will contain a header and trailer with identical format to those carried in the DMPDU.

[13] The MID page allocation scheme is part of the DQDB Layer Management protocol.

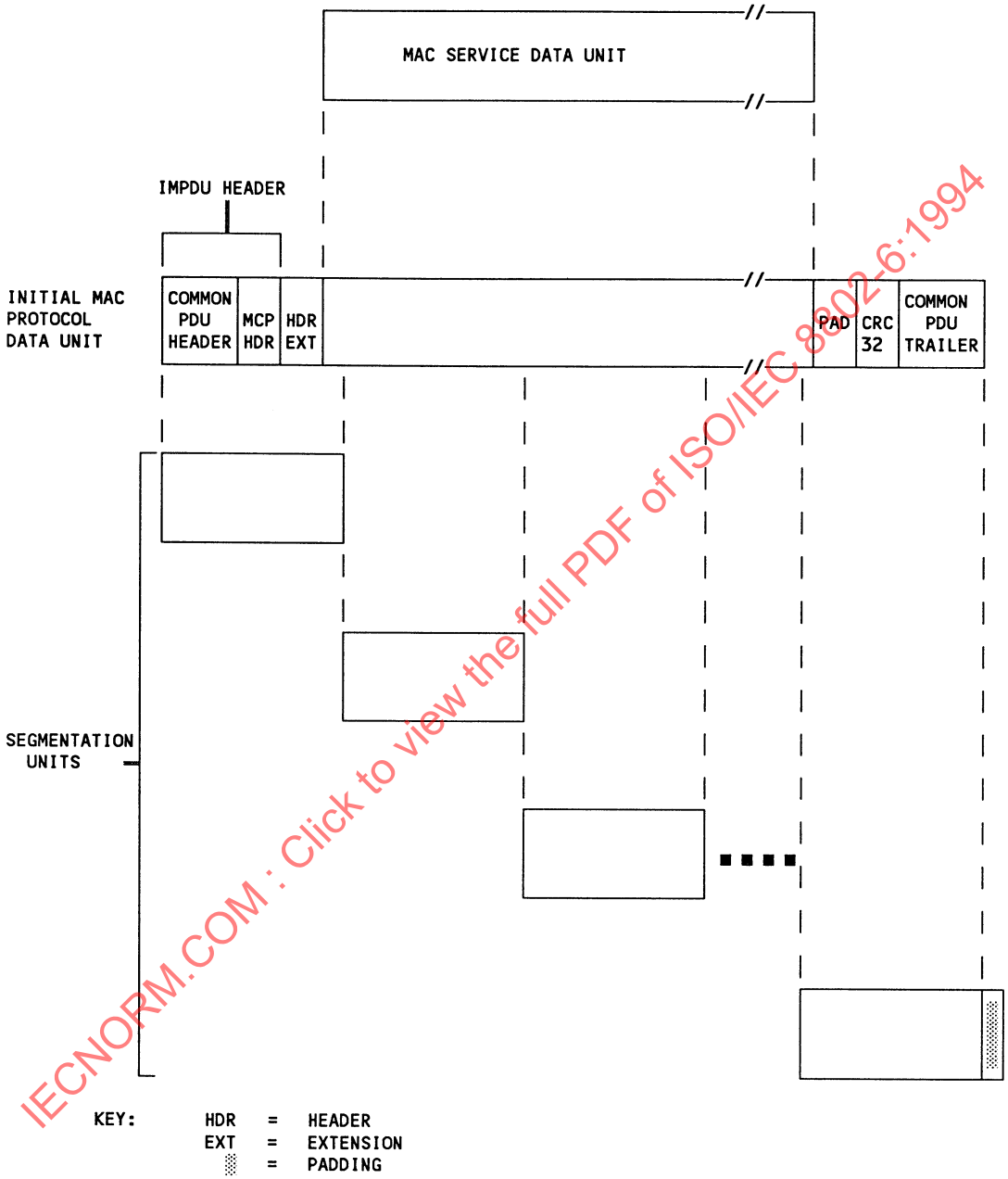


Figure 2-8—Segmentation of an IMPDU

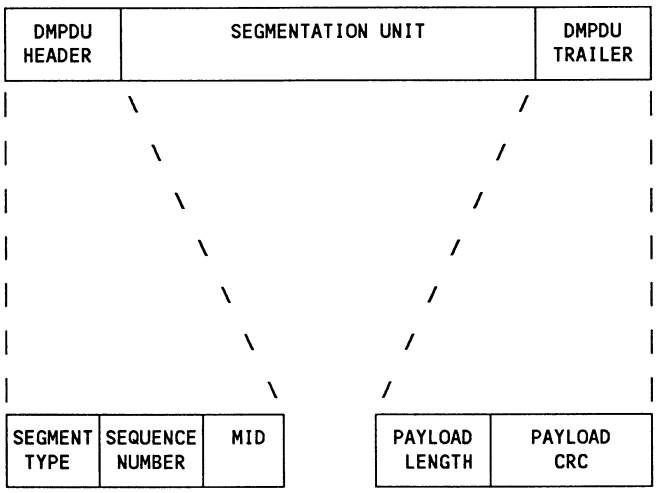


Figure 2-9—Format of a DMPDU

The MID identifies all of the DMPDUs derived from a single IMPDU, and is used in the reassembly of the segmented IMPDU at the destination. To describe the operation of this scheme, the segmentation of the IMPDU at the source is described first and the reassembly at the destination is considered later.

2.1.3.1.1 Segmentation at the source

The train of DMPDUs sent as QA segment payloads by the source is shown in figure 2-10. The first segmentation unit of a multi-segmentation unit IMPDU is carried in a Beginning of Message (BOM) DMPDU. This DMPDU is identified by the BOM code in the Segment Type subfield and signifies the start of a new IMPDU transfer. The MID subfield of this DMPDU carries one of the MIDs obtained by the source and not currently being used for the sending of another IMPDU by the source. The Sequence Number subfield of this DMPDU carries the initial value of the sequence of numbers to be associated with sequential DMPDUs. The segmentation unit of the BOM DMPDU will include the IMPDU header and any header extension plus the first octets of the MSDU, sufficient to fill the segmentation unit.

All subsequent segmentation units of the IMPDU until the last are placed in the payload field of the train of segments following the first segment. These DMPDUs are identified by the COM (Continuation of Message) code in the Segment Type subfield. The transfer of the multisegmentation unit IMPDU is completed by sending the last segmentation unit in a DMPDU that contains the EOM (End of Message) code in the Segment Type subfield. The COM and EOM DMPDUs carry the value of Sequence Number in the BOM DMPDU incremented by one for each successive DMPDU. All COM DMPDUs and the EOM DMPDU derived from an IMPDU carry the same MID value as the BOM DMPDU.

For the transfer of an IMPDU that only requires a single segmentation unit, the SSM (Single Segment Unit Message) code is used in the Segment Type subfield of the DMPDU. The MID is not used in this case, thus it is set to the reserved value of zero.

Each DMPDU trailer is constructed by writing into the Payload Length subfield the number of IMPDU octets used in the segmentation unit. For the connectionless MAC service to LLC, this number is always 44 for BOM and COM DMPDUs. The number written into EOM DMPDUs indicates the remaining number of octets in the IMPDU that need to be transferred. This number can be any multiple of 4 in the range 4 to 44, inclusive. The number written into SSM DMPDUs indicates the length of the entire IMPDU. This number

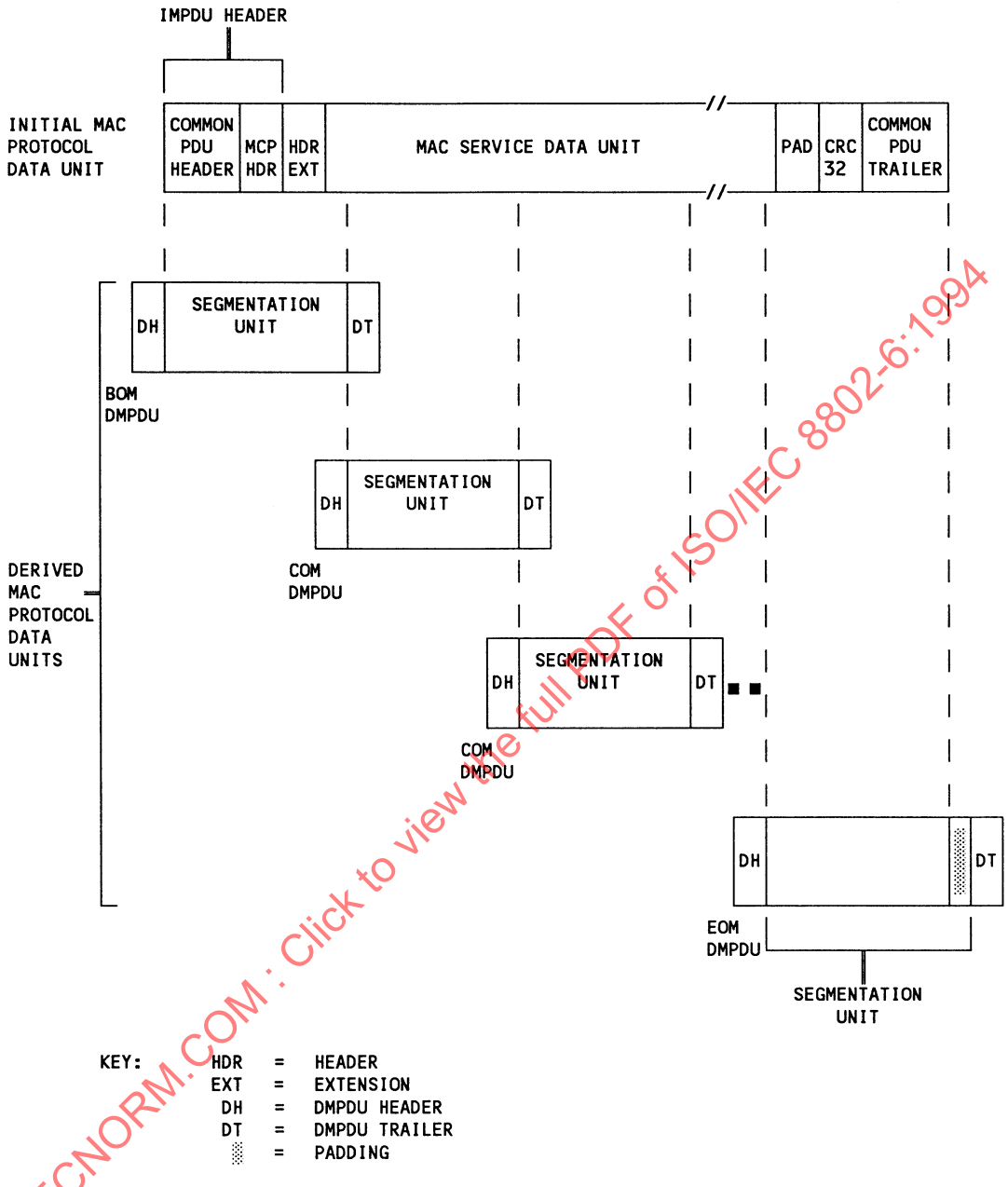


Figure 2-10—Transfer of an IMPDU

can be any multiple of 4 in the range 28 to 44, inclusive. The Payload CRC subfield is a CRC computed over all octets of the segment payload, including the DMPDU header, segmentation unit, and DMPDU trailer.

2.1.3.1.2 Reassembly at the destination

To receive IMPDUs segmented as described above, each AU will monitor all segments passing on the bus. All DMPDUs contain one of a particular set of VCI values in the segment header.[14] If the VCI value is one that the AU is programmed to receive, the AU will verify the DMPDU by means of the Payload CRC subfield. If the CRC verification fails then the DMPDU is discarded.

For each valid DMPDU with a BOM code in the Segment Type subfield, the AU will inspect the MCP header, which will be within the BOM segmentation unit. If the MCP header indicates that the IMPDU is addressed to the AU then it will copy the BOM segmentation unit and continue as described below. If the IMPDU is not addressed to the AU, the segmentation unit is not copied.

To receive the remainder of the IMPDU associated with the BOM segmentation unit, the AU will also record the sequence number and MID value from the BOM DMPDU. The following DMPDUs derived from the same IMPDU should all be received with the same VCI value in the segment header, and an incremented sequence number for each successive DMPDU, and the same MID value in the DMPDU header. The DMPDUs are recognized by the AU using the Payload CRC subfield to verify the DMPDU header of all DMPDUs received on the same VCI, and then comparing the verified MID value with the one recorded. When a match is made, the AU copies the verified segmentation unit of the DMPDU, provided the sequence number is the expected value. The complete IMPDU is received when a verified DMPDU with matching MID value, expected value of sequence number, and EOM code in the Segment Type subfield is received and the segmentation unit copied. Since segments are guaranteed to be delivered in order across the DQDB subnetwork, the AU can reassemble the received segmentation units into the original IMPDU by concatenating them in the order they were received.

The concatenation of all received segmentation units of an IMPDU is finally verified by using two pieces of information contained in the Common PDU header and Common PDU trailer. The length of the IMPDU, minus the length of the Common PDU header and Common PDU trailer, is sent in both the Common PDU header and Common PDU trailer. The length value received in the Common PDU trailer is compared against the number of octets received for the IMPDU. A mismatch causes the receiver to discard the IMPDU. This check is used to ensure that the correct number of DMPDUs have been received, and thus protects against loss or insertion of COM DMPDUs.

The second piece of information is a Beginning-End tag (BEtag). The same value of BEtag is sent in both the Common PDU header and Common PDU trailer of a given IMPDU. The BEtag value is incremented for the next IMPDU sent by the node. The two BEtag values for an IMPDU are compared at the receiver, and a mismatch causes the receiver to discard the IMPDU. The BEtag is used to ensure that the BOM DMPDU and the EOM DMPDU of a reassembled IMPDU were both actually derived from the same source IMPDU. This protects against loss of the EOM DMPDU from one IMPDU, loss of the BOM DMPDU from a subsequent IMPDU, and loss of the appropriate number of COM DMPDUs such that the IMPDU reassembled from the received DMPDUs is still of the length specified in the Common PDU trailer.

If the node supports checking of the IMPDU using the 32-bit CRC, and if the CRC32 field is present in the received IMPDU, as indicated by a bit in the MCP header, the AU will verify the IMPDU by means of the CRC32 field. If the CRC verification fails, then the IMPDU is discarded. If the CRC verification passes, the IMPDU is accepted as valid.

If all of these comparisons pass, then the IMPDU is accepted as valid.

---

[14] Note that the VCI value of all bits being set to one is reserved as a default value for MAC service to LLC. All conforming stations must recognize this VCI value and process the DMPDUs.

On the receipt of all DMPDUs of the IMPDU, the recorded MID value must be cleared because the source may reuse the same MID for a different IMPDU transfer.

To protect against the loss of the EOM DMPDU, which normally terminates the IMPDU transfer, a timer must be used to determine how long the MID value remains valid within the AU. If no EOM DMPDU with matching MID value is received within the timer period, the recorded MID is cleared and the received segmentation units associated with the MID are discarded.

With the above scheme, an AU can receive and reassemble multiple IMPDUs concurrently. This is achieved by recording multiple MID values for each VCI which the AU is programmed to receive and receiving all DMPDUs which match the recorded MID values on the appropriate VCIs.

Single segmentation unit IMPDUs are received in a similar manner to the BOM DMPDUs. The AU will verify the DMPDU using the Payload CRC subfield, inspect the MCP Header field in the segmentation unit, and copy the segmentation unit if the Payload CRC passes and the IMPDU is addressed to the AU. However, since with SSMs there are no following DMPDUs, there is no need to record the MID value. The IMPDU is completely received in this first DMPDU, and is then validated using the Length value in the Common PDU trailer, the BEtag values in the Common PDU header and Common PDU trailer, and possibly the CRC32 field as described above.

2.1.3.2 Provision of isochronous service

The Pre-Arbitrated access control mechanism does not necessarily accept or deliver isochronous service octets in an isochronous fashion. The variation from isochronous delivery occurs when the node at the head of bus does not generate Pre-Arbitrated slots in an isochronous fashion. If the user of the isochronous service requires the DQDB Layer to accept and/or deliver the octets isochronously, there is a requirement for some buffering. The nature of this buffering depends on the isochronous service user and is performed as part of an isochronous convergence function.

The isochronous service described in this part of ISO/IEC 8802 provides only the DQDB Layer functions required to access the medium to read and write isochronous service octets. This includes the Pre-Arbitrated Access functions within each AU and the periodic generation of the PA slots at the head of bus. The signaling procedures for establishing, maintaining, and clearing a connection are outside the scope of this part of ISO/IEC 8802. Similarly, the signaling procedures for indicating to the head of bus which PA slots are to be generated are also outside the scope of this part of ISO/IEC 8802.

However, annex B gives a tutorial description of the sequence of events that would be required to support complete isochronous communication capability. In particular, it explains the interactions that occur via the DQDB LMI and are defined in this part of ISO/IEC 8802.

2.1.3.3 Provision of Connection-Oriented Data Service

The Queued Arbitrated access control mechanism provides support for the transfer of QA segments. Therefore, support for the MAC service to LLC (described in 2.1.3.1) requires the definition of a convergence function.

Another convergence function is under study that will allow for the DQDB Layer to support a connection-oriented data service that uses Queued Arbitrated access. This will require some functions that differ from the convergence functions to support the MAC to LLC service. However, the connection-oriented data service will use a segmentation and reassembly procedure that is common with that used to support MAC service to LLC. In particular, it will use the same format for the Common PDU header and Common PDU trailer. It will also carry the same fields in QA segment payloads as are carried in the DMPDU header and DMPDU trailer.

This connection-oriented service will only be described in this standard in terms of the DQDB Layer functions required to read and write QA segments. The procedures for requesting, establishing, and clearing a connection are outside the scope of this part of ISO/IEC 8802.

2.1.4 Performance of the Distributed Queue

This clause provides guidance to network administrators in the application of this part of ISO/IEC 8802. DQDB subnetworks are intended to support a range of service types including statistical data traffic. Subnetworks should therefore be dimensioned by the network provider so as to ensure that periods of expected peak demands do not cause excessive delay to the particular services that are intended to be supported.

During the periods of overload that may still occur, a DQDB subnetwork will continue to operate at close to maximum throughput, but some service degradation must necessarily occur. To ensure manageable degradation of the services supported by a DQDB subnetwork, three mechanisms have been provided.

2.1.4.1 Pre-Arbitrated access

Pre-Arbitrated access enables bandwidth to be allocated to isochronous connections with constrained jitter. This service does not degrade with increasing load. Any bandwidth not Pre-Arbitrated for isochronous connections is available for Queued Arbitrated access.

2.1.4.2 Queued Arbitrated access with priority

Ideally, priority queueing enables selective degradation, under overload conditions, of lower priority services, allowing preferential access to higher priority delay-sensitive services. With bandwidth balancing disabled (i.e., BWB_MOD = 0), the Distributed Queue mechanism with priority, specified in this part of ISO/IEC 8802, can be relied upon to provide fair sharing of capacity and preferential access if the active stations accessing a given bus span a distance that is less than the equivalent of one 53-octet slot,[15] i.e., approximately the following:

2 km	at	44.736 Mbit/s
546 m	at	155.520 Mbit/s
137 m	at	622.080 Mbit/s

Beyond these distances, particularly under overload conditions, the effectiveness of the priority queueing mechanism decreases with increasing number of slots on the bus between contending stations.

2.1.4.3 Bandwidth balancing

Bandwidth balancing may be desirable in a DQDB subnetwork when the subnetwork physical configuration exceeds the distance-transmission speed product(s) stated in 2.1.4.2. An example of exceeding the aforementioned product is as follows: the distance between any two stations exceeds 546 m on a shared medium whose transmission rate is 155.52 Mbit/s.

The bandwidth balancing technique is used to ensure fair sharing of bandwidth between stations operating at a single priority. "Fair" in this context is defined as giving an approximately equal share of bandwidth to all stations attempting to access the medium for transmission. When the physical condi-tions are as stated above and when the offered load of all stations exceeds the bandwidth available on the medium, the use of bandwidth balancing allows all stations to receive an equal share of the bandwidth in the steady state. The performance during the transition period to steady state is dependent on the Bandwidth Balancing Modulus (BWB_MOD) and the distance between stations. In the steady state, the medium utilization is less than

[15] Assuming no station latency.

100%, being equal to BWB_MOD/(BWB_MOD+1) × 100% if there is only one active node. The steady-state medium utilization increases with the number of active nodes.

The bandwidth balancing mechanism divides bus bandwidth among the stations and allows some bandwidth to go unused. For example, for a bus with N stations, if all the following conditions are met:

— No station has any Pre-Arbitrated traffic.
— Each station always has Queued Arbitrated segments waiting to be transmitted on the bus.
— All segments have the same priority.
— The value of the BWB_MOD at each station is M (this means that each station uses a fraction M/(M+1) of the slots not used by the other stations).
— These load conditions persist for a sufficiently long time.

Then the bandwidth balancing mechanism provides each station with a steady-state average throughput of 1/(N+1/M) segments per slot time,[16] and the total utilization of the bus in steady state is N/(N+1/M) segments per slot time. Note that the total utilization increases as the BWB_MOD increases and as the number of active stations increases. The station throughputs approach their steady-state values gradually. The convergence is faster if the BWB_MOD is smaller.

This International Standard contains three REQ bits in the ACF octet, each one designating a different priority level for station access to the medium. However, the metric for judging the performance of the multiple priority access case is undefined in the initial version of this part of ISO/IEC 8802, and is for further study. Therefore, the performance of the multiple priority access mechanism with bandwidth balancing is also for further study.

If the distance-transmission speed product for the subnetwork indicates that bandwidth balancing is desirable, and if the network operator wishes to give higher priority to certain classes of messages, then the following guideline should be followed: it is recommended that all stations on the subnetwork that send high-volume traffic do so at the lowest priority. Only stations which send low-volume traffic that is delay sensitive (e.g., management messages) should do so at a higher priority.

2.1.5 DQDB subnetwork configuration control

An extension of the Dual Bus architecture is the looped bus topology shown in figure 2-11. The only change in this from the open Dual Bus topology shown in figure 2-1 is that the end points of the buses are collocated. However, data does not flow from the end of a bus through to the head of that bus. Thus, while the looped bus topology may appear similar to a ring, it has no logical similarity to it.

The looping of the DQDB bus architecture is important in that it permits a subnetwork reconfiguration capability in the presence of bus faults. In the case of a bus fault, the subnetwork can isolate the fault and close the data buses through the head point of the loop. Hence, the subnetwork is reconfigured and fully operational in the reconfigured state.

The healing mechanism is depicted in figure 2-12 and shows how physical faults or line breaks are healed by repositioning the natural break in the loop to the position of the break due to failure. This reconfiguration action effectively moves the Head of Bus functions to the nodes adjacent to the break. If a node adjacent to the break does not support the head of bus capability, then reconfiguration will be completed by the node closest to the break with that capability. In this case, the node without head of bus capability will not maintain communication connectivity with the remainder of the subnetwork.

[16] To understand this formula, recall that the throughput r of a station should equal a fraction M/(M+1) of the bandwidth not used by the other N–1 stations: $r = (M/(M+1)) \times (1-(N-1) \times r)$ segments per slot time. Solving for r gives the stated result.

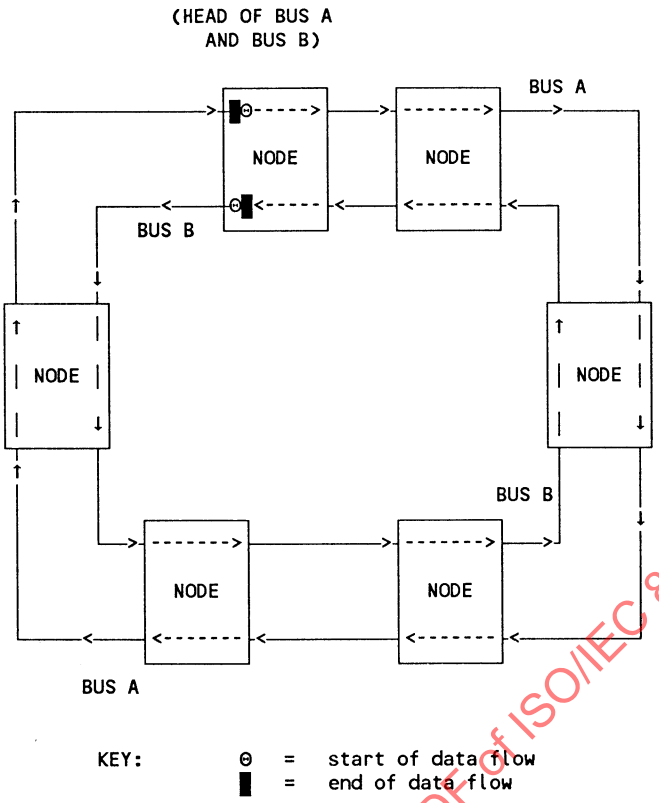


Figure 2-11—Looped bus topology

The subnetwork reconfiguration scheme can also handle single bus failures in the case of an open bus topology or multiple bus failures for both the open or looped topology. However, bus failures in either of these cases will result in the subnetwork being split into isolated islands, so that full connectivity cannot be maintained.

2.2 Node functional architecture

The functional architecture for a DQDB node is shown in figure 2-13. It consists of two layers: the Physical Layer and the DQDB Layer. The DQDB Layer uses the services of the Physical Layer to provide a number of different services. One of these services is the MAC Sublayer service to the LLC Sublayer. Other services, which are under study, include a range of connection-oriented data services and a range of isochronous services.

The following clauses describe the functions of a DQDB node, that is, of the Physical Layer and the DQDB Layer.

2.2.1 Physical Layer functions

The generic Physical Layer contains three functional entities: the transmission system, the Physical Layer convergence function, and the Layer Management functions. Each of these functions is described in the following subclauses.

The Physical Layer service is provided to the DQDB Layer entity at a node through two SAPs. Each SAP is associated with one duplex transmission link connecting the node to an adjacent node.

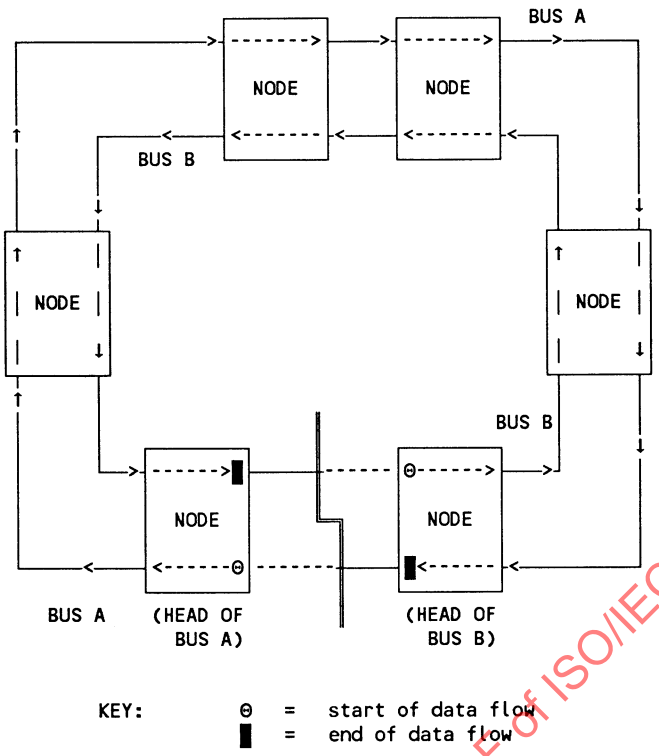


Figure 2-12—Subnetwork reconfiguration

2.2.1.1 Transmission system

The transmission system function provides a standard transmission interface used to access the transmission link between adjacent nodes. Standard public network transmission systems identified as suitable to provide this function are those specified in CCITT Recommendation G.703, the DS3 rate specified in T1.102, T1.107, and CCITT Recommendations G.707, G708, and G.709 SDH. Other transmission systems may also be supported in the future.

2.2.1.2 Physical Layer Convergence function

In order to allow the DQDB Layer to operate independently of the nature of the transmission system, a Physical Layer Convergence function is used. This function provides the standard Physical Layer to DQDB Layer service, irrespective of the nature of the underlying transmission system. Therefore, each different transmission system will require the definition of a unique Physical Layer Convergence Procedure (PLCP). This part of ISO/IEC 8802 will define the PLCP for each standard transmission interface supported. If the transmission system already provides the defined Physical Layer service, the PLCP would be a null function.

In the future this part of ISO/IEC 8802 may also define a PLCP that does not use an existing standard transmission system, but which defines this function as part of the protocol. This approach may be suitable for private backbone networks using the DQDB access mechanism.

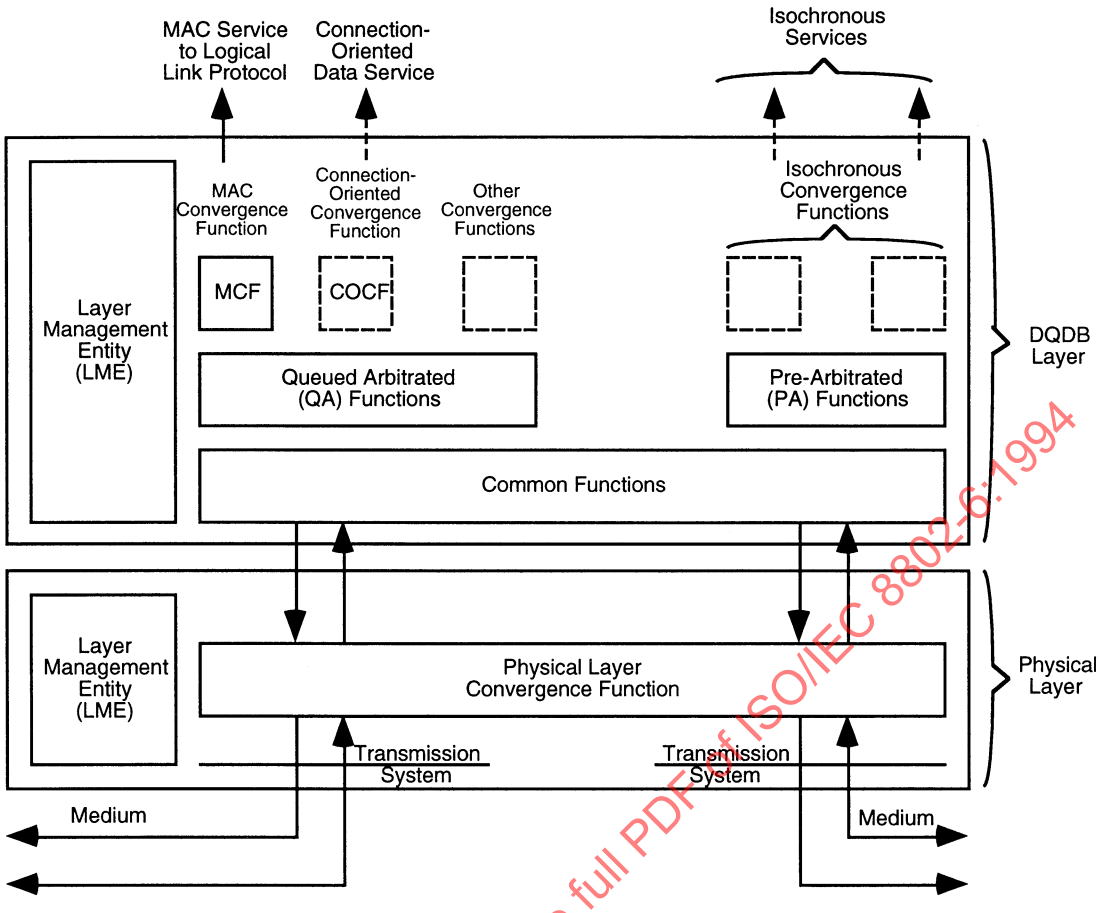


Figure 2-13—DQDB node functional architecture

2.2.1.3 Physical Layer Management Entity (LME)

The Physical LME performs management of the local Physical Layer functions. The Physical LME also provides the Physical Layer Management Interface (LMI) to the Network Management Process (NMP) for the remote management of the local physical subsystem.

2.2.2 DQDB Layer functions

Within the DQDB Layer there are four principal types of functions: the common functions, the access control functions (Queued Arbitrated and Pre-Arbitrated), the convergence functions, and the Layer Management functions. Each of these functions is described below.

2.2.2.1 Common Functions

The Common Functions block acts as a DQDB Layer relay for the transfer of slots and management information octets between the two SAPs to the local Physical Layer entity. Thus, the Common Functions block allows the QA Functions block and PA Functions block to gain read and write access to the QA and PA slots.

The Common Functions block also operates on the slots and management information octets to support functions that operate at some or all of the nodes in the subnetwork. These functions include the Head of Bus function, the Configuration Control function, and the MID Page Allocation function, and are described in the following subclauses.

The Configuration Control and MID Page Allocation functions manage DQDB Layer objects necessary for nodes to communicate on the subnetwork. Hence, these functions are part of the DQDB LME. However, there is a requirement for coordination of the DQDB LMEs at all nodes to support these functions. The Common Functions block of the DQDB Layer supports the DQDB Layer Management Protocol for coordination of the Configuration Control and MID Page Allocation functions.

All or some of the common functions within the DQDB Layer can be shared across several nodes.

2.2.2.1.1 Head of Bus function

The Head of Bus function is performed by the node at the head of each bus and by no other nodes on the Dual Bus. It includes the Slot Marking function, which is the process of creating empty slots that are to be written onto the bus. This includes the marking of PA slots and the writing of the VCI in the PA segment header. The node at the head of each bus must also write appropriate values into the management information octets.

The Relay function performed by the Common Functions block operates even if the node is performing the Head of Bus function in an open Dual Bus subnetwork, as the Physical Layer entity is the source of all timing in the node. If a node is at the head of an open Dual Bus subnetwork, then the Physical Layer delivers empty slots and empty management information octets to the appropriate Physical Layer service access point. As the DQDB access mechanism operates by logical OR-writing, empty octets are conveyed with all bits set to zero. These slots and management information octets are written into, if required, by the Head of Bus function at that node. Any Slot Marking operation occurs before the slot is accessed by the QA or PA functions block within the node at the head of bus.

2.2.2.1.2 Configuration Control function

The Configuration Control function is responsible for ensuring that the resources of all nodes of a subnetwork are configured into a correct Dual Bus topology, which may be looped or open. The resources that are managed include Head of Bus functions and subnetwork timing reference functions. Therefore, the Configuration Control function will be employed at subnetwork start-up to correctly configure the subnetwork. The Configuration Control function will also reconfigure the subnetwork in the case of bus failures, as described in 2.1.5, or failures in the external timing reference for the subnetwork, as described in 2.1.1. The function will also return the subnetwork to its original configuration when the failure is reliably repaired. An example of the operation of the Configuration Control function is the activation and deactivation of the Head of Bus functions at appropriate nodes during the process of reconfiguration, as described in 2.1.5.

2.2.2.1.3 MID Page Allocation function

The MID Page Allocation function participates in a distributed protocol with all nodes on the subnetwork to control the allocation of the MID values to nodes. The MID values are used by the node in the transfer of multiple segmentation unit IMPDUs, described in 2.1.3.1. The MID Page Allocation function will ensure that no two nodes are allocated the same MID value.

2.2.2.2 Queued Arbitrated (QA) functions

The QA functional entity provides an asynchronous data transfer service for 48-octet segment payloads. The QA functional entity accepts the segment payloads from a convergence function, and adds the appropriate segment header, including VCI, to the segment payload to create a QA segment. The QA segment is queued for access to the Dual Bus by use of the Distributed Queueing function. QA segments received by the QA functional entity are stripped of the segment header and the payload is passed to the correct convergence function, based on the VCI value in the segment header.

2.2.2.3 Pre-Arbitrated (PA) functions

The PA functional entity provides access control for the connection-oriented transfer over a guaranteed bandwidth channel of octets which form part of an octet stream. The operation of the PA functional entity requires the previous establishment of a connection. As a result of the connection establishment, the PA functional entity will be informed of the VCI value for segments used in the connection and the offset of the octets to be used for reading and writing within the multiple user PA segment payload.

The PA entity accepts the single octets from a convergence function and writes them into the pre-allocated positions within the payload of PA segments with the appropriate VCI value in the segment header. The VCI value will have been set by the Slot Marking function at the head of bus.

To receive an octet stream, the PA functional entity, on receiving a PA segment with the correct VCI value, will copy octets from the pre-allocated positions within the segment payload. The octets are passed to the correct convergence function, based on the VCI value in the segment header and the offset of the octet in the PA segment payload.

NOTE—The PA functional entity can support the transfer of more than one octet from the same source in a given segment in order to support higher rate services.

2.2.2.4 Convergence functions

It is intended that the DQDB Layer will provide a range of services including connectionless data transfer, isochronous data transfer, and connection-oriented data transfer. These services are provided by convergence functions placed above the QA and PA functional entities. This part of ISO/IEC 8802 specifies the convergence function to support the connectionless MAC data service to LLC and gives guidelines for the provision of an isochronous service. The connection-oriented data service is under study.

The operation of these convergence functions is described below.

2.2.2.5 MAC Convergence Function (MCF)

The MCF is responsible for adapting the segment-payload-based service provided by the QA functional entity to the standard MAC service required by the LLC Sublayer. The convergence function is realized primarily by the implementation of the segmentation and reassembly protocol, which is described in detail in 2.1.3.1.

The MCF transmit process involves encapsulating the LLC Protocol Data Unit (MAC Service Data Unit) to form an Initial MAC Protocol Data Unit (IMPDU). The IMPDU is segmented into segmentation units of 44 octets, as described in 2.1.3.1.1. Each segmentation unit has a segment type, sequence number, and MID value prepended, and a payload length and payload CRC appended, to form a Derived MAC Protocol Data Unit (DMPDU). This can then be transferred by the QA functional entity.

The MCF receive process involves the destination reassembling the original IMPDU from the received DMPDUs, as described in 2.1.3.1.2. The LLC Protocol Data Unit is extracted from the received IMPDU and passed to the LLC entity.

2.2.2.5.1 Isochronous Convergence Function (ICF)

There is one ICF for each Isochronous Service User (ISU). An ICF is responsible for adapting the guaranteed-bandwidth octet-based service provided by the PA functional entity to an isochronous octet-based service. Therefore, the primary function of the ICF is buffering to allow for instantaneous rate differences between the PA service and the provided isochronous service.

In particular, the ICF will provide buffering both for the isochronous octets to be transmitted on behalf of the ISU by the PA functional entity and for the isochronous octets received from the PA functional entity and to be delivered to the ISU. In the establishment of a connection that will require use of an ICF, it is necessary to specify the rate of the isochronous service and the possible variability in delivery of octets over the PA channel. This is necessary to determine the size of the buffer required in the ICF.

2.2.2.5.2 Other convergence functions

The provision of a connection-oriented data service by the DQDB Layer is under study. The Connection-Oriented Convergence Function (COCF) will adapt the segment-payload-based service provided by the QA functional entity to a connection-oriented data service with the following characteristics:

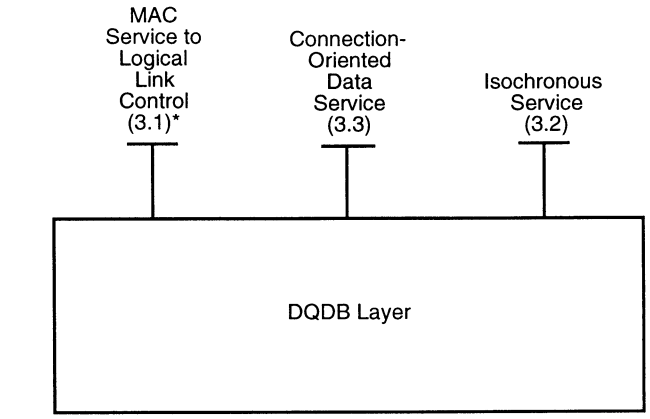
— It uses the same segmentation and reassembly procedures as the MCF.
— It is 32-bit aligned.
— It supports pipelining.
— It has protection on any control fields.
— It has a facility to allow buffer allocation at the receiver.

2.2.2.6 DQDB Layer Management Entity (LME)

The DQDB LME performs management of the local DQDB Layer functions. It also communicates with the DQDB LMEs at other nodes to provide distributed management of the DQDB Layer resources. The communication uses a DQDB Layer Management Protocol supported by the DQDB Layer Common functions block (see 2.2.2.1). The DQDB LME also provides the DQDB Layer Management Interface to the Network Management Process for the remote management of the local DQDB Layer subsystem.

3. DQDB Layer service definition

A DQDB subnetwork of a MAN is capable of supporting a broad range of services and applications. This clause describes the services currently defined and provided by the DQDB Layer. (See figure 3-1.)



*Numbers in parentheses refer to clauses and subclauses of this part of ISO/IEC 8802 where the services are defined.

Figure 3-1—DQDB Layer services

The DQDB Layer services are as follows:

a) *The MAC service provided to the LLC Sublayer.* The LLC Sublayer, operating over the DQDB Layer, provides the service of the OSI Data Link Layer and thus supports data communications between two open systems.

b) *Isochronous Service, provided to an Isochronous Service User (ISU).* This service supports the transfer of isochronous service octets with a constant interarrival time over an isochronous connection.

c) *A connection-oriented data service that supports the transfer of data over virtual channels.* This service is asynchronous because there is no guarantee of a constant interarrival time for data units.

These services are defined using the service primitives notation specified in ISO/TR 8509. This notation provides an abstract description and does not imply any particular implementation or exposed interface.

3.1 MAC service provided to the LLC Sublayer

The MAC service to the LLC Sublayer is defined in ISO/IEC 10039. It is a connectionless service that supports the transfer of variable length MAC Service Data Units (MSDUs) between LLC Sublayer peer entities, without the need for the LLC entities to request the establishment of a connection between them. There is no guarantee of delivery of the MSDUs by the MAC service.

The MAC service primitives and their parameters cited in this clause are those specified in ISO/IEC 10039. These primitives are as follows:

— MA-UNITDATA request
— MA-UNITDATA indication

3.2 Isochronous service

The isochronous service supports the transfer of isochronous service octets between two ISUs over an already established isochronous connection. The service is described as isochronous because there is a constant interarrival time for the service octets.

This part of ISO/IEC 8802 specifies the isochronous service provided by the DQDB Layer once an isochronous connection is established. The procedure for establishing, maintaining, and releasing an isochronous connection is outside the scope of this part of ISO/IEC 8802, but it will be specified by other standards. (Refer to annex B.)

For consistency in notation, the isochronous service is described abstractly by means of the service primitives notation defined in ISO/TR 8509. However, it should be noted that this technique does not efficiently capture the important timing properties of the interactions between the isochronous service and a user of the service.

Further, the abstract description does not constrain an implementation in any way. For example, an implementation might pass a number of octets simultaneously, if this meets the requirements of the ISU.

The primitives used to describe the service are the following:

ISU-DATA request
ISU-DATA indication

Isochronous service octets to be sent via the isochronous service are presented by the ISU at the specified rate for the isochronous connection. Similarly, the isochronous service periodically delivers an isochronous service octet to an ISU at the rate specified for the connection.

3.2.1 ISU-DATA request

Function:

This primitive requests the transfer of an isochronous service octet over an established isochronous connection.

Semantics of the Service Primitive:

```
ISU-DATA request     (
                     ISDU
                     )
```

The ISDU parameter conveys a single isochronous service octet.

When Generated:

This primitive is generated by an ISU whenever an isochronous service octet must be transferred over a connection.

Effect on Receipt:

The receipt of this primitive by the DQDB Layer results in the DQDB Layer attempting to transfer the isochronous service octet over the established connection, as described in 5.2.1.1. Isochronous service octets are transferred in the same order in which they are submitted by the ISU.

Additional Comments:

A possible logical sequence of primitives associated with successful isochronous service octet transfer is illustrated in figure 1-3c), which shows that the ISU-DATA request from the local ISU results in an ISU-DATA indication to the ISU associated with the other end of the isochronous connection.

3.2.2 ISU-DATA indication

Function:

This primitive indicates the arrival of an isochronous service octet over an established isochronous connection.

Semantics of the Service Primitive:

ISU-DATA indication	(
	ISDU
	)

The ISDU parameter conveys a single isochronous service octet.

When Generated:

This primitive is generated by the DQDB Layer to deliver an isochronous service octet that has arrived over an established isochronous connection.

Effect on Receipt:

The effect of receipt of this primitive is dependent upon the ISU.

Additional Comments:

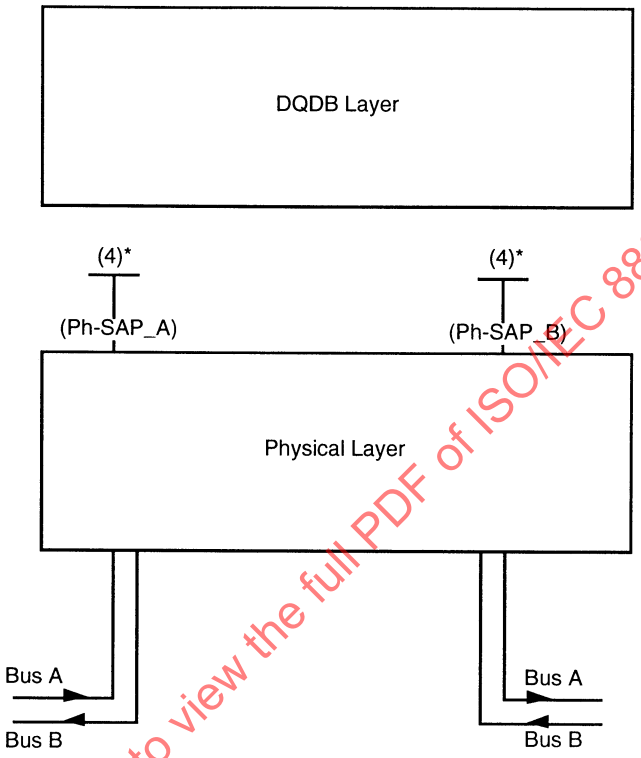
None.

3.3 Connection-Oriented Data Service

The Connection-Oriented Data Service supports a virtual channel between a pair of Connection-Oriented Data Service Users. The details of the service and the service primitives used to represent it are still under study (see also 5.3.1, which describes functions of the DQDB Layer under study for the support of the Connection-Oriented Data Service).

4. Physical Layer service definition

Each node that is capable of communicating with other nodes and is not performing the Head of Bus function in an open Dual Bus subnetwork is connected to the two adjacent nodes by separate active duplex transmission links.[17] The Physical Layer entity at each node provides two Physical Layer Service Access Points (Ph-SAPs) to the local DQDB Layer entity, with each duplex transmission link being associated at each node with one Ph-SAP.[18] (See figure 4-1.)



*Numbers in parentheses refer to clauses and subclauses of this part of ISO/IEC 8802 where the services are defined.

Figure 4-1—Physical Layer services (at each node)

This clause describes the services provided by the Physical Layer at each service access point to the DQDB Layer. These services are defined using the service primitives notation specified in ISO/TR 8509. This notation provides an abstract description and does not imply any particular implementation or exposed interface. However, it should be noted that this technique does not capture the 125 μs timing properties of the interactions between the DQDB Layer and the Physical Layer.

---

[17] A node that is not capable of supporting Head of Bus functions and is not connected to both adjacent nodes by active duplex transmission links is not capable of communicating with any other node. A node performing the Head of Bus function in an open Dual Bus subnetwork will be connected to only one adjacent node by an active duplex transmission link.

[18] If the node is performing the Head of Bus function in an open Dual Bus subnetwork, then the duplex transmission link, which is either not active or not physically present, is still associated with a service access point. Similarly, if the node is not capable of supporting Head of Bus functions, then a duplex transmission link, which is not active at the node, is also associated with a service access point. Hence the Physical Layer entity at all nodes still provides two Ph-SAPs to the DQDB Layer entity. This is done because the Physical Layer is the source of all timing at a node.

The following primitives are defined:

— Ph-DATA request
— Ph-DATA indication
— Ph-TIMING-SOURCE request
— Ph-TIMING-MARK indication
— Ph-STATUS indication

The Ph-SAP associated with the transmission link where Bus A enters the node is designated as Ph-SAP_A. The Ph-SAP associated with the transmission link where Bus B enters the node is designated as Ph-SAP_B. The association of Ph-SAPs with transmission links is established during network initialization, as described in clause 10.

4.1 Ph-DATA request

Function:

This primitive requests the transfer of an octet from the local DQDB Layer entity to the peer DQDB Layer entity.

Semantics of the Service Primitive:

Ph-DATA request (
octet,
type,
[status]
)

The representation and transfer of the octet shall be consistent with 6.1. All possible combinations of eight bits are valid. The type parameter specifies the type of the octet. The values that can be associated with the type parameter are the following:

SLOT_START, or
SLOT_DATA, or
DQDB_MANAGEMENT

The value SLOT_START specifies that the octet is the first in a slot, i.e., the Access Control Field (ACF). The value SLOT_DATA specifies any subsequent octet of a slot. The value DQDB_MANAGEMENT specifies that the octet is used to carry DQDB Layer Management Protocol information. The optional status parameter provides an indication to the Physical Layer that the octet presented for transfer is either valid or invalid. The values that can be associated with the status parameter, if provided, are the following:

VALID, or
INVALID

When Generated:

The local DQDB Layer entity generates a Ph-DATA request whenever an octet must be transferred to the peer DQDB Layer entity. This occurs upon receipt of an appropriate Ph-DATA indication, as defined in 4.3.

A Ph-DATA request with SLOT_START type will (in normal operation) be followed by exactly 52 Ph-DATA request primitives with SLOT_DATA type before the next Ph-DATA request with SLOT_START type occurs. The occurrence of Ph-DATA request primitives with DQDB_MANAGEMENT type with

respect to Ph-DATA request primitives with SLOT_START or SLOT_DATA type is a function of the Physical Layer protocol.

Effect on Receipt:

The effect on receipt of this primitive is one of the following cases:

a) If the transmission link associated with a Ph-SAP has a status of UP, as signaled in the last Ph-STATUS notification at that Ph-SAP, or if the transmission link associated with a Ph-SAP has a status of DOWN and the type of the octet is DQDB_MANAGEMENT, then receipt of this primitive causes the local Physical Layer entity to attempt to transmit the octet to the peer Physical Layer entity over the transmission link.

b) If the transmission link associated with a Ph-SAP has a status of DOWN, as signaled in the last Ph-STATUS indication at that Ph-SAP, and the type of the octet is either SLOT_START or SLOT_DATA, then the effect on receipt of this primitive is unspecified.

Additional Comments:

A possible logical sequence of primitives associated with a successful octet transfer is illustrated in figure 1-3c), which shows that the Ph-DATA request at the local Ph-SAP results in a Ph-DATA indication at the Ph-SAP associated with the other end of the transmission link. This corresponds to Case 1 under *Effect on Receipt*. The receipt of this primitive when the Ph-SAP is not associated with an active duplex transmission link, as described in Case 2 under *Effect on Receipt* (and possibly Case 1 if the type of the octet is DQDB_MANAGEMENT), is illustrated in figure 1-3a).

The relationship between Ph-DATA indication primitives and Ph-DATA request primitives at a node is defined in 4.3.

The status parameter sent in a Ph-Data request primitive will only be set to INVALID if this value is relayed from the Ph-DATA indication primitive in which the associated octet was delivered.

4.2 Ph-DATA indication

Function:

If a node is connected to another node by an active duplex transmission link associated with the Ph-SAP, then this primitive indicates the arrival of an octet from the peer Physical Layer entity. If a node is not connected to another node by an active duplex transmission link associated with the Ph-SAP and the node is capable of providing the Head of Bus function, or the node is providing the function of both head of Bus A and head of Bus B, then this primitive indicates the local delivery of an EMPTY octet to the Head of Bus function at the node.

Semantics of the Service Primitive:

```
Ph-DATA indication      (
                        octet,
                        type,
                        [status]
                        )
```

The representation and transfer of the octet shall be consistent with 6.1. All possible combinations of eight bits are valid. In addition, the octet value designated EMPTY, used to carry octet type and timing informa-

tion to the DQDB Layer at the head of a bus, is defined to have all eight bits set to zero. The type parameter specifies the type of the octet. The values that can be associated with the type parameter are the following:

SLOT_START, or
SLOT_DATA, or
DQDB_MANAGEMENT

The value SLOT_START specifies that the octet is the first in a slot, i.e., the ACF. The value SLOT_DATA specifies any subsequent octet of a slot. The value DQDB_MANAGEMENT specifies that the octet is used to carry DQDB Layer Management Protocol information. The optional status parameter provides an indication from the Physical Layer that the octet delivered is either valid or invalid. The values that can be associated with the status parameter, if provided, are the following:

VALID, or
INVALID

When Generated:

The local Physical Layer entity generates this primitive in one of the following cases:

a) If the transmission link associated with a Ph-SAP has a status of UP, as signaled in the last Ph-STATUS indication at that Ph-SAP, then this primitive is generated whenever an octet is received from the peer Physical Layer entity over the transmission link. The Ph-DATA indication contains the value and type of the received octet. The octet value in this case will not be EMPTY. If the Physical Layer supports the optional status parameter, then the Ph-DATA indication contains the status of the received octet.

b) If the transmission link associated with a Ph-SAP has a status of DOWN, as signaled in the last Ph-STATUS indication at that Ph-SAP, and the node is capable of providing the Head of Bus function, or the node is providing the function of both head of Bus A and head of Bus B, then this primitive is generated in accordance with the requirements of the Physical Layer. The Ph-DATA indication contains an EMPTY octet of the type required by the Physical Layer, and is used to convey both octet timing and octet type information. Note that for nodes providing the function of both head of Bus A and head of Bus B, there is no relationship at Ph-SAP_A between Ph-DATA indications with EMPTY octets and Ph-DATA indications with non-EMPTY octets. Ph-DATA indications with EMPTY octets are generated by the Physical Layer when octets need to be sent out at the head of Bus A and head of Bus B. Ph-DATA indications with non-EMPTY octets are generated by the Physical Layer when an octet is received from either transmission link from the adjacent nodes.

A Ph-DATA indication with SLOT_START type will (in normal operation) be followed by exactly 52 Ph-DATA indication primitives with SLOT_DATA type before the next Ph-DATA indication with SLOT_START type occurs. The occurrence of Ph-DATA indication primitives with DQDB_MANAGEMENT type with respect to Ph-DATA indication primitives with SLOT_START or SLOT_DATA type is a function of the Physical Layer protocol.

Effect on Receipt:

The effect on receipt of this primitive is dependent upon the type of the octet and the operation of the DQDB Layer. Upon receipt of a Ph-DATA indication, the local DQDB Layer entity will generate either zero or one Ph-DATA request primitives at each of the two Ph-SAPs to the local Physical Layer entity, as defined in 4.3. The effect of receipt of a status indication is not specified.

Additional Comments:

The generation of this primitive upon receipt of an octet from a peer Physical Layer entity, as described in Case 1 under *When Generated,* corresponds to the indication primitive shown in figure 1-3c). The local generation of this primitive when there is no peer Physical Layer entity, as described in Case 2 under *When Generated,* is illustrated in figure 1-3b). This primitive is not generated if the transmission link associated with a Ph-SAP has a status of DOWN, as signaled in the last Ph-STATUS indication at that Ph-SAP, and the node is not capable of providing the Head of Bus function.

The relationship between Ph-DATA indication primitives and Ph-DATA request primitives at a node is defined in 4.3.

4.3 Relationship of Ph-DATA request primitives and Ph-DATA indication primitives

If a node is not capable of providing Head of Bus functions, and either of the transmission links at the node have a status of DOWN, as signaled in the last Ph-STATUS indication at the respective Ph-SAP, then the node is not capable of communicating with other nodes. In this case, Ph-DATA indi-cation primitives are not generated at the Ph-SAP associated with the transmission link that is down.

In all other cases, there is a relationship between the generation of Ph-DATA indication primitives at each Ph-SAP of a node and the receipt of Ph-DATA request primitives at both Ph-SAPs of the node. The relationship depends on whether or not the node is performing the function of head of a bus, and is outlined in 4.3.1 and 4.3.2.

4.3.1 Node not performing Head of Bus functions

If a node is capable of communicating with other nodes and is not performing the function of head of either bus, then both Ph-SAPs are associated with an active duplex transmission link.[19] Receipt of a Ph-DATA indication at either Ph-SAP of this node will result, after the octet is relayed through the DQDB Layer, in the generation of a Ph-DATA request at the other Ph-SAP, as shown in figure 4-2.

The Ph-DATA request at one Ph-SAP shall contain the octet received in the Ph-DATA indication at the other Ph-SAP, as modified by the DQDB Layer in accordance with the following rules:

a) The type of the octet shall not be modified.
b) For octets of type SLOT_START and SLOT_DATA, the value of each bit in the octet shall be modified according to the DQDB access mechanism. The set of permitted operations on each bit of the octet, as seen at the respective Ph-SAPs, is specified in table 4-1.
c) For octets of type DQDB_MANAGEMENT, the value of each bit in the octet shall be modified according to the rules of the DQDB Layer Management Protocol.

4.3.2 Node performing Head of Bus functions

If a node is performing the function of head of bus, then three cases occur, as described in the following three subclauses.

4.3.2.1 Node performing Head of Bus A function in an open subnetwork

If the node is performing the Head of Bus A function in an open Dual Bus subnetwork, then Ph-SAP_B at that node is associated with an active duplex transmission link, and Ph-SAP_A is not. Receipt of a Ph-DATA

[19] A node that is not capable of supporting Head of Bus functions and is not connected to both adjacent nodes by active duplex transmission links is not capable of communicating with any other node.

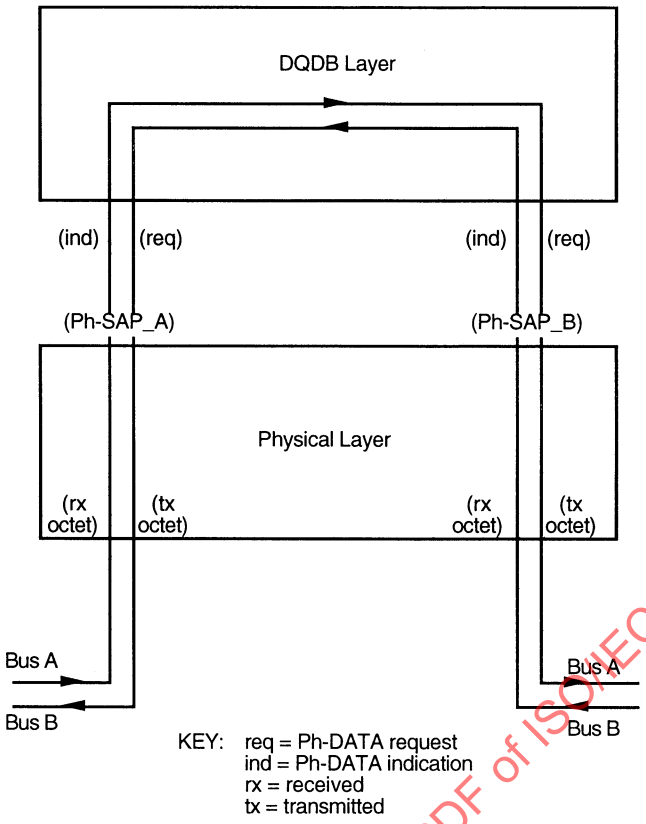


Figure 4-2—Relationship between Ph-DATA indication and Ph-DATA request: Node performing Head of Bus A functions

Table 4-1—Permitted operations on bits of octets: Node not performing Head of Bus functions

Bit in Ph-DATA indication (Ph-SAP_A/B)	Bit in Ph-DATA request (Ph-SAP_B/A)	Operation permitted?
0	0	Yes
0	1	Yes
1	0	No
1	1	Yes

indication with an octet of type SLOT_START or SLOT_DATA at Ph-SAP_B allows the octet to be accessed by the DQDB Layer, but does not lead to the generation of a Ph-DATA request at either Ph-SAP_A or Ph-SAP_B. Receipt of a Ph-DATA indication with an octet of type DQDB_MANAGEMENT at Ph-SAP_B allows the octet to be accessed by the DQDB Layer, and leads to the generation of a Ph-DATA request at Ph-SAP_A.

The Physical Layer generates EMPTY octets in Ph-DATA indication primitives at Ph-SAP_A, as defined in 4.2, *When Generated,* Case 2. The EMPTY octets are conveyed to the Head of Bus A function in the node

and, after then being relayed through the DQDB Layer, lead to the generation of a Ph-DATA request at Ph-SAP_B. This process is depicted in figure 4-3.

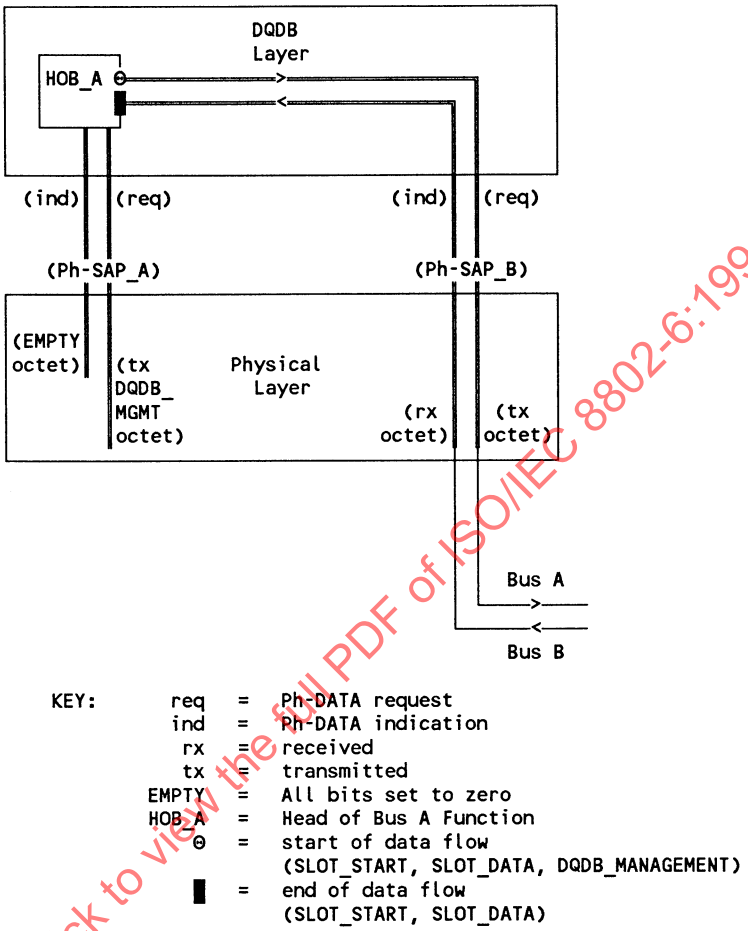


Figure 4-3—Relationship between Ph-DATA indication and Ph-DATA request: Node performing Head of Bus A function

The Ph-DATA request at Ph-SAP_B shall contain the octet received in the Ph-DATA indication at Ph-SAP_A, as modified by the DQDB Layer in accordance with the following rules:

a) The type of the octet shall not be modified.
b) For octets of type SLOT_START and SLOT_DATA, the value of each bit in the octet shall be modified according to the DQDB access mechanism.
c) For octets of type DQDB_MANAGEMENT, the value of each bit in the octet shall be modified according to the rules of the DQDB Layer Management Protocol. Note that the Head of Bus function shall set the TYPE bit alternately to (TYPE = 0) and (TYPE = 1) for successive DQDB_MANAGEMENT octets. (See 10.1.)

A Ph-DATA request with an octet of type DQDB_MANAGEMENT at Ph-SAP_A shall contain the octet of the same type received in the Ph-DATA indication at Ph-SAP_B, as modified according to the DQDB Layer Management Protocol rules.

4.3.2.2 Node performing Head of Bus B function in an open subnetwork

If the node is performing the Head of Bus B function in an open Dual Bus subnetwork, then Ph-SAP_A at that node is associated with an active duplex transmission link, and Ph-SAP_B is not. Receipt of a Ph-DATA indication with an octet of type SLOT_START or SLOT_DATA at Ph-SAP_A allows the octet to be accessed by the DQDB Layer, but does not lead to the generation of a Ph-DATA request at either Ph-SAP_B or Ph-SAP_A. Receipt of a Ph-DATA indication with an octet of type DQDB_MANAGEMENT at Ph-SAP_A allows the octet to be accessed by the DQDB Layer, and leads to the generation of a Ph-DATA request at Ph-SAP_B.

The Physical Layer generates EMPTY octets in Ph-DATA indication primitives at Ph-SAP_B, as defined in 4.2, *When Generated*, Case 2. The EMPTY octets are conveyed to the Head of Bus B function in the node and, after then being relayed through the DQDB Layer, lead to the generation of a Ph-DATA request at Ph-SAP_A. This process is depicted in figure 4-4.

The Ph-DATA request at Ph-SAP_A shall contain the octet received in the Ph-DATA indication at Ph-SAP_B, as modified by the DQDB Layer in accordance with the following rules:

a) The type of the octet shall not be modified.
b) For octets of type SLOT_START and SLOT_DATA, the value of each bit in the octet shall be modified according to the DQDB access mechanism.
c) For octets of type DQDB_MANAGEMENT, the value of each bit in the octet shall be modified according to the rules of the DQDB Layer Management Protocol. Note that the Head of Bus function shall set the TYPE bit alternately to (TYPE = 0) and (TYPE = 1) for successive DQDB_MANAGEMENT octets. (See 10.1.)

A Ph-DATA request with an octet of type DQDB_MANAGEMENT at Ph-SAP_B shall contain the octet of the same type received in the Ph-DATA indication at Ph-SAP_A, as modified according to the DQDB Layer Management Protocol rules.

4.3.2.3 Node performing Head of Bus A and Head of Bus B functions in a looped subnetwork

If the node is performing the Head of Bus function for both buses in a looped Dual Bus subnetwork, then both of the Ph-SAPs at that node are associated with an active duplex transmission link.

Receipt of a Ph-DATA indication at Ph-SAP_B allows the octet to be accessed by the DQDB Layer, but does not lead to the generation of a Ph-DATA request at either Ph-SAP_A or Ph-SAP_B. (The octet in the Ph-DATA indication received at Ph-SAP_B cannot take the value EMPTY.) Receipt of a Ph-DATA indication with an octet value not equal to EMPTY at Ph-SAP_A allows the octet to be accessed by the DQDB Layer, but does not lead to the generation of a Ph-DATA request at either Ph-SAP_A or Ph-SAP_B. This process is depicted in figure 4-5a) and describes operation at the end of Bus A and at the end of Bus B.

An octet received in a Ph-DATA indication at Ph-SAP_A with a value of EMPTY is conveyed to the Head of Bus A and Head of Bus B functions in the node. The octet is used to convey both octet timing and octet type information to the Head of Bus function. Receipt of a Ph-DATA indication at Ph-SAP_A with a value of EMPTY will result, after the octet is relayed through the DQDB Layer, in the generation of a Ph-DATA request at Ph-SAP_B. Receipt of a Ph-DATA indication at Ph-SAP_A with a value of EMPTY will also lead to the generation of a Ph-DATA request at Ph-SAP_A. This process is depicted in figure 4-5b).

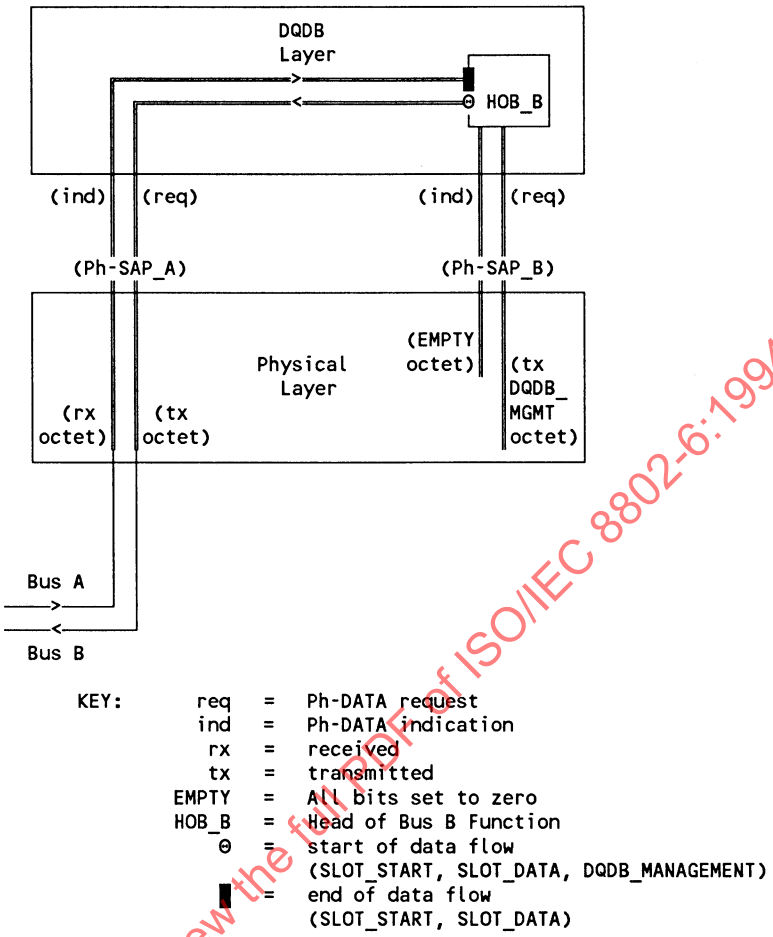


Figure 4-4—Relationship between Ph-DATA indication and Ph-DATA request: Node performing Head of Bus B function

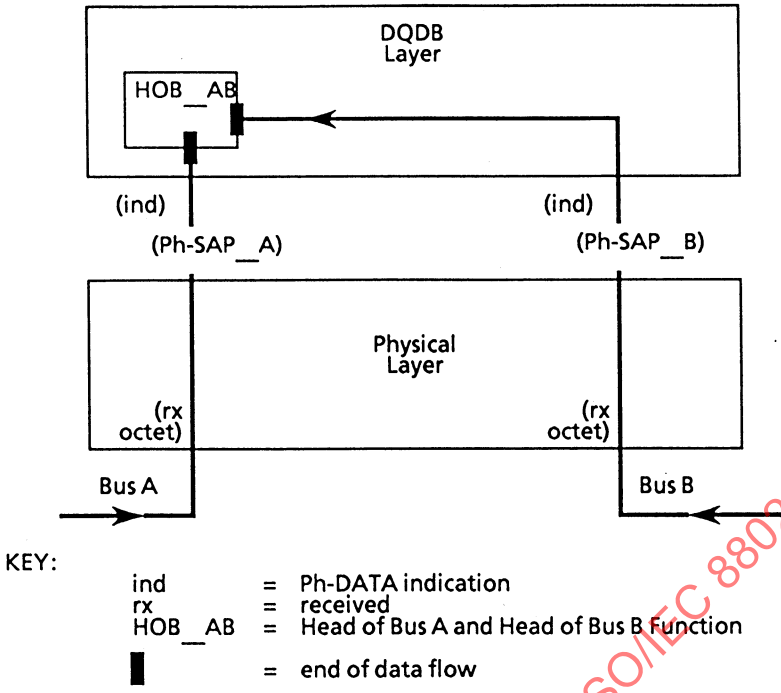
The Ph-DATA request at both Ph-SAP_A and Ph-SAP_B shall contain the EMPTY octet received in the Ph-DATA indication at Ph-SAP_A, as modified by the DQDB Layer in accordance with the following rules:

a) The type of the octet shall not be modified.

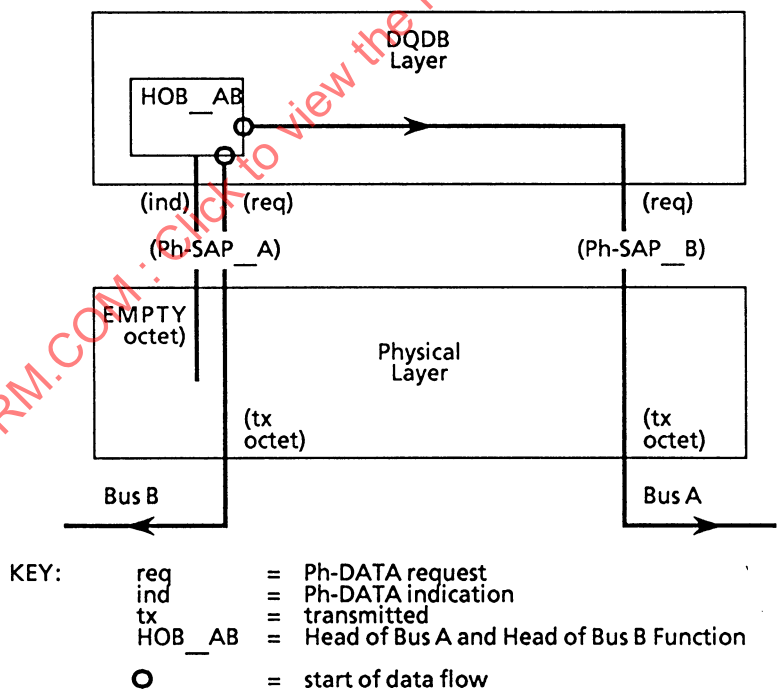
b) For octets of type SLOT_START and SLOT_DATA, the value of each bit in the octet shall be modified according to the DQDB access mechanism.

c) For octets of type DQDB_MANAGEMENT, the value of each bit in the octet shall be modified according to the rules of the DQDB Layer Management Protocol.[20] Note that the Head of Bus function shall ensure that the TYPE bit is set alternately to (TYPE = 0) and (TYPE = 1) for successive DQDB_MANAGEMENT octets. (See 10.1.)

[20] Note that there are DQDB Layer restrictions on the modifications allowed for some of the management information fields, according to the operation of the DQDB Layer Management Protocol. For example, the MID Page Allocation field, in management information octets of TYPE 1, should be relayed from the end of Bus A to the head of Bus B.



a) Node performing End of Bus A and End of Bus B functions



b) Node performing Head of Bus A and Head of Bus B functions

Figure 4-5—Relationship between Ph-DATA indication and Ph-DATA request

4.4 Ph-TIMING-SOURCE request

Function:

This primitive directs the Physical Layer protocol entity at the node to which source of 125 µs timing to use.

Semantics of the Service Primitive:

Ph-TIMING-SOURCE request (
source
)

The source parameter specifies the timing source to be used by the Physical Layer at the node. The values that can be associated with the source parameter are the following:

EXTERNAL_CLOCK, or
NODE_CLOCK, or
BUS_A, or
BUS_B, or
EITHER_BUS

When Generated:

This primitive is generated to indicate to the Physical Layer protocol entity at the node that it should change the source being used for 125 µs timing. The source may be changed by the Subnetwork Configuration Control procedures after a change in the subnetwork configuration. (See 10.2.4 to 10.2.6.) An EXTERNAL_CLOCK source directs the Physical Layer protocol entity to use the timing source traceable to an external timing source. A NODE_CLOCK source directs the Physical Layer protocol entity to use the local node clock. A BUS_A source directs the Physical Layer protocol entity to use the timing received at the node from Bus A. A BUS_B source directs the Physical Layer protocol entity to use the timing received at the node from Bus B. An EITHER_BUS source allows the Physical Layer protocol entity to use the timing derived at the node from either bus.

Effect on Receipt:

The Physical Layer protocol entity starts to use the specified timing source for 125 µs timing. The EITHER_BUS value of source parameter applies only if it has been received at both Ph-SAPs. If EITHER_BUS is received at one Ph-SAP and any of the other values of source is received at the other Ph-SAP, then the source value received at the other Ph-SAP applies. If the EITHER_BUS value is received at both Ph-SAPs, then the Physical Layer shall interpret this as a Ph-TIMING-Source request at Ph-SAP_A with a value of BUS_B and a Ph-TIMING-SOURCE request at Ph-SAP_B with a value of BUS_A. This will effectively relay the timing on the two buses.

Additional Comments:

This primitive is of the type shown in figure 1-3a), which shows that it has local effect. Note, however, that sufficient information is signaled to other nodes by the Configuration Control protocol to allow the other nodes to determine which timing source to use.

4.5 Ph-TIMING-MARK indication

Function:

This primitive indicates the arrival of 125 µs timing information from the primary timing source for the subnetwork, either within the node, or as relayed by the peer Physical Layer protocol entity.

Semantics of the Service Primitive:

Ph-TIMING-MARK indication (
)

When Generated:

The local Physical Layer protocol entity sends a Ph-TIMING-MARK indication whenever DQDB Layer timing information is received from exactly one of the following:

a) The primary timing source for the subnetwork, if it is contained within the node, or otherwise,
b) The peer Physical Layer protocol entity.

Effect on Receipt:

The effect of receipt of this primitive is dependent upon the operation of the DQDB Layer. It is only used if the DQDB Layer is supporting service to an Isochronous Service User (ISU).

Additional Comments:

The receipt of this primitive when the node contains the primary timing source for the subnetwork, as described in *When Generated*, Case 1, is illustrated in figure 1-3(b). The timing information is relayed from the primary timing source for the subnetwork via all intermediate Physical Layer entities acting as Physical Layer relays. Hence, the receipt of this primitive at other nodes, as described in *When Generated*, Case 2, is also of the type shown in figure 1-3b).

4.6 Ph-STATUS indication

Function:

This primitive is generated by the Physical Layer entity to indicate the operational state of the duplex transmission link associated with the Physical Layer Service Access Point (Ph-SAP).

Semantics of the Service Primitive:

Ph-STATUS indication (
status
)

The value of the status parameter is one of the following:

UP, or
DOWN

When Generated:

This primitive is generated to indicate a change in the status of the duplex transmission link associated with the Ph-SAP. A status of UP indicates that the Physical Layer considers that the duplex transmission link is active between the two nodes at each end of the link, and that there is an active Head of Bus function upstream on the bus that enters the node at this Ph-SAP. A status of DOWN indicates that the Physical Layer considers that either one or both directions of the duplex transmission link are not active, or that there is no active Head of Bus function upstream on the bus that enters the node at this Ph-SAP.

Effect on Receipt:

The value of the status parameter is used to set the Link Status Indicator associated with the Ph-SAP. (See 7.5.3.)

Additional Comments:

This primitive is of the type shown in figure 1-3d), which shows that the Ph-STATUS indication will be generated at both Ph-SAPs associated with the duplex transmission link, although there is no specified time relationship between the two primitives.

The conditions under which the Physical Layer considers the duplex transmission link to be either UP or DOWN are a function of the underlying transmission system and are defined as part of each Physical Layer Convergence Procedure (PLCP). The general requirements are outlined in 11.5.3 and 11.5.4.

5. DQDB node functional description

The DQDB Layer uses the services of the Physical Layer, defined in clause 4, to provide the services defined in clause 3. The functional architecture of a DQDB node is shown in figure 5-1 and is described informally in 2.2. This clause provides a rigorous description of the functional operation of the DQDB Layer subsystem of a node, with reference to the DQDB Layer Protocol Data Unit formats (clause 6), facilities (clause 7), protocol machine operation (clause 8), Layer Management Interface (clause 9), and Layer Management protocol (clause 10).

The operation of each functional element in the DQDB Layer, as shown in figure 5-1, is described in the clauses below. The functional element being described in each subclause is shown as the shaded box in a reduced-size copy of figure 5-1.

Subclause 5.1 describes the functions performed by the MAC Convergence Function (MCF) block and the Queued Arbitrated Functions block to support the MAC Sublayer service to the LLC Sublayer.

Subclause 5.2 gives guidelines for the definition of an Isochronous Convergence Function (ICF) block and describes the functions performed by the Pre-Arbitrated (PA) Functions block to support the isochronous service.

Subclause 5.3.1 gives guidelines for the functions that will be specified for the Connection-Oriented Convergence Function (COCF) which, together with the Queued Arbitrated (QA) Functions block described in 5.1.2, will provide the connection-oriented data service to be defined in the future.

Subclause 5.4 provides a functional description of the Common Functions block, which provides functions common to some or all of the other functional blocks, such as Configuration Control functions, Head of Bus functions, and MID Page Allocation functions.

5.1 Provision of MAC service to LLC

This clause describes the functions performed by the DQDB Layer to support the transfer of a LLC PDU from one DQDB node to one or more peer DQDB nodes.

The LLC PDU is received at the source node as the MAC Service Data Unit (MSDU) in an MA-UNITDATA request, as defined in ISO/IEC 10039. Figure 5-2 summarizes the transmit and receive functions that are required of the MCF block and the QA Functions block to transfer an MSDU between nodes. The MSDU is delivered by each destination node in an MA-UNITDATA indication, as defined in ISO/IEC 10039.

5.1.1 MAC Convergence Function (MCF) block

5.1.1.1 MCF transmit functions

This clause specifies the functions performed by the MCF block upon receipt of a single MSDU in an MA-UNITDATA request, as depicted in figure 5-3. Subclause 5.1.1.1.1 specifies the creation of an Initial MAC Protocol Data Unit (IMPDU) by the addition of protocol control information to the MSDU. Subclause 5.1.1.1.2 specifies the segmentation of the IMPDU into fixed-length segmentation units. Subclause 5.1.1.1.3 specifies how a Derived MAC Protocol Data Unit (DMPDU) is created by the addition of protocol control information to a segmentation unit. Subclause 5.1.1.1.4 specifies how the DMPDU is passed to the QA Functions block as a QA segment payload, along with data needed by the QA Functions block to generate the QA segment header, and other control information needed to transfer the DMPDU.

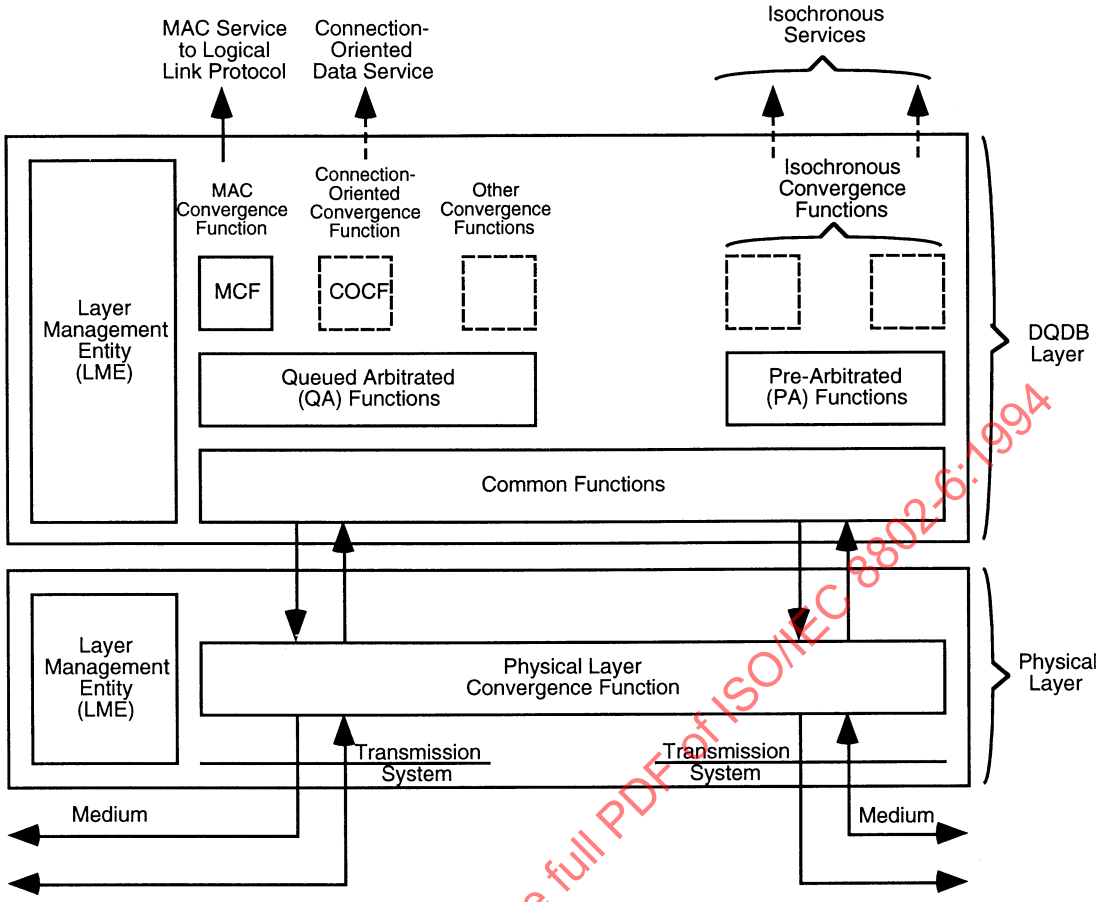


Figure 5-1—DQDB node functional architecture

5.1.1.1.1 Creation of the IMPDU

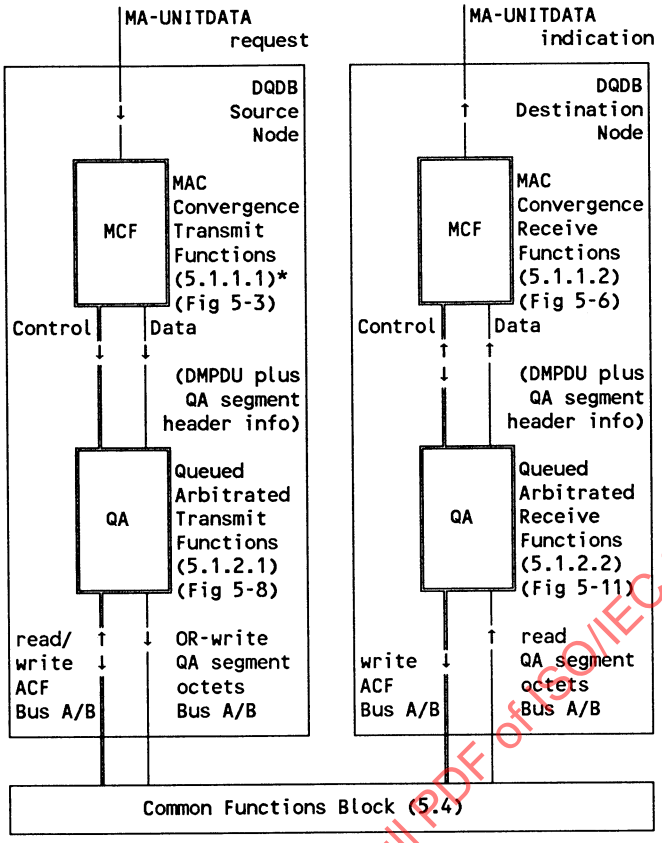
Upon receipt of an MA-UNITDATA request from the LLC Sublayer, the MCF block shall perform the following operations to create an IMPDU from the MAC Service Data Unit (MSDU) parameter contained in the MA-UNITDATA request. The format of the IMPDU is given in 6.5.1.

Steps 1)–3) are for the creation of the Common PDU header.

1) Code the Reserved field of the Common PDU header to zero, according to 6.5.1.1.1.
2) Select a BEtag value and code the BEtag field of the Common PDU header, according to 6.5.1.1.2. The BEtag field value shall be incremented by one (modulo 256) for sequential IMPDUs sent by the node.
3) Code the BAsize field of the Common PDU header to the number of octets contained in the MAC Convergence Protocol header, the Header Extension field, the MSDU, the PAD field, and the CRC32 field (if present), according to 6.5.1.1.3.

Steps 4)–12) are for the creation of the MAC Convergence Protocol (MCP) header.

4) Code the destination_address parameter received in the MA-UNITDATA request into the DA field of the MCP header, according to 6.5.1.2.2.



* Numbers in parentheses refer to clauses and subclauses of this part of ISO/IEC 8802 where the services are defined.

Figure 5-2—DQDB Layer functions to support LLC service

5) Code the source_address parameter received in the MA-UNITDATA request into the SA field of the MCP header, according to 6.5.1.2.3.
6) Code the PI field of the MCP header to decimal 1, according to 6.5.1.2.4.1.
7) Code the PAD Length (PL) field according to 6.5.1.2.4.2. Number_PAD_octets = 3 – [(Length of INFO field + 3)(mod 4)]
8) Code the priority parameter received in the MA-UNITDATA request into the QOS_DELAY subfield of the MCP header, according to 6.5.1.2.5.1.
9) Code the QOS_LOSS subfield of the MCP header to zero, according to 6.5.1.2.5.2.
10) If the CRC32_GEN_CONTROL Flag (see 7.4.3) is set to OFF, then the CIB (CRC32 Indicator bit) subfield of the MCP header is coded to zero. If the CRC32_GEN_CONTROL flag is set to ON, then the CIB subfield is coded to one, according to 6.5.1.2.5.3.
11) Compare the parameters received in the MA-UNITDATA request with the HE_selection_info values currently installed at the MCF block by DQDB Layer Management HEXT_INSTAL and HEXT_PURGE actions (9.3.1; 9.3.2). If no match is made then the HEL subfield of the MCP header is coded to zero. If a match is made then the HEL subfield is coded to the length of the corresponding HE_value, according to 6.5.1.2.5.4.

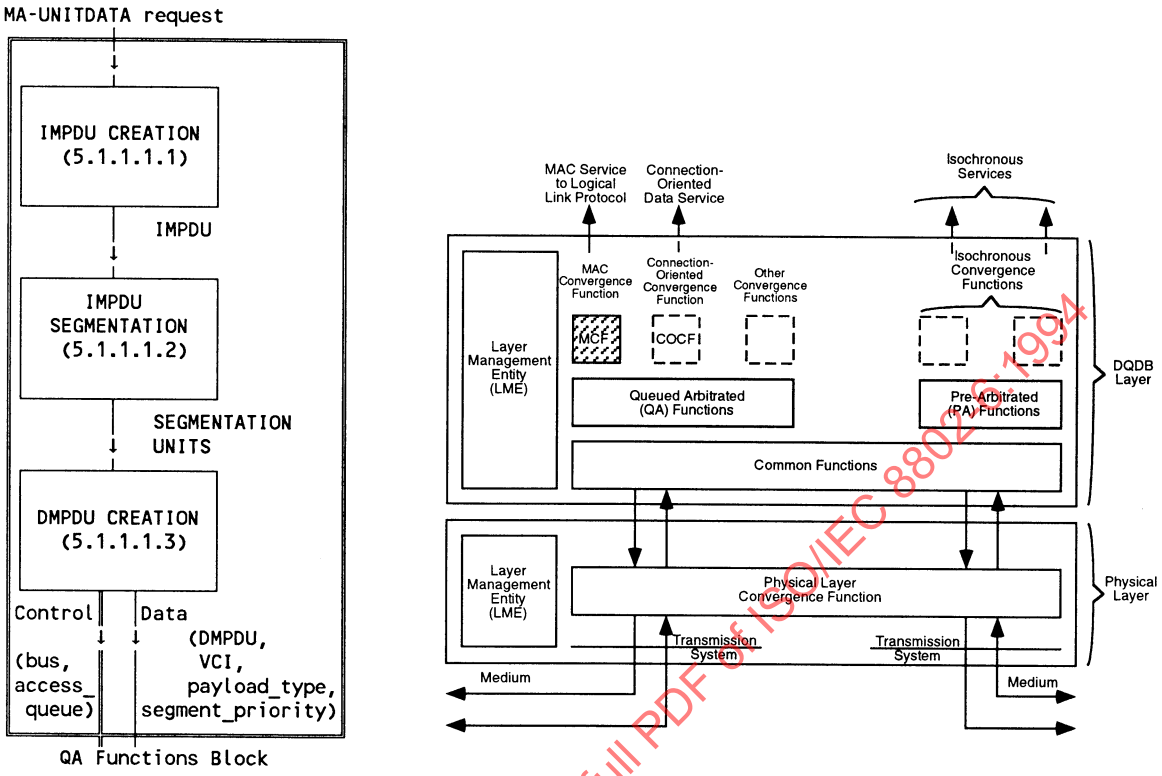


Figure 5-3—MCF transmit functions

12) Code the BRIDGING field of the MCP header to zero, according to 6.5.1.2.6.

Step 13) is for the creation of the Header Extension field.

13) If the HEL subfield generated in 11) above contains a value of zero, then there is no Header Extension field used for this IMPDU. If the HEL subfield generated in 11) above contains a nonzero value, then the Header Extension field is coded with the HE_value corresponding to the matched HE_selection_info value for this IMPDU, according to 6.5.1.3.

Step 14) is for the creation of the INFO field.

14) Code the data parameter (MSDU) received in the MA-UNITDATA request into the INFO field, according to 6.5.1.4.

Step 15) is for the creation of the PAD field.

15) Create the PAD field such that the length of INFO field plus PAD field is an integral multiple of four octets. The number of PAD octets is either 0, 1, 2, or 3, as indicated by the PAD Length (PL) field in 7) above. Each PAD octet is coded as zero, according to 6.5.1.5.

Step 16) is for the creation of the CRC32 field.

16) If the CIB subfield generated in j) above contains a value of zero, then there is no CRC32 field used for this IMPDU. If the CIB subfield generated in 10) above contains a value of one, then the CRC32 field is coded according to 6.5.1.6. (Note that the fields covered by the CRC32 field are those created in Steps 4)–15), inclusive; that is, the MCP header, the Header Extension field, the INFO field, and the PAD field.)

Steps 17)–19) are for the creation of the Common PDU trailer.

17) Code the Reserved field of the Common PDU trailer to zero, according to 6.5.1.7.1.
18) Code the BEtag field of the Common PDU trailer (see 6.5.1.7.2) with the same value as contained in the BEtag field in the Common PDU header, as per 2) above.
19) Code the Length field of the Common PDU trailer to the number of octets contained in the MAC Convergence Protocol header, the Header Extension field, the MSDU, the PAD field, and the CRC32 field (if present), according to 6.5.1.7.3. This shall be the same as the value in the BAsize field in the Common PDU header, as per 3) above.

5.1.1.1.2 Segmentation of the IMPDU

The MCF block takes an IMPDU created as described in 5.1.1.1.1 and divides it into one or more segmentation units. Each segmentation unit is 44 octets long and shall contain 44 octets of the IMPDU, with the exception of the last segmentation unit, which may contain less than 44 octets of the IMPDU. The number and type of segmentation units depends on the length of the IMPDU.

5.1.1.1.2.1 IMPDU of one segmentation unit

If the length of the IMPDU is less than or equal to 44 octets, then only one segmentation unit is generated, which is a Single Segment Message (SSM) segmentation unit. The SSM segmentation unit contains the entire IMPDU, according to 6.5.2.1.1.4.

5.1.1.1.2.2 IMPDU of two segmentation units

If the length of the IMPDU is greater than 44 octets, and less than or equal to 88 octets, then two segmentation units are generated. The order of octets in the IMPDU shall be preserved by the segmentation. The first segmentation unit is a Beginning of Message (BOM) segmentation unit, which contains the first 44 octets of the IMPDU, according to 6.5.2.1.1.1. The second segmentation unit is an End of Message (EOM) segmentation unit, which contains the remaining octets of the IMPDU, according to 6.5.2.1.1.3.

5.1.1.1.2.3 IMPDU of more than two segmentation units

If the length of the IMPDU is greater than 88 octets, then more than two segmentation units are generated. The order of octets in the IMPDU shall be preserved by the segmentation. The first segmentation unit is a Beginning of Message (BOM) segmentation unit, which contains the first 44 octets of the IMPDU, according to 6.5.2.1.1.1. All of the other segmentation units, except the last, are Continuation of Message (COM) segmentation units, which contain 44 octets of the IMPDU, according to 6.5.2.1.1.2. The last segmentation unit is an End of Message (EOM) segmentation unit, which contains the remaining octets of the IMPDU, according to 6.5.2.1.1.3. This process is depicted in figure 5-4.

5.1.1.1.3 Creation of the DMPDUs

The MCF block creates a DMPDU from each segmentation unit created, as described in 5.1.1.1.2. The type of DMPDU created depends upon the type of segmentation unit, and is described in 5.1.1.1.3.1 and

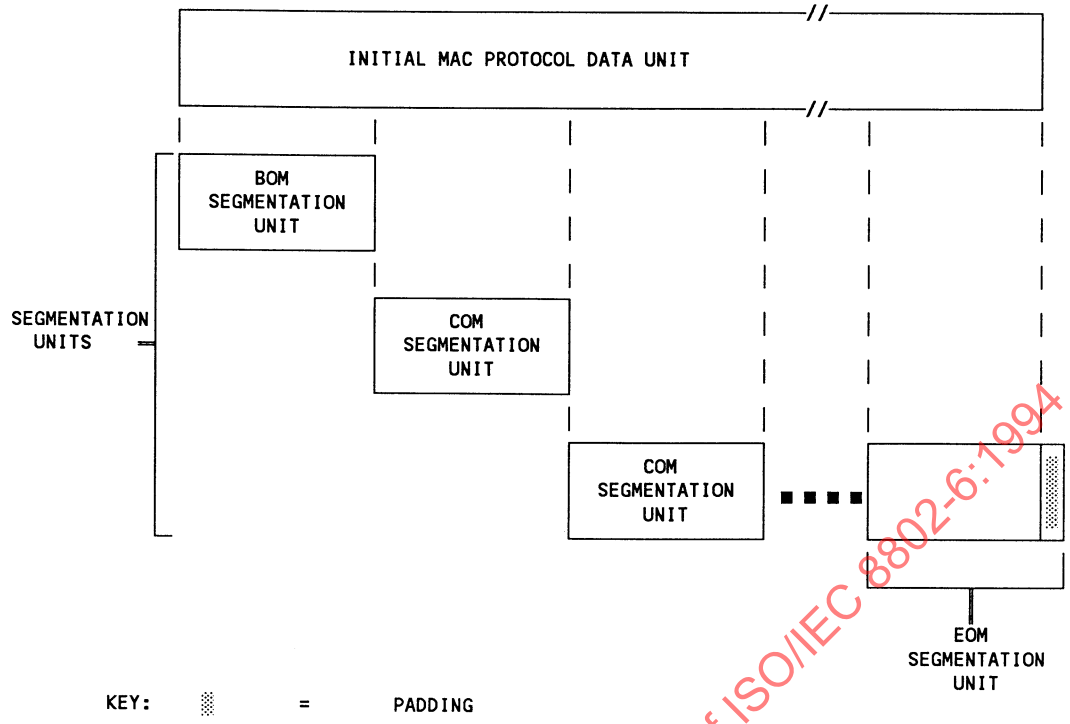


Figure 5-4—Segmentation of an IMPDU of greater than 88 octets in length

5.1.1.1.3.2. Each DMPDU is passed to the QA Functions block, as described in 5.1.1.1.4. The order of passing DMPDUs shall preserve the order of the octets in the original IMPDU.

5.1.1.1.3.1 DMPDU from an IMPDU of one segmentation unit

If segmentation of the IMPDU generates an SSM segmentation unit, as described in 5.1.1.1.2.1, the MCF block shall add the following protocol control information to the segmentation unit to create a DMPDU. The format of the DMPDU is given in 6.5.2.

a) Code the Segment_Type subfield of the SSM DMPDU header with the binary value 11 (SSM), according to 6.5.2.1.1.

b) Take the current value of the TX_SEQUENCE_NUM counter (see 7.2.6) associated with the MID value selected in c) and the VCI selected in 5.1.1.1.4, step c), and code the Sequence_Number subfield of the SSM DMPDU header with this value according to 6.5.2.1.2. The value of TX_SEQUENCE_NUM shall then be incremented by one (modulo 16) for use with the next SSM DMPDU to be sent by the node with the same VCI.

c) Code the MID subfield of the SSM DMPDU header with the reserved SSM value of all ten bits set to zero, according to 6.5.2.1.3.

d) Code the Payload_Length subfield of the DMPDU trailer with the number of octets of the IMPDU contained in the SSM segmentation unit, according to 6.5.2.2.1. The value of this subfield shall be any decimal number which is a multiple of 4 in the range 28 to 44, inclusive.
e) Calculate and code the Payload_CRC subfield of the DMPDU trailer, according to 6.5.2.2.2.

The SSM DMPDU is then passed to the QA Functions block as described in 5.1.1.1.4.

5.1.1.1.3.2 DMPDUs from an IMPDU of more than one segmentation unit

If segmentation of the IMPDU generates more than one segmentation unit, as described in 5.1.1.1.2.2 and 5.1.1.1.2.3, the MCF block shall add the following protocol control information to each segmentation unit to create a DMPDU. The format of the DMPDU is given in 6.5.2.

a) Code the Segment_Type subfield of the DMPDU header with the appropriate binary value of either 10 for Beginning of Message (BOM), 00 for Continuation of Message (COM), or 01 for End of Message (EOM), according to 6.5.2.1.1.
b) Take the current value of the TX_SEQUENCE_NUM counter (see 7.2.6) associated with the MID value selected in c) and the VCI selected in 5.1.1.1.4, step c), and code the Sequence_Number subfield of the DMPDU header with this value according to 6.5.2.1.2. The value of TX_SEQUENCE_NUM shall then be incremented by one (modulo 16) for use with the next DMPDU to be sent by the node with the same MID value, and using the same VCI.
c) Select an MID value and code the MID subfield of the DMPDU header, according to 6.5.2.1.3, and subject to the following conditions:
    1) The MID value shall be the same for every DMPDU derived from the IMPDU.
    2) The selection mechanism for the MID value should ensure that when the DMPDU is sent by the source node, it can be unambiguously associated on the subnetwork with the IMPDU from which it is derived. The MID subfield value is selected from one of the MID Page values currently available to the node via the MID Page Allocation functions described in 5.4.4.2.
d) Code the Payload_Length subfield of the DMPDU trailer with the number of octets of the IMPDU contained in the segmentation unit, according to 6.5.2.2.1. For the BOM DMPDU and any COM DMPDUs the value of this subfield shall be decimal 44. For the EOM DMPDU the value of this subfield shall be any number which is a multiple of 4 in the range 4 to 44, inclusive.
    3) Calculate and code the Payload_CRC subfield of the DMPDU trailer, according to 6.5.2.2.2.

The DMPDU is then passed to the QA Functions block as described in 5.1.1.1.4.

5.1.1.1.4 Transmit interactions between MCF block and QA Functions block

The interaction between the MCF block and the QA Functions block for the transfer of a DMPDU as the payload of a QA segment is dependent on the parameters received in the MA-UNITDATA request, but is independent of the type of the DMPDU, and is depicted in figure 5-5.

The MCF block shall perform the following operations to establish the interaction between itself and the QA Functions block for the transfer of each DMPDU derived from a given IMPDU.

a) The TX_BUS_SIGNAL control shall be asserted by the MCF block to the same value for every DMPDU derived from the IMPDU. The TX_BUS_SIGNAL control indicates on which bus or buses each DMPDU of the IMPDU should be sent. The values that can be associated with TX_BUS_SIGNAL are as follows:

    BUS_A, or
    BUS_B, or
    BOTH

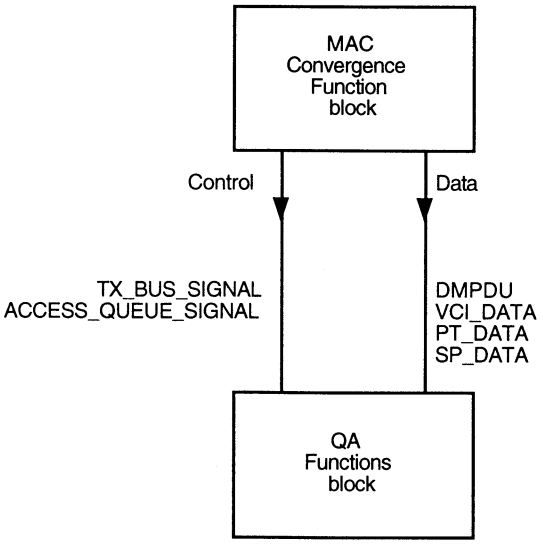


Figure 5-5—Transmit interaction between MCF block and QA Functions block

The MCF block determines the appropriate value for TX_BUS_SIGNAL by examination of the destination_address parameter from the MA-UNITDATA request that generated the IMPDU. The MCF block may know which bus or buses should be used for sending each DMPDU of the IMPDU, using either one of the suggested procedures outlined in annex C or any other suitable procedure. Then,

1) If the MCF block knows which bus or buses to use, then TX_BUS_SIGNAL is asserted to the appropriate value for the IMPDU.

2) If the MCF block does not know which bus to use, then TX_BUS_SIGNAL shall be asserted to BOTH for the IMPDU.

b) The ACCESS_QUEUE_SIGNAL control shall be asserted by the MCF block to the same value for every DMPDU derived from the IMPDU and indicates in which priority level access queue the QA segment used to transfer the DMPDU should be placed. The values that can be associated with ACCESS_QUEUE_SIGNAL are 0, 1, or 2.

The MCF block determines the appropriate value for ACCESS_QUEUE_SIGNAL by examination of the priority parameter from the MA-UNITDATA request that generated the IMPDU. The value of the requested priority parameter, J, is used to determine the access queue priority level, I, to be used for the IMPDU by reading the value of QOS_MAP_J, from the QOS_MAP system parameter defined in 7.3.3. ACCESS_QUEUE_SIGNAL is asserted to the value of QOS_MAP_J for the IMPDU.

c) The VCI_DATA shall be the same for every DMPDU derived from the IMPDU and contains the value of VCI to be placed in the header of each QA segment used to transfer one of the DMPDUs. All nodes conforming to this part of ISO/IEC 8802 shall be able to transmit and receive DMPDUs at the default connectionless VCI. (See 6.3.1.1.1.1.) If an implementation is capable of recognizing connectionless VCIs additional to the default value, then the VCIs shall be constrained such that the range is expressed by the right-most bits, with all the left-most bits set to one. Such a node may optionally be programmed to transmit and receive DMPDUs for the MCF block at other connectionless VCIs by an LM-ACTION invoke (CL_VCI_ADD). (See 9.2.1.)

The MCF block determines the appropriate value for VCI_DATA by examination of the parameters received in the MA-UNITDATA request which generated the IMPDU. The parameters are compared with the mapping_criteria value currently installed for each transmit VCI available for use by the MCF block, as controlled by DQDB Layer Management CL_VCI_ADD and CL_VCI_DELETE actions. (See 9.2.1 and 9.2.2.) Then,

1) If a match is made for the mapping_criteria, then VCI_DATA is set to the corresponding transmit VCI value for the IMPDU.
2) If no match is made for the mapping_criteria, then VCI_DATA is set to the default connectionless VCI for the IMPDU.

d) The PT_DATA shall be the same for every DMPDU derived from the IMPDU and contains the value of Payload_Type to be placed in the header of each QA segment used to transfer one of the DMPDUs. PT_DATA is set to binary 00 for all DMPDU transfers at the default connectionless VCI. The setting of PT_DATA for any other value of VCI_DATA is under study. The default setting of PT_DATA for other values of VCI_DATA is binary 00.

e) The SP_DATA shall be the same for every DMPDU derived from the IMPDU and contains the value of Segment_Priority to be placed in the header of each QA segment used to transfer one of the DMPDUs. SP_DATA is set to binary 00 for all DMPDU transfers at the default connectionless VCI. The setting of SP_DATA for any other value of VCI_DATA is under study. The default setting of SP_DATA for other values of VCI_DATA is binary 00.

5.1.1.2 MCF Receive functions

The process of reassembly is used to reconstruct an IMPDU from the segmentation units contained in DMPDUs received by the MCF block. Any IMPDU that has more than one DMPDU derived from it, and is currently being received and reassembled by the node, has an instance of the Reassembly State Machine (RSM) (see 8.2.1) associated with it to control the reassembly process.

As the DMPDUs from different IMPDUs destined to the node may arrive in an interspersed manner, the MCF block includes the functions required to support the simultaneous operation of multiple reassembly processes, each controlled by an active instance of the RSM. For descriptive purposes in this part of ISO/IEC 8802, the abstract model of reassembly assumes that there is an instance of the RSM for all of the MIDs of every VCI that the MCF block is programmed to receive. Therefore, each RSM is uniquely associated with a VCI/MID pair. However, the number of reassembly processes that can be supported simultaneously in a given node is an implementation choice.

This clause specifies the functions performed by the MCF block upon receipt of a stream of DMPDUs from the QA Functions block, as depicted in figure 5-6. Subclause 5.1.1.2.1 specifies how a DMPDU is passed from the QA Functions block along with data extracted from the header of the QA segment that carried the DMPDU as payload, and other control information needed to process the DMPDU. Subclause 5.1.1.2.2 specifies the operation of the MCF block to direct the DMPDU to the appropriate reassembly state machine. Subclause 5.1.1.2.3 specifies the function of reassembly of an IMPDU from received DMPDUs. Subclause 5.1.1.2.4 specifies how a reassembled IMPDU is validated. Subclause 5.1.1.2.5 specifies how the MSDU and appropriate parameters are extracted from the IMPDU and passed by the MCF block in an MA-UNITDATA indication.

5.1.1.2.1 Receive interactions between QA Functions block and MCF block

The interaction between the QA Functions block and the MCF block upon the receipt at the node of a DMPDU as the payload of a QA segment is depicted in figure 5-7.

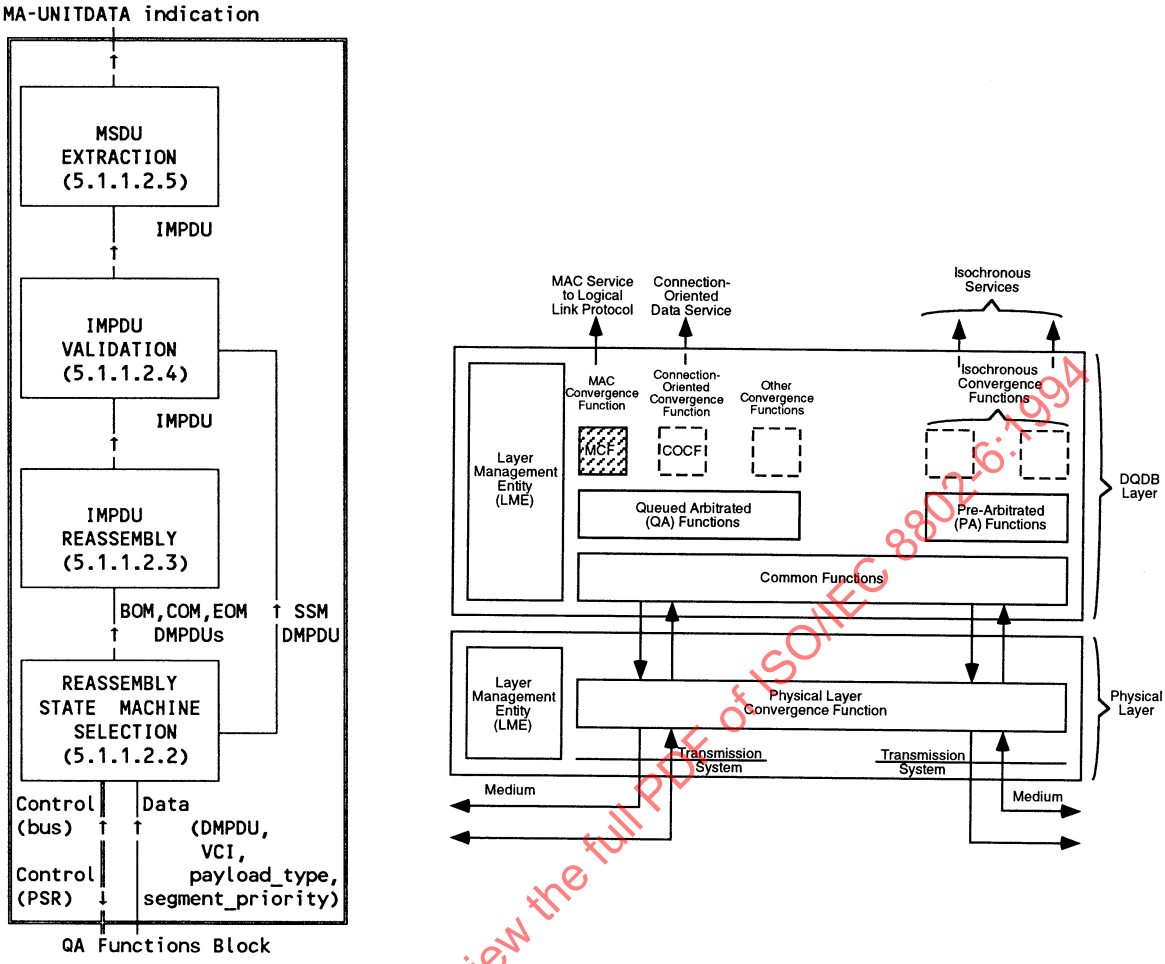


Figure 5-6—MCF Receive functions

a) The RX_BUS_SIGNAL control indicates on which bus the DMPDU was received. The values that can be associated with RX_BUS_SIGNAL are as follows:

BUS_A, or
BUS_B

b) The PSR_x_SIGNAL (x = A or B) is asserted by the MCF block to indicate that the DMPDU received in a slot on the bus indicated by RX_BUS_SIGNAL was destined only to this node. If the PSR_x_SIGNAL is to be asserted upon receipt of a DMPDU, then it shall be asserted before the Access Control Field (ACF) of the next slot arrives on Bus x at the node. If PSR_x_SIGNAL is asserted, then the QA Functions block shall set the Previous Segment Received (PSR) bit to one in the ACF of the next slot passing on Bus x, irrespective of the SL_TYPE bit in the ACF of the slot.

c) The VCI_DATA contains the value of VCI received in the header of the QA segment that carried the DMPDU. All nodes conforming to this part of ISO/IEC 8802 shall be able to transmit and receive DMPDUs at the default connectionless VCI. (See 6.3.1.1.1.1.) If an implementation is capable of recognizing connectionless VCIs additional to the default value, then the VCIs shall be constrained

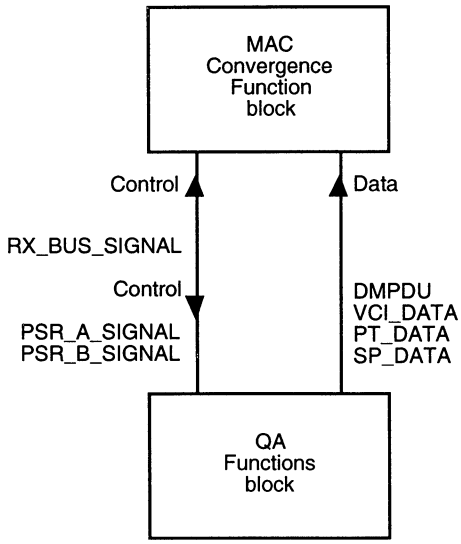


Figure 5-7—Receive interaction between QA Functions block and MCF block

such that the range is expressed by the right-most bits, with all the left-most bits set to one. Such a node may optionally be programmed to transmit and receive DMPDUs for the MCF block at other connectionless VCIs by an LM-ACTION invoke (CL_VCI_ADD). (See 9.2.1.)

d) The PT_DATA contains the value of Payload_Type received in the header of the QA segment that carried the DMPDU.

e) The SP_DATA contains the value of Segment_Priority received in the header of the QA segment that carried the DMPDU.

5.1.1.2.2 Reassembly state machine selection

When the MCF block receives a DMPDU from the QA Functions block at a connectionless VCI which the MCF block is programmed to receive, then the MCF block validates the correctness of the DMPDU using the Payload_CRC subfield in the DMPDU trailer. (See 6.5.2.2.2.) Then,

a) If the DMPDU is valid, then the MCF block shall perform one of the following operations, depending upon the value in the Segment_Type subfield of the DMPDU header (see 6.5.2.1.1):

1) *Segment_Type = BOM, COM, or EOM.* If the Segment_Type of the DMPDU is either BOM, COM, or EOM, then the MCF block forwards the DMPDU to the reassembly state machine associated with the VCI/MID pair corresponding to the VCI_DATA value and the MID contained in the DMPDU header (see 6.5.2.1.3). The DMPDU shall then be processed as described functionally in 5.1.1.2.3 and formally in 8.2.1.

If the DMPDU is destined only to this node, as determined by either condition i) or condition ii) below, the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block.

i) If the DMPDU is a BOM DMPDU, and the Destination Address (DA) field of the MCP header (see 6.5.1.2.2) matches an individual MSAP address which the node is programmed to receive, then PSR_x_SIGNAL shall be asserted.

ii) If the DMPDU is a COM or EOM DMPDU, and the value of the MID subfield of the DMPDU header (see 6.5.2.1.3) matches that associated with an active reassembly process for an individual MSAP address for that MID/VCI_DATA pair, then PSR_x_SIGNAL shall be asserted.

If neither condition i) nor condition ii) above holds for a valid DMPDU, then the MCF block shall not assert the PSR_x_SIGNAL.

2) *Segment_Type = SSM.* If the Segment_Type of the DMPDU is SSM, then the DMPDU contains a complete IMPDU and does not need to be forwarded to a reassembly process.

The MCF block shall examine the value in the MID subfield of the DMPDU header. (See 6.5.2.1.3.) Then,

i) If the MID value does not match the reserved SSM value of all ten bits being set to zero, the SSM DMPDU shall be discarded. The MCF block shall also assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block.

ii) If the MID value does match the reserved SSM value, then the MCF block shall examine the value in the Destination Address (DA) field of the MCP header (see 6.5.1.2.2) to determine if it matches an MSAP address which the node is programmed to receive. Then,

(a) If no DA match is made, then the SSM DMPDU shall be discarded and no further action shall be taken. In particular, the MCF block shall not assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block.

(b) If a DA match is made, then the MCF block shall extract the IMPDU from the first N octets (where N is the value of the Payload_Length subfield in the DMPDU trailer (see 6.5.2.2.1) of the SSM segmentation unit, and pass the IMPDU to the validation process specified in 5.1.1.2.4.

— If the value of the DA field is an individual address, the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block.

— If the value of the DA field is not an individual address, then the MCF block shall not assert the PSR_x_SIGNAL.

b) If the DMPDU is not valid, the DMPDU shall be discarded.

If any of the Conditions A, B, or C below holds, the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block.

If none of the Conditions A, B, or C below holds, then the MCF block shall not assert the PSR_x_SIGNAL.

Condition A. The Segment_Type of the DMPDU is BOM or SSM, and the Destination Address field of the MCP header matches an individual MSAP address which the node is programmed to receive.

Condition B. The Segment_Type of the DMPDU is SSM, the node does not support single bit error correction of the DMPDU header via the Payload_CRC subfield in the DMPDU trailer, and the MID value does not match the reserved SSM value of all ten bits being set to zero.

Condition C. The Segment_Type of the DMPDU is COM or EOM, the node does not support single bit error correction of the DMPDU header via the Payload_CRC subfield in the DMPDU trailer, and the MID value matches that associated with an active reassembly process for an individual MSAP address for the MID/VCI_DATA pair.

5.1.1.2.3 Reassembly of the IMPDU

This clause specifies the process of reassembly of an IMPDU from received DMPDUs, assuming that no errors occur in the transfer of the DMPDUs. The complete description of the instance of the RSM, which controls this process, is given in 8.2.1 and takes precedence over this clause.

A reassembly process is activated when a BOM DMPDU is received from the RSM Selection function described in 5.1.1.2.2, and the Destination Address (DA) field of the MCP header (see 6.5.1.2.2) contains a value that matches an MSAP address that the node is programmed to receive. The reassembly process is associated with the value of VCI_DATA for the BOM DMPDU and the MID value in the BOM DMPDU header. (See 6.5.2.1.3.) The reassembly process extracts and stores the BOM segmentation unit.

If the segmentation of the IMPDU led to more than two segmentation units (as described in 5.1.1.1.2.3), then the MCF block should receive a series of COM DMPDUs with the same value of VCI_DATA and the same MID value in each COM DMPDU header as was in the BOM DMPDU header. The value of Sequence Number received for each COM DMPDU should be larger than the Sequence Number of the previous COM DMPDU (or BOM DMPDU in the case of the first COM DMPDU) by one (modulo 16). The reassembly process extracts the segmentation unit from each such COM DMPDU and appends it to the previously received BOM segmentation unit and COM segmentation unit(s).

If the segmentation of the IMPDU led to more than one segmentation unit (as described in 5.1.1.1.2.2 and 5.1.1.1.2.3), then the MCF block should receive an EOM DMPDU with the same value of VCI_DATA and the same MID value in the EOM DMPDU header as was in the BOM DMPDU header. The value of Sequence Number received for the EOM DMPDU should be larger than the Sequence Number of the previous COM DMPDU (or BOM DMPDU in the case of a two segmentation unit IMPDU) by one (modulo 16). The reassembly process extracts the number of octets from the EOM segmentation unit that contain IMPDU data, as indicated by the value of the Payload_Length field in the DMPDU trailer (see 6.5.2.2.1), and appends them to the previously received BOM segmentation unit and COM segmentation unit(s). This completes the reassembly process.

The reassembled IMPDU is passed to the IMPDU Validation process, described in 5.1.1.2.4. The reassembly process is then stopped and disassociated from the VCI_DATA value and MID value.

5.1.1.2.4 Validation of the IMPDU

Each completely received IMPDU, either fully contained in an SSM DMPDU (described in 5.1.1.2.2) or from a completed reassembly process (described in 5.1.1.2.3) is validated by performing the four operations described below:

a) The value in the Length field of the Common PDU trailer (see 6.5.1.7.3) is compared with the number of octets received for the IMPDU. The (number of received IMPDU octets minus 8)[21] should equal the value of Length field. A mismatch shall cause the MCF block to discard the IMPDU.

[21] That is, number of received octets less the size of the Common PDU header (4 octets) and less the size of the Common PDU trailer (4 octets).

b) The value in the BEtag field of the Common PDU header (see 6.5.1.1.2) is compared with the value in the BEtag field of the Common PDU trailer (see 6.5.1.7.2). A mismatch shall cause the MCF block to discard the IMPDU.

c) If the CRC32 Indicator Bit (see 6.5.1.2.5.3) is set to one in this IMPDU, and if the CRC32_CHECK_CONTROL Flag (see 7.4.4) is set to ON, then the CRC32 field (see 6.5.1.6) is used to validate the IMPDU. The fields included in the calculation of the 32-bit CRC are all those in the MCP header (see 6.5.1.2), the Header Extension field (see 6.5.1.3), the INFO field (see 6.5.1.4), and the PAD field (see 6.5.1.5). These fields are referred to as the calculation fields. However, for the purposes of calculation of the 32-bit CRC at the receiver, the contents of the BRIDGING field of the MCP header (see 6.5.1.2.6) shall be masked to have a value of all zeros,[22] irrespective of the contents of this field when received.

   If the CRC32 field is present and checked, a mismatch between the value received in the CRC32 field and the value calculated at the receiver shall cause the MCF block to discard the IMPDU.

d) The value in the Header Extension Length field of the MCP header (see 6.5.1.2.5.4) is checked to see if it is valid. A value outside the valid range of 0 to 5, inclusive, shall cause the MCF block to discard the IMPDU.

If all four operations are successful, then the IMPDU is passed to the MSDU Extraction function, described in 5.1.1.2.5.

The IMPDU Validation function shall receive IMPDUs in the same order in which the final DMPDU of each IMPDU (EOM DMPDU in the case of a multiple segmentation unit IMPDU, or SSM DMPDU in the case of a single segmentation unit IMPDU) is received by the MCF block. The IMPDU Validation function shall forward IMPDUs to the MSDU Extraction function in the same order as it receives them.

5.1.1.2.5 Extraction of the MSDU

The MSDU Extraction function takes a validated IMPDU and performs the following functions to create the parameters passed in an MA-UNITDATA indication:

a) Use the value in the DA field of the MCP header to create the destination_address parameter.
b) Use the value in the SA field of the MCP header to create the source_address parameter.
c) Use the value in the QOS_DELAY subfield of the MCP header to create the priority parameter.
d) Extract the MSDU contained in the INFO field to create the data parameter.

The MSDU Extraction function then generates an MA-UNITDATA indication. The MSDU Extraction function shall generate MA-UNITDATA indication primitives in the same order as it receives the IMPDUs from the IMPDU Validation function.

5.1.2 Queued Arbitrated (QA) Functions block

The QA Functions block controls the transfer in QA segments of QA segment payloads generated by Convergence Function blocks, such as the Derived MAC Protocol Data Units (DMPDUs) generated by the MAC Convergence Function (MCF) block.

The relationship between the QA Functions block and the MCF block for the default connectionless VCI is defined in this part of ISO/IEC 8802, and requires that all nodes conforming to this part of ISO/IEC 8802

[22] For the purposes of calculating the 32-bit CRC, it must be assumed that the BRIDGING field has been changed in transit from the value of all zeros generated at the transmitter. However, in the absence of transmission errors in all of the remaining fields covered by the CRC32 field, the assumption of a zero value BRIDGING field will lead to a matching value of 32-bit CRC being calculated. It is also assumed that all fields in the calculation fields, apart from the BRIDGING field, will not be changed in transit.

shall be able to transmit and receive DMPDUs at the default connectionless VCI. (See 6.3.1.1.1.1.) This support shall include transmission and reception of DMPDUs on both buses, with the VCI, Payload_Type and Segment_Priority Fields of the QA segment header set to the default values defined in 6.3.1.1.

The relationship between the QA Functions block and a connectionless Convergence Function block, including the MCF, for other VCI values is established by the optional DQDB Layer Management CL_VCI_ADD and CL_VCI_DELETE actions. (See 9.2.1 and 9.2.2.) These actions establish the conditions under which a particular connectionless VCI may be used.

The relationship between the QA Functions block and a Connection-Oriented Convergence Function (COCF) block is established by the optional DQDB Layer Management OPEN_CE_COCF and CLOSE_CE actions. (See 9.2.4 and 9.2.5.) These actions establish the bus and VCI to be used to transmit and receive QA segment payloads for a COCF block. The actions also establish the Segment_Priority and access queue priority level to be used to transmit QA segment payloads.

5.1.2.1 QA transmit functions

This clause specifies the functions performed by the QA Functions block upon receipt of a single QA segment payload from a Convergence Function block,[23] as depicted in figure 5-8. Subclause 5.1.2.1.1 specifies how the QA segment payload is passed by the Convergence Function block to the QA Functions block, along with data needed by the QA Functions block to generate the QA segment header, and other control information needed to transfer the QA segment payload. Subclause 5.1.2.1.2 specifies the creation of the QA segment by the addition of protocol control information to the QA segment payload. Subclause 5.1.2.1.3 specifies the function of FIFO queueing of QA segments that cannot yet be placed in the Distributed Queue. Subclause 5.1.2.1.4 specifies the functional operation of the Distributed Queue. Subclause 5.1.2.1.5 specifies how a QA segment octet is passed to the Common Functions block.

5.1.2.1.1 Transmit interactions between Convergence Function block and QA Functions block

The interaction between a Convergence Function block and the QA Functions block for the transfer of a QA segment payload is depicted in figure 5-9.

NOTE—The specific case of the Convergence function being the MAC Convergence Function is described in 5.1.1.1.4.

a) The TX_BUS_SIGNAL control indicates on which bus or buses the QA segment payload should be sent. The values that can be associated with TX_BUS_SIGNAL are as follows:

   BUS_A, or
   BUS_B, or
   BOTH

b) The ACCESS_QUEUE_SIGNAL control indicates in which priority level access queue the QA segment used to transfer the QA segment payload should be placed. The values that can be associated with ACCESS_QUEUE_SIGNAL are 0, 1, or 2.
c) The VCI_DATA contains the value of VCI to be placed in the header of the QA segment used to transfer the QA segment payload.
d) The PT_DATA contains the value of Payload_Type to be placed in the header of the QA segment used to transfer the QA segment payload.
e) The SP_DATA contains the value of Segment_Priority to be placed in the header of the QA segment used to transfer the QA segment payload.

[23] In the case of the MAC Convergence Function block, the QA segment payload is a DMPDU.

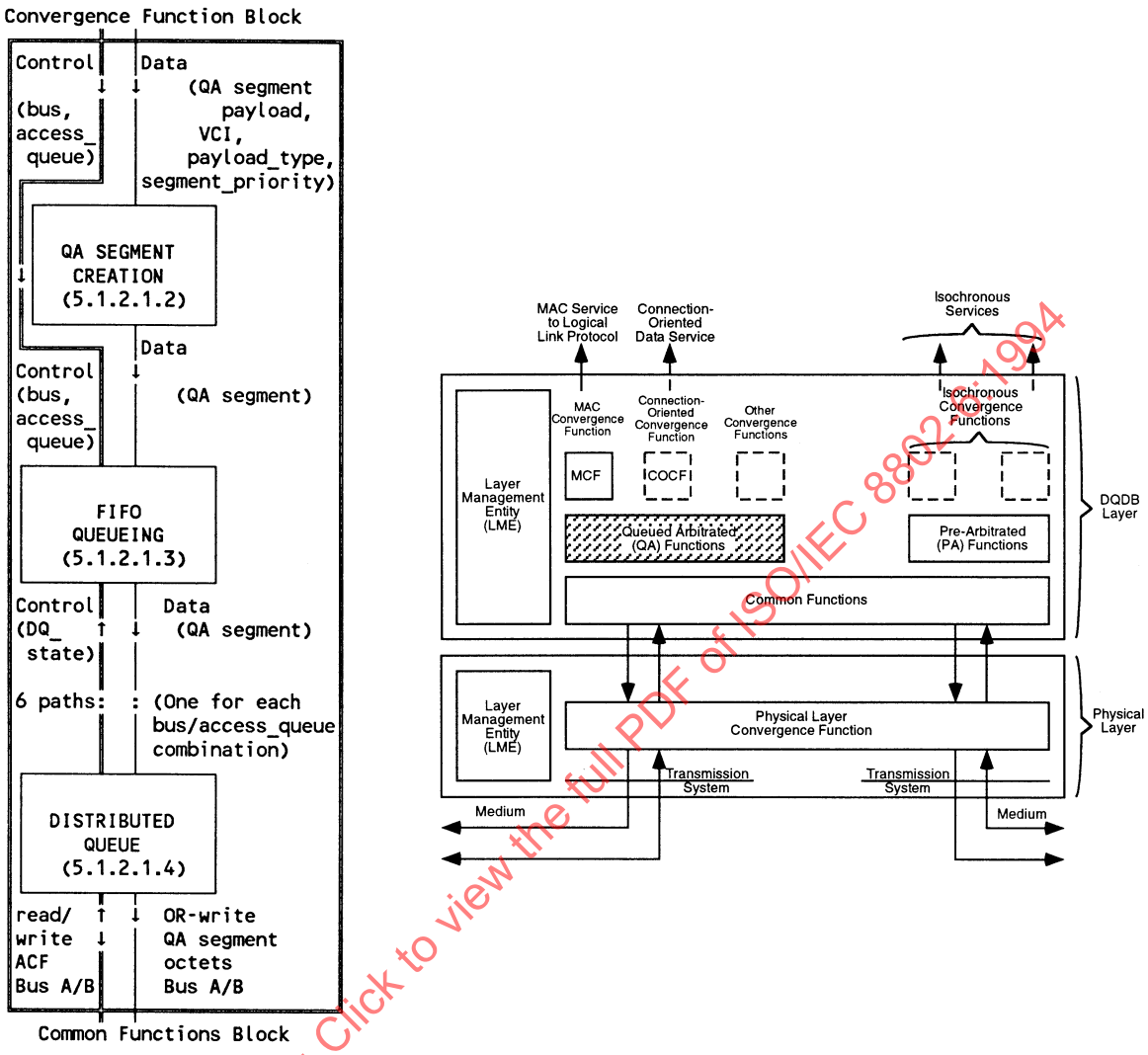


Figure 5-8—QA Functions block Transmit functions

5.1.2.1.2 QA segment creation

Upon receipt of a QA segment payload from a Convergence Function block, the QA Functions block shall add the following protocol control information to the QA segment payload to create a QA segment. The format of the QA segment is given in 6.3.1.

a) Code the VCI_DATA value into the VCI field of the QA segment header, according to 6.3.1.1.1.
b) Code the PT_DATA value into the Payload_Type field of the QA segment header, according to 6.3.1.1.2.
c) Code the SP_DATA value into the Segment_Priority field of the QA segment header, according to 6.3.1.1.3.

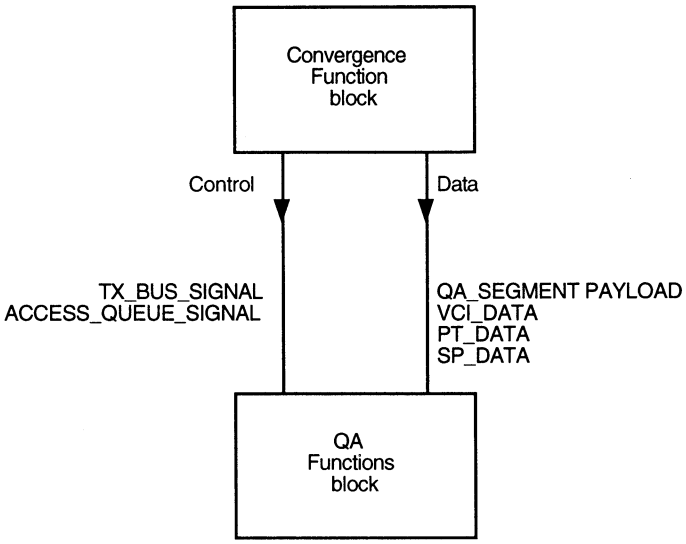


Figure 5-9—Transmit interaction between Convergence Function block and QA Functions block

d) Calculate and code the Segment Header Check Sequence (HCS) field of the QA segment header, according to 6.3.1.1.4.

The QA segment is then ready to be placed in the Distributed Queue indicated by the values of TX_BUS_SIGNAL and ACCESS_QUEUE_SIGNAL asserted when the QA segment payload was received by the QA Functions block. This process is described in 5.1.2.1.3 and 5.1.2.1.4.

5.1.2.1.3 FIFO queueing of QA segments

There can be a maximum of six QA segments queued within the Distributed Queue for access at each node, one for each combination of TX_BUS_SIGNAL and ACCESS_QUEUE_ SIGNAL that can be asserted to the QA Functions block. However, the QA Functions block can accept multiple QA segment payloads waiting for access to any given bus at any given access queue priority level. This is done by maintaining six local first-in-first-out (FIFO) queues of QA segments for each combination of bus and access queue priority level.

Each segment created as described in 5.1.2.1.2 is placed in the FIFO queue associated with the value of TX_BUS_SIGNAL and ACCESS_QUEUE_ SIGNAL asserted when the QA segment payload is received by the QA Functions block. If TX_BUS_SIGNAL is asserted to BOTH, then the QA segment is placed in both queues for the ACCESS_QUEUE_SIGNAL value.

The QA segment at the head of a FIFO queue is placed in the Distributed Queue for that combination of bus and access queue priority level when the DQ_state control signal shown in figure 5-8 indicates that the Distributed Queue State Machine (DQSM) for that particular combination is in the Idle state. (See 5.1.2.1.4.2 and 8.1.1.)

NOTE—The FIFO queueing function does not guarantee relative ordering of segments in queues for different buses or for different access queue priority levels.

5.1.2.1.4 Distributed queueing of QA segments

The Distributed Queue performs the medium access control procedure for the write access of QA segments into empty QA slots. The Distributed Queue operates request counters, countdown counters, and local request queue counters for each combination of bus and access queue priority level. The Distributed Queue also operates a bandwidth balancing counter for each bus.

Operation of the request and countdown counters for a Bus x (x = A or B) involves read operations on the BUSY bit and SL_TYPE bit in the Access Control Field (ACF) (see 6.2.1) for all slots on Bus x and read operations on the REQUEST field in the ACF of all slots on the opposite bus, Bus y (y = B or A, respectively). For each combination of Bus x and access queue priority level I (I = 0,1,2) there is an instance of the DQSM, which controls the request and countdown counters for that Bus x and access queue priority level I, denoted REQ_I_CNTR_x and CD_I_CNTR_x, respectively.

A request for access to Bus x at priority level I is signaled to other nodes by a write operation performed on the REQ_I bit in the ACF of each slot passing on the opposite bus, Bus y. The write operation stops after it sets to one the first zero REQ_I bit on the opposite bus. For each instance of the DQSM there is an associated instance of the REQ Queue Machine, which operates a local request queue counter, REQ_I_Q_y, to control the sending of requests at a given access queue priority level I on the opposite bus, Bus y.

For each Bus x, there is an instance of the Bandwidth Balancing Machine (BWBM), which controls the bandwidth balancing counter for that bus, denoted BWB_CNTR_x.

This clause gives a functional description of the operation of the Distributed Queue. The complete description of an instance of the DQSM and the associated instance of the REQ Queue Machine are given in 8.1.1 and 8.1.2, respectively, and take precedence over this clause. The description of an instance of the BWBM is given in 8.1.3 and takes precedence over this clause.

5.1.2.1.4.1 Node not queued to send

When no QA segment is queued for access to Bus x at access queue priority level I, the associated instance of the DQSM is in the Idle state (DQ1). While the DQSM is in this state, the Distributed Queue maintains the value of the associated request counter, REQ_I_CNTR_x, at the number of requests perceived at the node as unsatisfied for access by QA segments downstream on Bus x at access queue priority levels equal to or higher than I, by

— Incrementing the REQ_I_CNTR_x for any REQ_J bit set to one on the opposite bus at an access queue priority level J equal to or higher than I.
— Incrementing the REQ_I_CNTR_x for each request made by the node itself to access Bus x at an access queue priority level higher than I.
— Decrementing the REQ_I_CNTR_x for each slot which passes on Bus x with the BUSY bit and SL_TYPE bit both set to zero, i.e., an empty QA slot.
— Incrementing the REQ_I_CNTR_x when BWB_CNTR_x is reset to zero.

5.1.2.1.4.2 Node queueing to send

When a DQSM is in the DQ1 state of Idle, the Distributed Queue can accept a QA segment from the FIFO queue associated with the same bus and access queue priority level. (See 5.1.2.1.3.) If there is at least one QA segment in that FIFO queue, then the Distributed Queue accepts the QA segment at the head of the FIFO queue, transfers the value of the REQ_I_CNTR_x to the CD_I_CNTR_x, sets the REQ_I_CNTR_x to zero, and increments the value of REQ_I_Q_y, the local request queue counter at access queue priority level I for the opposite bus, Bus y. The DQSM then enters the Countdown state. (The relationship among the FIFO queue, the Distributed Queue, and the local request queue is shown in annex E.)

5.1.2.1.4.3 Sending the request

Incrementing REQ_I_Q_y causes the sending of a request at access queue priority level I on the opposite bus, Bus y, by successfully changing an REQ_I bit that was zero to one in an ACF on the opposite bus. After sending the request, REQ_I_Q_y is decremented by one. Sending the request is done under the control of the REQ Queue Machine. The request does not need to have been sent to allow the QA segment to be sent, as described in 5.1.2.1.4.5.

5.1.2.1.4.4 Node queued to send

When a QA segment is queued for access to Bus x at access queue priority level I, the associated instance of the DQSM is in the Countdown state (DQ2). While the DQSM is in this state, the Distributed Queue maintains the value of the associated countdown counter, CD_I_CNTR_x, at the number of requests for access by QA segments downstream on Bus x that have to be satisfied before the QA segment, queued locally for access to Bus x at access queue priority level I, can gain access. It does this by

— Incrementing the CD_I_CNTR_x for any REQ_J bit set to one on the opposite bus at an access queue priority level J higher than I.
— Incrementing the CD_I_CNTR_x for each request made by the node itself to access Bus x at an access queue priority level higher than I.
— Decrementing the CD_I_CNTR_x for each slot that passes on Bus x with the BUSY bit and SL_TYPE bit both set to zero, i.e., an empty QA slot.
— Incrementing the CD_I_CNTR_x when BWB_CNTR_x is reset to zero.

While the DQSM is in this state, the Distributed Queue also maintains the value of the associated request counter, REQ_I_CNTR_x, at the number of requests for access by QA segments downstream on Bus x at access queue priority level I that were made after the local QA segment was queued for access to Bus x at access queue priority level I. It does this by

— Incrementing the REQ_I_CNTR_x for any REQ_J bit set to one on the opposite bus at an access queue priority level J equal to I.

5.1.2.1.4.5 Node Sending a QA segment

When the value of the CD_I_CNTR_x associated with a QA segment queued for access to Bus x at access queue priority level I equals zero, and the Distributed Queue receives an ACF on Bus x, which has both the BUSY bit and the SL_TYPE bit set to zero (i.e., an empty QA slot), then the Distributed Queue function shall set the BUSY bit to one and shall OR-write the QA segment octets with the octets of the segment field of the QA slot as they pass along the bus in the Common Functions block.

If the value of BWB_MOD is zero, then bandwidth balancing operation is disabled. Otherwise, when the Distributed Queue changes the BUSY bit to one it shall also increment the BWB_CNTR_x, provided that the counter does not have a value of (BWB_MOD – 1), where BWB_MOD is the value of the Bandwidth Balancing Modulus. Otherwise, if the BWB_CNTR_x does have a value of (BWB_MOD – 1), then it shall be reset to zero instead. If BWB_MOD has a value of 1, then BWB_CNTR_x shall be considered as reset from zero to zero.

NOTE—The DQSM is allowed to accept another QA segment from the FIFO queue associated with the same bus and priority level, as described in 5.1.2.1.4.2, immediately upon changing the BUSY bit from zero to one.

5.1.2.1.5 Transmit interactions between QA Functions block and Common Functions block

The interaction between the QA Functions block and the Common Functions block for the transfer of the octets of a QA segment is depicted in figure 5-10.

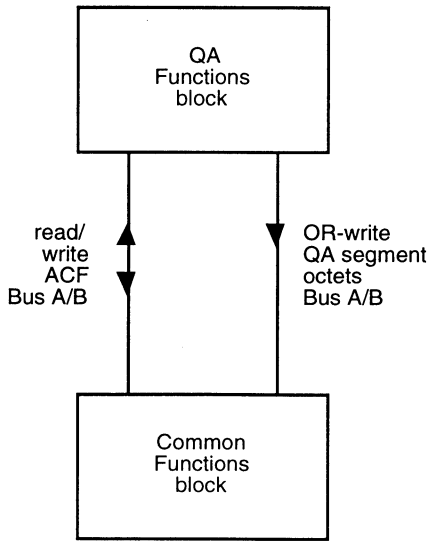


Figure 5-10—Transmit interaction between QA Functions block and Common Functions block

The Distributed Queue function of the QA Functions block needs to be able to read the BUSY bit, SL_TYPE bit, and REQUEST field of the ACF of all slots passing on both buses. The Distributed Queue function also needs to be able to write to the BUSY bit and REQUEST field of the ACF of all slots passing on both buses. When the Distributed Queue function determines that a QA segment is permitted to gain access to an empty QA slot on either bus, then the QA Functions block OR-writes the QA segment octets with the octets of the segment field of the QA slot as they pass along the bus in the Common Functions block.

5.1.2.2 QA Receive functions

This clause specifies the functions performed by the QA Functions block upon receipt at the node of an ACF with the BUSY bit set to one and the SL_TYPE bit set to zero (i.e., the ACF of a Busy, QA slot), as depicted in figure 5-11. Subclause 5.1.2.2.1 specifies how the QA segment octets of the busy QA slot are passed by the Common Functions block to the QA Functions block. Subclause 5.1.2.2.2 specifies the functions performed on the octets of the busy QA slot to collect the octets of the QA segment and the function performed on the Previous Segment Received (PSR) bit in the ACF of the subsequent slot on that bus. Subclause 5.1.2.2.3 specifies the function of validating the QA segment header. Subclause 5.1.2.2.4 specifies the function of extracting the QA segment payload and protocol control information from a QA segment with valid header, and the passing of this information to the appropriate Convergence Function block. Subclause 5.1.2.2.5 specifies how a QA segment payload is passed to the Convergence Function block, along with the data extracted from the QA segment header.

NOTE—The MCF and QA Functions Block Receive functions for a QA slot received on a bus must be done before the next slot arrives on that bus to allow the MCF block to indicate whether or not the PSR bit should be set.

5.1.2.2.1 Receive interactions between Common Functions block and QA Functions block

The interaction between the Common Functions block and the QA Functions block for the transfer of the octets of a QA segment is depicted in figure 5-12.

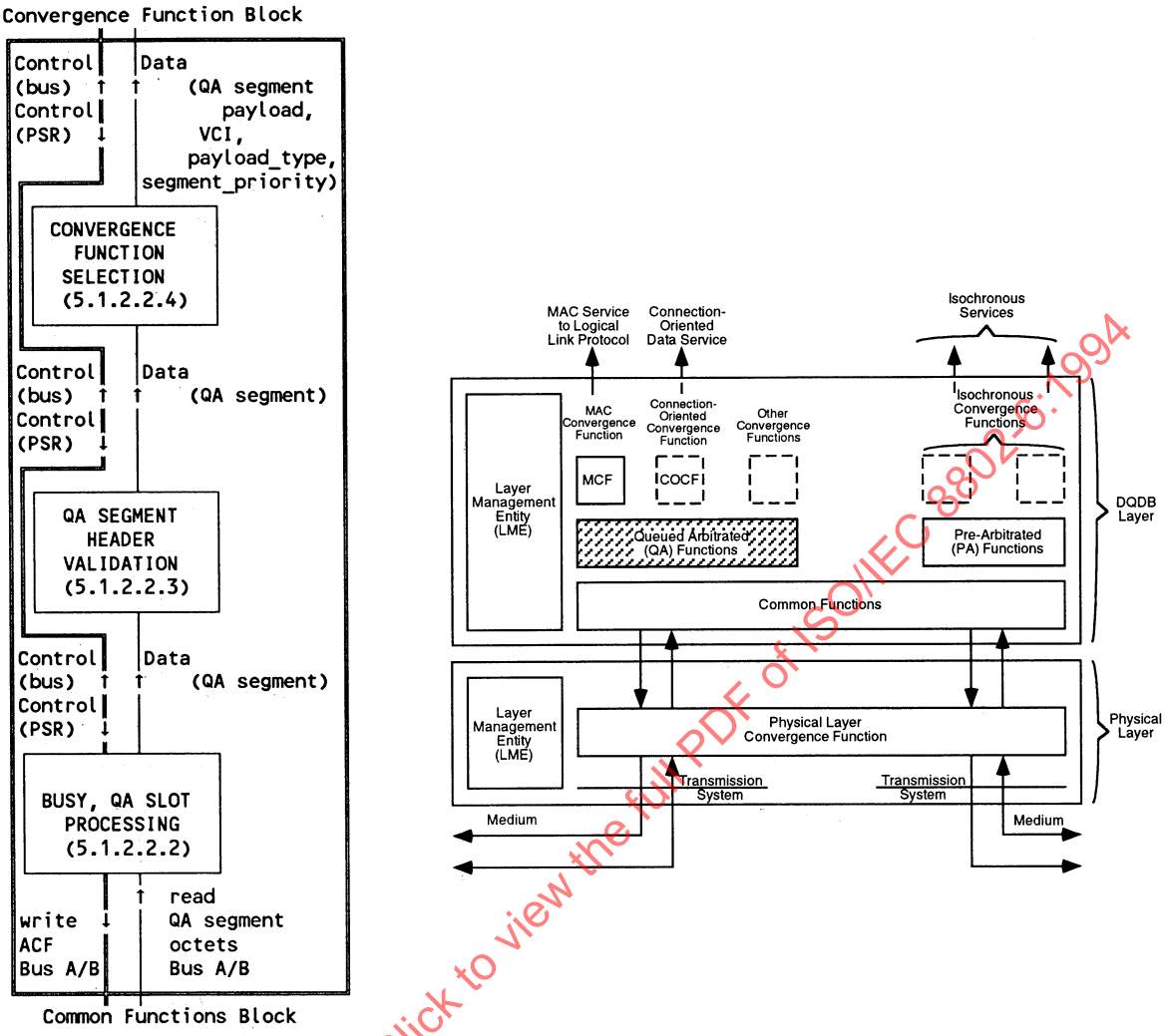


Figure 5-11—QA Functions block Receive functions

When the Common Functions block receives an ACF with the BUSY bit set to one and the SL_TYPE bit set to zero (i.e., the ACF of a busy QA slot), it then delivers a copy of the octets of the QA segment to the QA Functions block as they pass on the bus in the Common Functions block. If the QA Functions block is notified that the current QA segment received on a bus was destined only to this node, then it shall set the PSR bit to one in the ACF of the next slot on that bus, irrespective of the value of the SL_TYPE bit of the slot.

5.1.2.2.2 Busy QA slot processing

This function collects the QA segment octets received from a busy QA slot on Bus x (x = A or B) and passes the QA segment to the QA segment header validation function, described in 5.1.2.2.3.

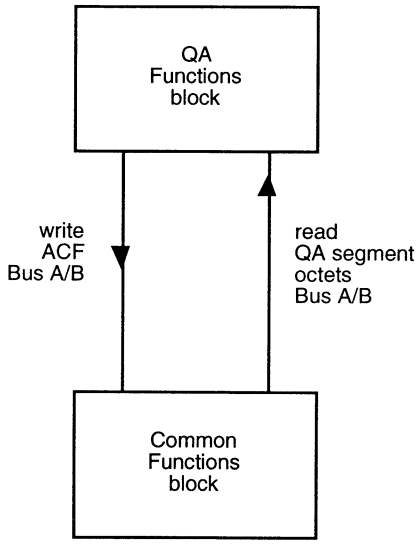


Figure 5-12—Receive interacation between Common Functions block and QA Functions block

If the QA segment header is valid, and the QA segment payload is destined only to a Convergence function at this node, then the Convergence function shall assert the PSR_x_SIGNAL (x = A or B) for the bus on which the QA segment was received. If the PSR_x_SIGNAL is to be asserted upon receipt of the QA segment payload, then it shall be asserted before the ACF of the next slot arrives on Bus x at the node. If PSR_x_SIGNAL is asserted, then the QA Functions Block function shall set the PSR bit to one in the ACF of the next slot passing on Bus x, irrespective of the SL_TYPE bit in the ACF of the slot.

5.1.2.2.3 QA segment header validation

Upon receipt of a QA segment, the QA Functions block validates the correctness of the QA segment header using the HCS. (See 6.3.1.1.4.) Then,

a) If the HCS is not valid, the QA Functions block may perform error correction on the QA segment header, using the HCS procedure described in 8.3. If error correction is not performed, then the QA segment shall be discarded.
b) If the HCS is valid, or if error correction is performed on an errored QA segment header, then the QA segment is passed to the Convergence Function Selection function, described in 5.1.2.2.4.

5.1.2.2.4 Convergence function selection

On completion of the QA Segment Header Validation function, the QA Functions block shall extract the VCI value in the QA segment header. (See 6.3.1.1.1.) It shall then examine the VCI value to determine whether it is currently associated with any Convergence Function block at this node.[24] Then,

a) If the VCI value is not associated with any Convergence Function block at the node, then the QA segment is discarded.
b) If the VCI value is associated with a Convergence Function block at the node, then the QA segment payload is passed to the appropriate Convergence Function block, along with an indication of on which bus the QA segment was received, as described in 5.1.2.2.5.

[24] In the case of the default connectionless VCI, this value is permanently associated with the MAC Convergence Function.

5.1.2.2.5 Receive interactions between QA Functions block and Convergence Function block

The interaction between the QA Functions block and the selected Convergence Function block upon the receipt at the node of a QA segment payload destined to the Convergence function is depicted in figure 5-13.

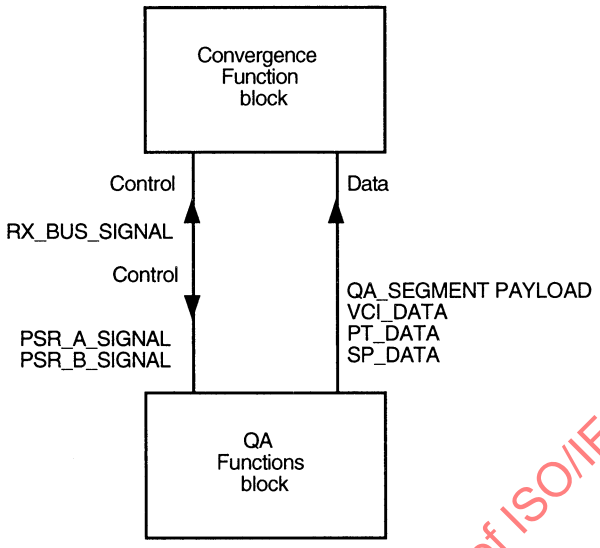


Figure 5-13—Receive interaction between QA Functions block and Convergence Function block

The QA Functions block shall perform the following operations to establish the interaction between itself and the Convergence Function block for the transfer of a QA segment payload.

a) The RX_BUS_SIGNAL control is asserted by the QA Functions block to indicate on which bus the QA segment payload was received. The values that can be associated with RX_BUS_SIGNAL are as follows:

   BUS_A, or
   BUS_B

b) The PSR_x_SIGNAL (x = A or B) is asserted by the Convergence Function block to indicate that the QA segment payload received in a slot on the bus indicated by RX_BUS_SIGNAL was destined only to this node. If the PSR_x_SIGNAL is to be asserted upon receipt of a QA segment payload, then it shall be asserted before the ACF of the next slot arrives on Bus x at the node. If PSR_x_SIGNAL is asserted, then the QA Functions block shall set the Previous Segment Received (PSR) bit to one in the ACF of the next slot passing on Bus x, irrespective of the SL_TYPE bit in the ACF of the slot.

c) The VCI_DATA contains the value of VCI received in the header of the QA segment which carried the QA segment payload. (See 6.3.1.1.1.)

d) The PT_DATA contains the value of Payload_Type received in the header of the QA segment that carried the QA segment payload. (See 6.3.1.1.2.)

e) The SP_DATA contains the value of Segment_Priority received in the header of the QA segment which carried the QA segment payload. (See 6.3.1.1.3.)

5.1.3 MAC Sublayer service management functions

All nodes conforming to this part of ISO/IEC 8802 shall support the Message Identifier (MID) Management functions and managed objects described in 9.4.1–9.4.5. All other DQDB Layer Management Interface (LMI) functions are optional for support of the MAC Sublayer service to the LLC Sublayer.

If an implementation is to support the allocation of more than the default number of MID page values, then it needs to support the management operation on the MAX_MID_PAGES system parameter, described in 9.6.1.

If an implementation is capable of recognizing connectionless VCIs additional to the default value, then the VCIs shall be constrained such that the range is expressed by the right-most bits, with all the left-most bits set to one. If such a implementation is to support connectionless VCIs for the MCF block other than the default connectionless VCI, then it needs to support the managed objects and operations described in 9.2.1 and 9.2.2.

If an implementation of the MCF block is to support the generation of nonzero-length Header Extension Fields for the transfer of an IMPDU, then it needs to support the managed objects and operations described in 9.3.1 and 9.3.2.

If an implementation is to support the recognition of MSAP addresses other than the 48-bit, universally administered address, then it needs to support the managed objects and operations described in 9.5.1 and 9.5.2.

If an implementation of the MCF block is to support access to priority level queues other than the default level specified at power-up, then it needs to support the management operation on the QOS_MAP system parameter, described in 9.6.1.

5.2 Provision of isochronous service

This clause describes the functions performed by the DQDB Layer to support the transfer of an isochronous service octet from one DQDB node to one or more peer DQDB nodes.

The isochronous service octet is received from the Isochronous Service User (ISU) at the source node as the Isochronous Service Data Unit (ISDU) in an ISU-DATA request, as defined in 3.2.1. Figure 5-14 summarizes the transmit and receive functions that are required of the ICF block and the Pre-Arbitrated (PA) Functions block to transfer an ISDU between nodes. The ISDU is delivered by each destination node in an ISU-DATA indication, as defined in 3.2.2.

5.2.1 Isochronous Convergence Function (ICF) block

The PA Functions block will receive and transmit isochronous service octets at a guaranteed average rate, but they will not necessarily be evenly distributed. The irregularity in arrival will depend on the distribution of PA slots generated by the Slot Marking function at each active Head of Bus function, and the number of octets carried for each isochronous connection by each PA slot.[25] The function of an ICF is, if necessary, to provide buffering to smooth any irregular arrival of isochronous service octets from the PA Functions block to the arrival rate expected by the Isochronous Service User (ISU).

There is one instance of an ICF block for each ISU. (See figure 5-15.) The relationship between the PA Functions block, and the ICF block and the connection end-point used to support a given ISU is established

[25] The irregularity in PA slot arrival depends on the PA slot generation attributes given to the Head of Bus function. These attributes, and their effect on the Head of Bus function, may be described in subsequent additions to this International Standard.

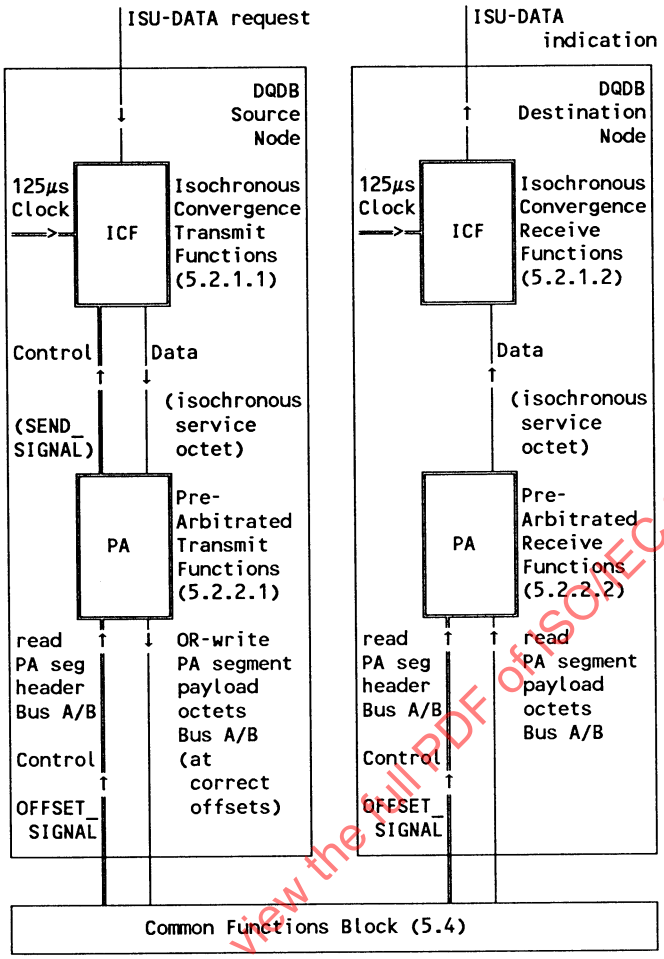


Figure 5-14—DQDB Layer functions to support isochronous service

and maintained by the optional DQDB Layer Management OPEN_CE_ICF and CLOSE_CE actions. (See 9.2.3 and 9.2.5.)

An ICF block maintains a separate transmit and receive buffer. It is expected that there will be different types of ICF defined for different types of isochronous service, depending upon the buffering requirements of the particular ISU. The basic operation of each of these buffers is described below, but their detailed operation is outside the scope of this edition of this part of ISO/IEC 8802.[26]

5.2.1.1 ICF Transmit functions

The ICF block receives ISU-DATA request primitives isochronously, according to the requirements of the ISU.

[26] These functions may be described in subsequent additions to this International Standard.

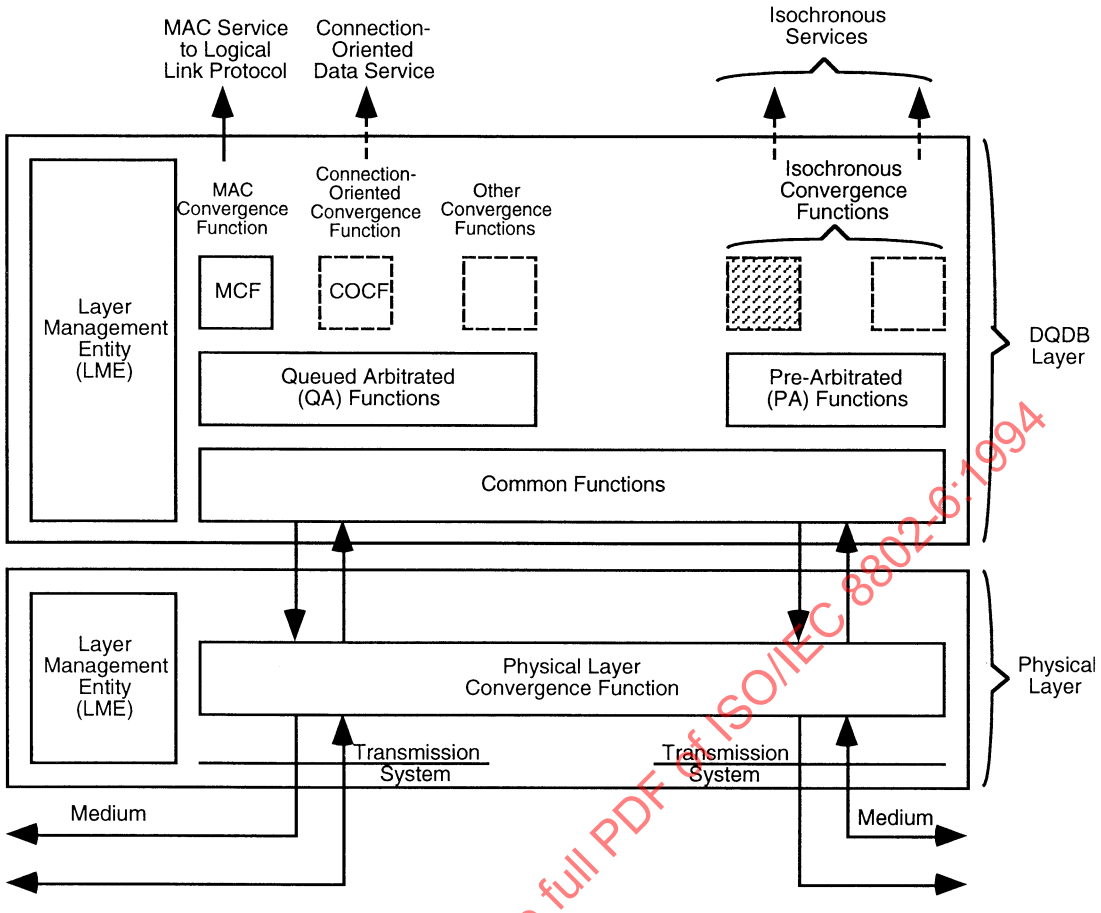


Figure 5-15—DQDB node functional architecture showing ICF block

When the ICF block receives an ISU-DATA request it places the isochronous service octet received as the Isochronous Service Data Unit (ISDU) in a transmit buffer, which is used to store the ISDUs in the order received from the ISU.

When the PA Functions block asserts the SEND_SIGNAL to the ICF block, it delivers the octet at the head of the transmit buffer to the PA Functions block.

The definition of an ICF will specify the actions taken if the transmit buffer either underflows or overflows.

5.2.1.1.1 CF Receive functions

The PA Functions block delivers isochronous octets to the ICF block as they are received, as described in 5.2.2.2.3. On receipt of an isochronous service octet, the ICF places it in a receive buffer, which is used to store the octets in the order received from the PA Functions block.

The ICF block is required to generate ISU-DATA indication primitives according to the requirements of the ISU. To support this function, the ICF block uses a 125 µs clock derived by the DQDB Layer subsystem from the Ph-TIMING-MARK indication primitives (see 4.5) received at the node.

When the ICF block is required to send an ISU-DATA indication, it delivers the octet at the head of the receive buffer as the ISDU in an ISU-DATA indication.

The definition of an ICF will specify the actions taken if the receive buffer either underflows or overflows.

5.2.2 Pre-Arbitrated (PA) Functions block

The PA Functions block (see figure 5-16) controls the transfer in PA segment payloads of isochronous service octets received from ICF blocks.

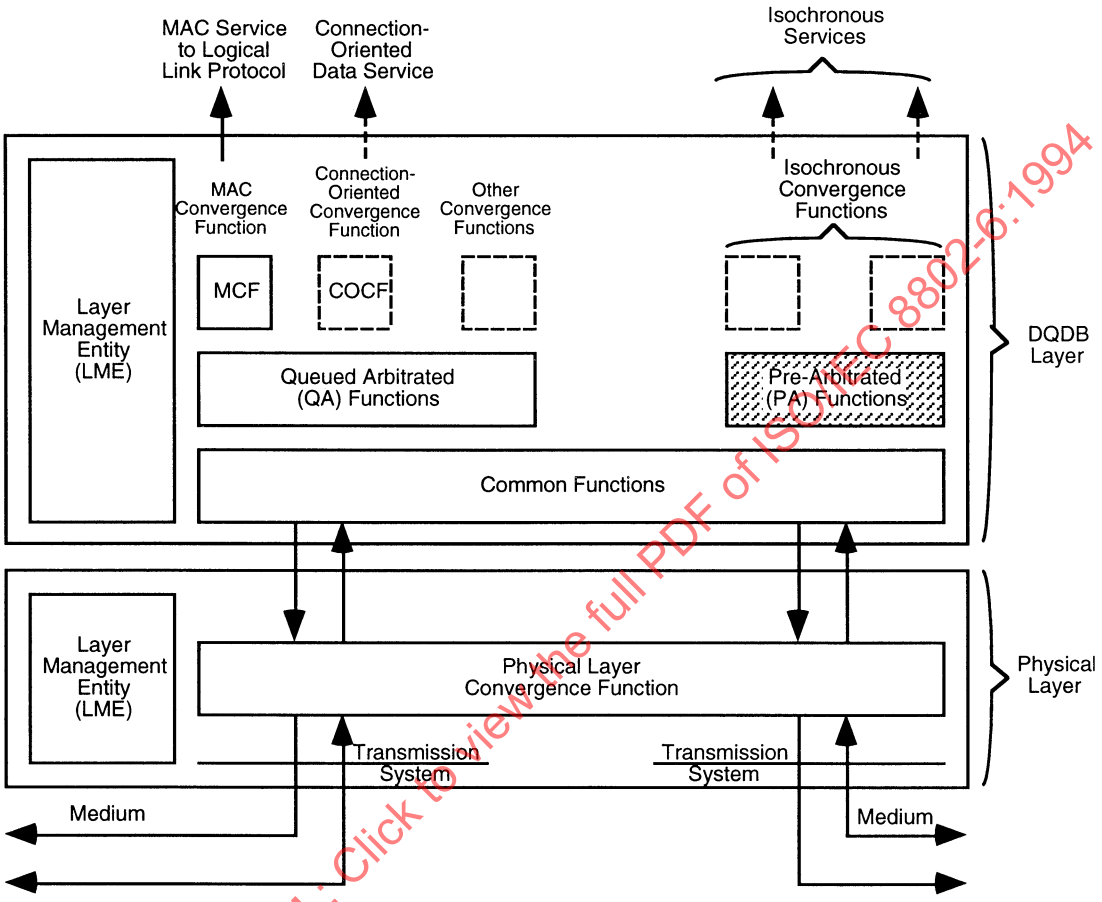


Figure 5-16—DQDB node functional architecture showing PA Functions block

The relationship between the PA Functions block and ICF blocks is established by the optional OPEN_CE_ICF and CLOSE_CE actions. (See 9.2.3 and 9.2.5.) These actions establish the transmit and receive bus and associated set of [VCI,offset] pairs for each bus used by the PA Functions block to transmit and receive isochronous service octets for the ICF blocks. Hence, the OPEN_CE_ICF and CLOSE_CE actions maintain the list of [VCI,offset] pairs that are currently being used at the node to transmit or receive isochronous service octets in PA segment payloads on either bus for any ICF.

The transmit and receive interactions between the PA Functions block and the Common Functions block make use of the OFFSET_SIGNAL control. This control is used by the Common Functions block to convey the position of each octet of a PA segment payload as an offset relative to the start of the PA segment payload. The value of OFFSET_SIGNAL control is an integer in the range 1 to 48, inclusive.

5.2.2.1 PA Transmit functions

The PA Transmit functions involve examining the VCI in the PA segment header of PA slots received on Bus x (x = A or B), and determining whether there are any offset values for this PA segment payload that are currently associated with transmission for an ICF block at the node. Subclause 5.2.2.1.1 specifies how the PA segment header octets and OFFSET_SIGNAL control are passed from the Common Functions block to the PA Functions block, and how octets that are to be written into the matching offsets of the PA segment payload are transferred from the PA Functions block to the Common Functions block. Subclause 5.2.2.1.2 specifies the function of validating the PA segment header. Subclause 5.2.2.1.3 specifies the operation of the PA Functions block to obtain an isochronous service octet from an ICF block to write at a matching offset in this PA segment payload.

5.2.2.1.1 Transmit interactions between PA Functions block and Common Functions block

The transmit interaction between the Common Functions block and the PA Functions block for the transfer of the octets of a PA segment header and PA segment payload is depicted in figure 5-17.

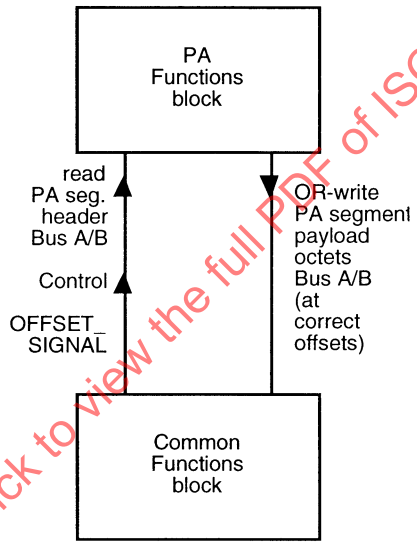


Figure 5-17—Transmit interaction between PA Functions block and Common Functions block

When the Common Functions block receives an ACF with the BUSY bit set to one and the SL_TYPE bit set to one (i.e., the ACF of a PA slot), it then delivers a copy of the octets of the PA segment header to the PA Functions block as they pass on the bus in the Common Functions block. It also asserts the OFFSET_SIGNAL control (see 5.2.2) to the offset of the PA segment payload as each octet passes on the bus in the Common Functions block. If the PA Functions block is to write an octet at the offset asserted by OFFSET_SIGNAL, then it OR-writes it as the octet passes along the bus in the Common Functions block.

5.2.2.1.2 PA segment header validation

Upon receipt of a PA segment header, the PA Functions block validates the correctness of the PA segment header using the HCS. (See 6.4.1.1.4.) Then,

a) If the HCS is not valid, the PA Functions block may perform error correction on the PA segment header, using the HCS procedure described in 8.3. If error correction is not performed, then the PA segment payload shall be ignored.
b) If the HCS is valid, or if error correction is performed on an errored PA segment header, then the value of VCI received in the PA segment header, or as corrected, is used for processing the associated PA segment payload, as described in 5.2.2.1.3.

5.2.2.1.3 Multiuser access to PA segment payloads

If the VCI in the PA segment header is accepted as valid, according to 5.2.2.1.2, then the PA Functions block stores the VCI and the bus on which it was received. The PA Functions block then takes each OFFSET_SIGNAL value asserted for this PA segment payload and compares it, along with the VCI and bus, with the set of transmit [VCI,bus,offset] values associated with all ICF blocks at the node. Then,

a) If the comparison does not match the transmit [VCI,bus,offset] value for any ICF block at this node no action is taken.
b) If the comparison does match the transmit [VCI,bus,offset] value for an ICF block at this node, the PA Functions block asserts the SEND_SIGNAL control to this ICF block and OR-writes the octet received from the ICF block with the octet at this offset as it passes along the bus in the Common Functions block.

5.2.2.2 PA Receive functions

The PA Receive functions involve examining the VCI in the PA segment header of PA slots received on Bus x (x = A or B), and determining whether there are any offset values for this PA segment payload that are currently associated with reception for an ICF block at the node. Subclause 5.2.2.2.1 specifies how the PA segment header octets, PA segment payload octets, and OFFSET_SIGNAL control are passed from the Common Functions block to the PA Functions block. Subclause 5.2.2.2.2 specifies the function of validating the PA segment header. Subclause 5.2.2.2.3 specifies the operation of the PA Functions block to send an isochronous service octet received at a matching offset of the PA segment payload to the appropriate ICF block.

5.2.2.2.1 Receive interactions between PA Functions block and Common Functions block

The receive interaction between the Common Functions block and the PA Functions block for the transfer of the octets of a PA segment header and PA segment payload is depicted in figure 5-18.

When the Common Functions block receives an ACF with the BUSY bit set to one and the SL_TYPE bit set to one (i.e., the ACF of a PA slot), it then delivers a copy of the octets of the PA segment header to the PA Functions block as they pass on the bus in the Common Functions block. It also delivers a copy of each octet of the PA segment payload to the PA Functions block and asserts the OFFSET_SIGNAL control (see 5.2.2) to the offset of the PA segment payload as the octet passes on the bus in the Common Functions block.

5.2.2.2.2 PA segment header validation

Upon receipt of a PA segment header, the PA Functions block validates the correctness of the PA segment header using the HCS. (See 6.4.1.1.4.) Then,

a) If the HCS is not valid, the PA Functions block may perform error correction on the PA segment header, using the HCS procedure described in 8.3. If error correction is not performed, then the PA segment payload shall be ignored.
b) If the HCS is valid, or if error correction is performed on an errored PA segment header, then the value of VCI received in the PA segment header, or as corrected, is used for processing the associated PA segment payload, as described in 5.2.2.2.3.

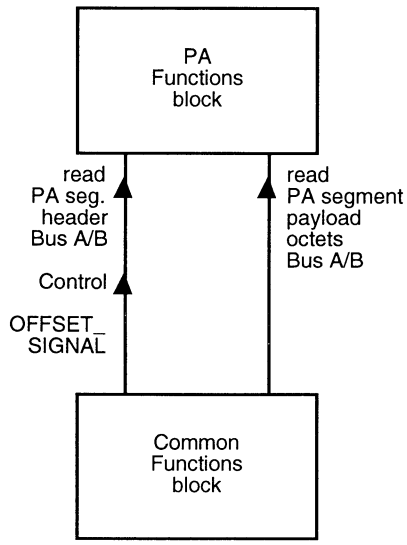


Figure 5-18—Receive interaction between Common Functions block and PA Functions block

5.2.2.2.3 ICF selection

If the VCI in the PA segment header is accepted as valid, according to 5.2.2.2.2, then the PA Functions block stores the VCI and the bus on which it was received. The PA Functions block then takes each OFFSET_SIGNAL value asserted for this PA segment payload and compares it, along with the VCI and bus, with the set of receive [VCI,bus,offset] values associated with all ICF blocks at the node. Then,

a) If the comparison does not match the receive [VCI,bus,offset] value for any ICF block at this node no action is taken.
b) If the comparison does match the receive [VCI,bus,offset] value for an ICF block at this node, the PA Functions block delivers the octet at this offset to that ICF block.

5.2.3 Isochronous service provider management functions

Nodes conforming to this version of this part of ISO/IEC 8802 are not required to support the isochronous service. If a conforming node is to support the isochronous service, then it needs to support the managed objects and operations described in 9.2.3 and 9.2.5.

5.3 Provision of other services

This clause gives guidelines for additional services that may be defined in future versions of this part of ISO/IEC 8802.[27]

5.3.1 Connection-Oriented Data Service

This clause outlines the functions expected to be performed by the DQDB Layer to support the transfer of data for the Connection-Oriented Data Service User from one DQDB node to one or more peer DQDB nodes.

[27] These services, and the DQDB Layer functions required to support them, may be described in subsequent additions to this International Standard.

Figure 5-19 summarizes the transmit and receive functions that will be required of the Connection-Oriented Convergence Function (COCF) block and the QA Functions block to transfer data between nodes.

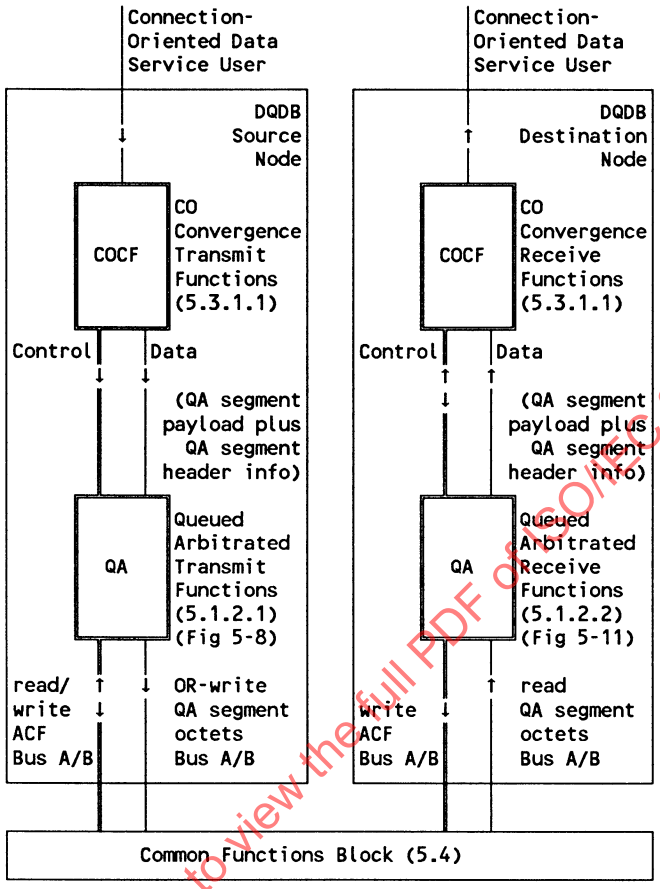


Figure 5-19—DQDB Layer functions to support Connection-Oriented Data Service

5.3.1.1 Characteristics for the Connection-Oriented Convergence Function (COCF)

The COCF block (see figure 5-20) will use the same segmentation and reassembly procedures as already defined for the MAC Convergence Function (MCF). Therefore, any data transferred by the COCF will carry the Common PDU header and Common PDU trailer defined in 6.5.1.1 and 6.5.1.7, respectively, for the transfer of an Initial MAC PDU. However, the use of some fields of the Common PDU header and Common PDU trailer will differ for the COCF when compared with the use defined for the MCF, as outlined below.

— The Reserved field in the Common PDU header (6.5.1.1.1) and the Reserved field in the Common PDU trailer (6.5.1.7.1) may contain nonzero values.

— The BAsize field in the Common PDU header (6.5.1.1.3) may be set by the COCF to a value equal to or larger than the Length field in the Common PDU trailer (6.5.1.7.3) for the same data transfer. This field is intended for buffer allocation purposes at the destination node.

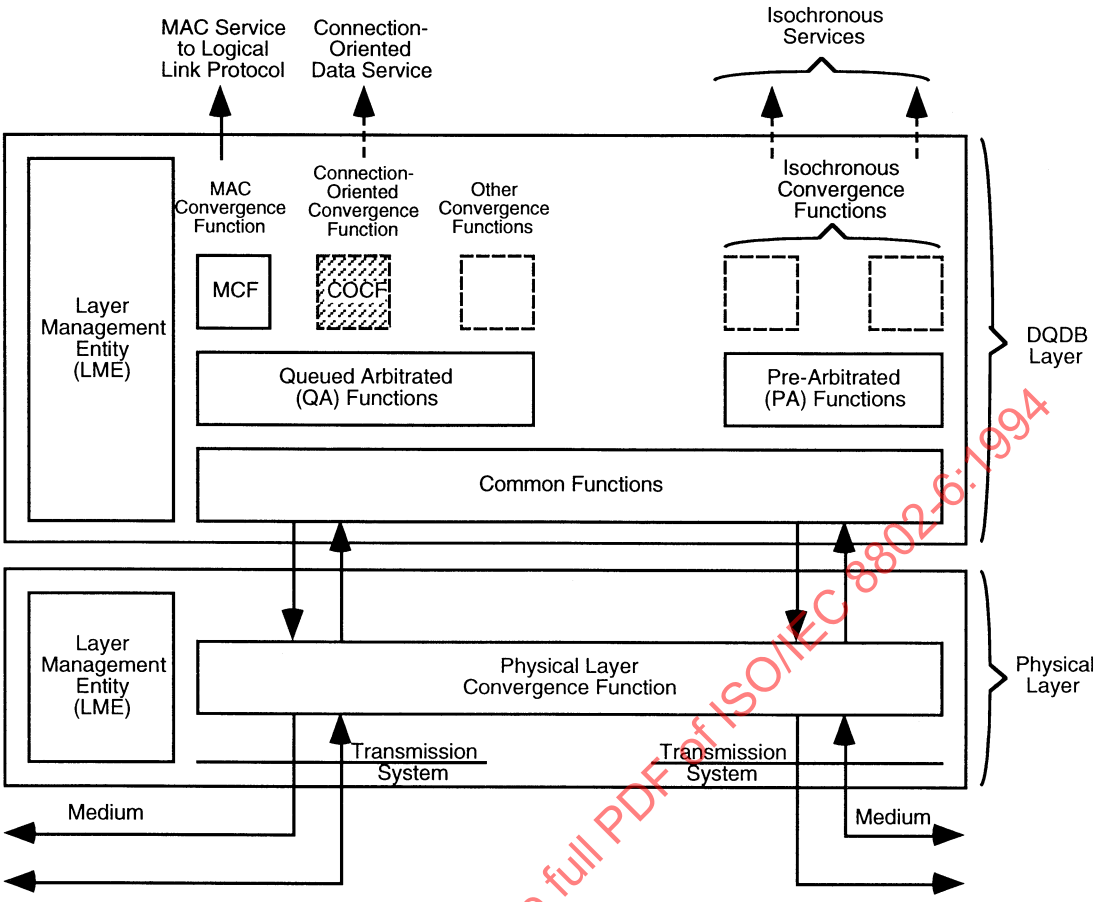


Figure 5-20—DQDB node functional architecture showing COCF block

The QA segment payloads generated by the COCF to carry data will also carry a header and trailer that are identical in format to the DMPDU header and DMPDU trailer defined in 6.5.2.1 and 6.5.2.2, respectively, for the transfer of a Derived MAC PDU. However, the use of the Payload_Length field will differ for data transfers by the COCF, as partial filling of BOM and COM segmentation units may be allowed. This function is expected to support pipelining.

The COCF will also support the PAD field associated with the Common PDU trailer. This is to ensure that the COCF data transfers are 32-bit aligned.

5.3.1.2 Interactions between COCF block and QA Functions block

The relationship between the QA Functions block and COCF blocks is established by the optional DQDB Layer Management OPEN_CE_COCF and CLOSE_CE actions (9.2.4 and 9.2.5). These actions establish the bus and VCI to be used to transmit and receive QA segment payloads for the COCF block. The actions also establish the Segment_Priority and access queue priority level to be used to transmit QA segment payloads.

5.3.1.2.1 Transmit interactions between COCF block and QA Functions block

The interaction between a COCF block and the QA Functions block for the transfer of a QA segment payload is depicted in figure 5-21.

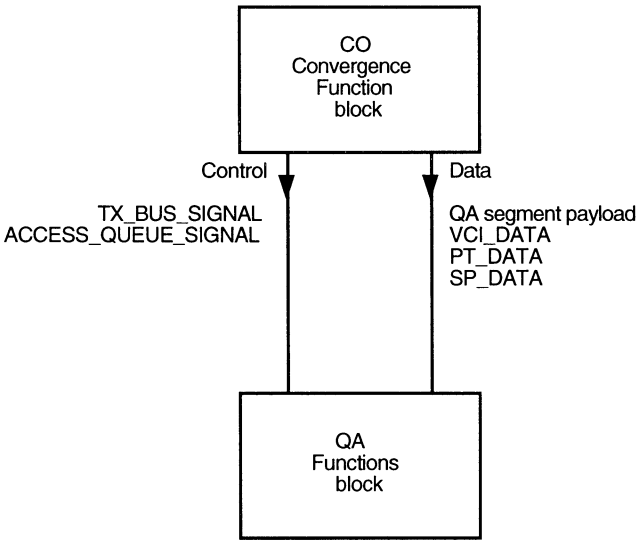


Figure 5-21—Transmit interaction between COCF block and QA Functions block

The TX_BUS_SIGNAL, ACCESS_QUEUE_SIGNAL, VCI_DATA, and SP_DATA are set to the values of tx_bus, tx_access_queue_priority, tx_vci, and tx_segment_priority, respectively, which are received in an OPEN_CE_ COCF action. The PT_DATA value will be provided for each QA segment payload either by the Connection-Oriented Data Service User or by the COCF.

5.3.1.2.2 Receive interactions between QA Functions block and COCF block

The interaction between the QA Functions block and the COCF block upon the receipt at the node of a QA segment payload destined to the COCF block is depicted in figure 5-22.

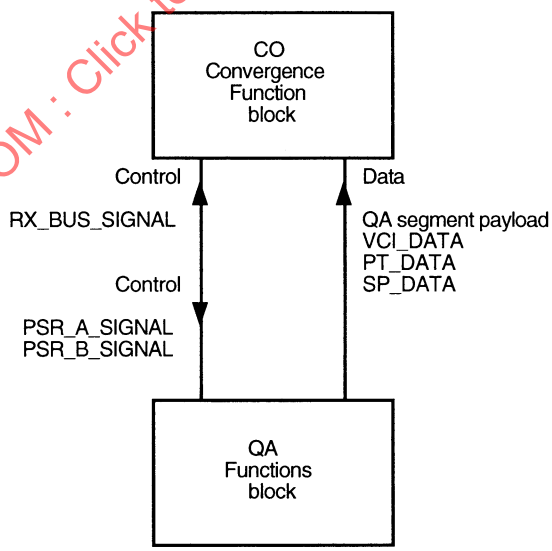


Figure 5-22—Receive interaction between QA Functions block and COCF block

The RX_BUS_SIGNAL and VCI_DATA are set to the values of rx_bus and rx_vci, respectively, which are received in an OPEN_CE_COCF action. The PSR_x_SIGNAL (x = A or B) is asserted upon receipt of any QA segment payload at the COCF block. The PT_DATA and SP_DATA contain the value of Payload_Type and Segment_Priority, respectively, which is received in the header of the QA segment which carried the QA segment payload.

5.4 Common Functions

The Common Functions block (see figure 5-23) provides functions that are needed by some or all of the other functional blocks in the local DQDB Layer subsystem.

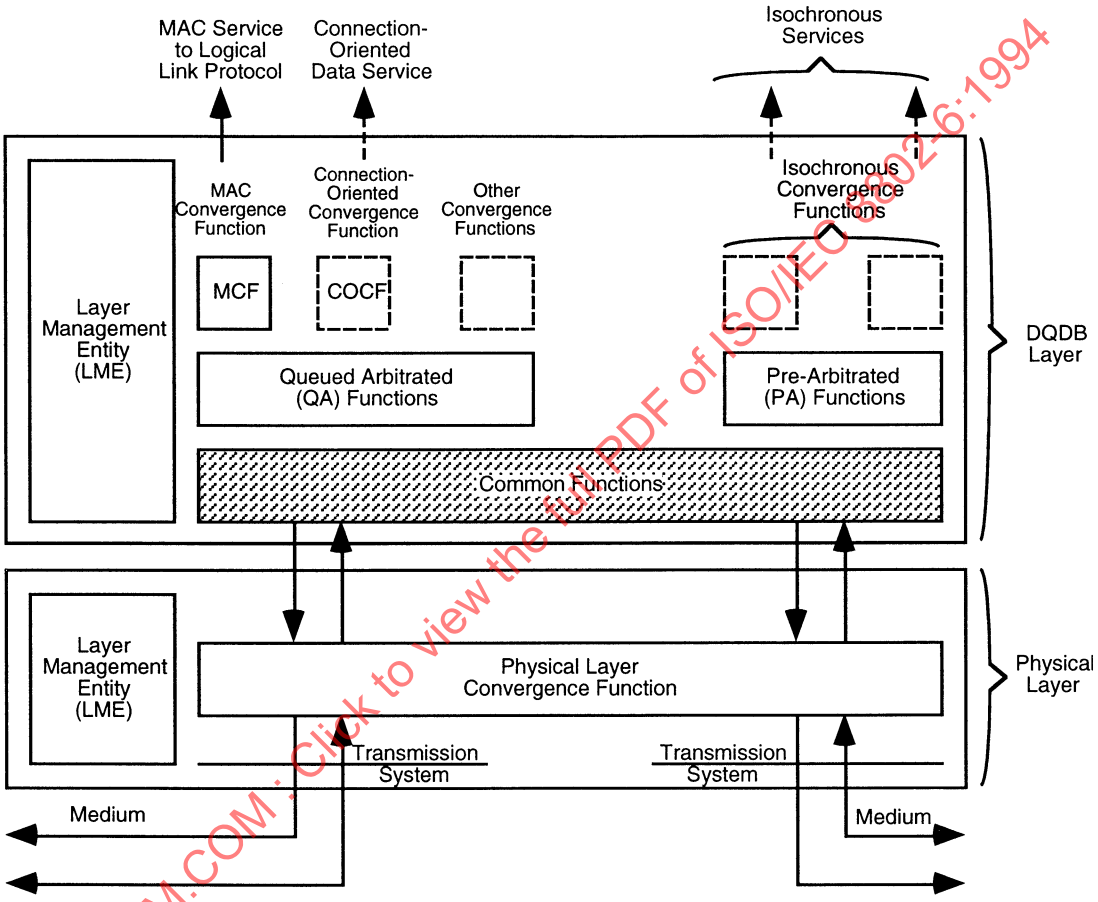


Figure 5-23—DQDB node functional architecture showing Common Functions block

The Common Functions block provides the DQDB Layer function of relaying the slot octets and management information octets between the two service access points to the local Physical Layer subsystem, as required by the Physical Layer facilities described in 4.3. As part of this function, the Common Functions block allows the QA and PA Functions blocks to gain read and write access to the slot octets as they are relayed, as described in 5.4.1.1–5.4.1.4.

The Subnetwork Configuration Control function is responsible for ensuring that the resources of all nodes of a subnetwork are configured into a correct dual bus topology. The MID Page Allocation function ensures that MID page values are allocated uniquely to nodes on a subnetwork-wide basis. Both of these functions

manage DQDB Layer objects necessary for nodes to communicate on the subnetwork, and hence are part of the DQDB Layer Management Entity (LME). However, the DQDB LMEs at all nodes must communicate to support these functions, by using the DQDB Layer Management Protocol described in clause 10.

The DQDB Layer Management Protocol Entity in the Common Functions block supports the transfer of the management information octets used to support the DQDB Layer Management Protocol. This function is described in 5.4.1.

The functional requirements for Subnetwork Configuration Control are described in 5.4.2. The Configuration Control aspects of the DQDB Layer Management Protocol are described in 10.2.

The functional requirements at a node for MID Page Allocation are described in 5.4.4.2. The MID Page Allocation aspects of the DQDB Layer Management Protocol are described in 10.3.

One of the resources controlled by the Configuration Control function is the Head of Bus function. This includes the Slot Marking function, which is the process of marking Pre-Arbitrated slots to be written at the head of bus. The Slot Marking function is described in 5.4.3.1. The Head of Bus function also includes functions required of the DQDB Layer Management Protocol Entity, including the Bus Identification functions, described in 5.4.3.3, and the MID Page Allocation functions at the head of bus, described in 5.4.4.1.

5.4.1 Relaying of slot octets and management information octets

There is a relationship between the generation of Ph-DATA indication primitives at each Physical Layer service access point (Ph-SAP) and the receipt of Ph-DATA request primitives at both Ph-SAPs of the node. The relationship depends on whether or not the node is performing the function of head of a bus, and it is described in 4.3. The Common Functions block provides the DQDB Layer function of relaying the slot octets and management information octets received in Ph-DATA indication primitives between the two Ph-SAPs in all cases described in 4.3.

Figure 5-24 presents a model for describing the relay function. The model presented shows two separate data paths for the DQDB_MANAGEMENT octets and the SLOT_START and SLOT_DATA octets. The selector function determines which of the two data paths into the node to choose for an octet upon receipt of a Ph-DATA indication, based upon the type of the octet. The selector function also determines which outgoing data path to choose to obtain an octet upon the need to generate a Ph-DATA request. The actual function provided by each Slot Generator function and the DQDB Layer Management Protocol Entity depends on whether or not the node is performing Head of Bus functions, and is defined in 5.4.3.1 and 5.4.3.2.

Note that figure 5-24 identifies the logical separation of the QA and PA Functions blocks. The figure is not meant to imply that the QA Functions block and PA Functions block must be functionally or physically placed in any particular relationship to Ph-SAP_A and Ph-SAP_B, respectively.

5.4.1.1 Transmit interactions between QA Functions block and Common Functions block

The interaction between the QA Functions block and the Common Functions block for the transfer of the octets of a QA segment is depicted in figure 5-25.

The QA Functions block needs to be able to read the BUSY bit, SL_TYPE bit, and REQUEST field of the SLOT_START octets passing on both buses. The QA Functions block also needs to be able to write to the BUSY bit and REQUEST field of the SLOT_START octets passing on both buses. When the QA Functions block determines that a QA segment is permitted to gain access to an empty QA slot on either bus, then the QA Functions block OR-writes the QA segment octets with the SLOT_DATA octets as they pass along the bus.

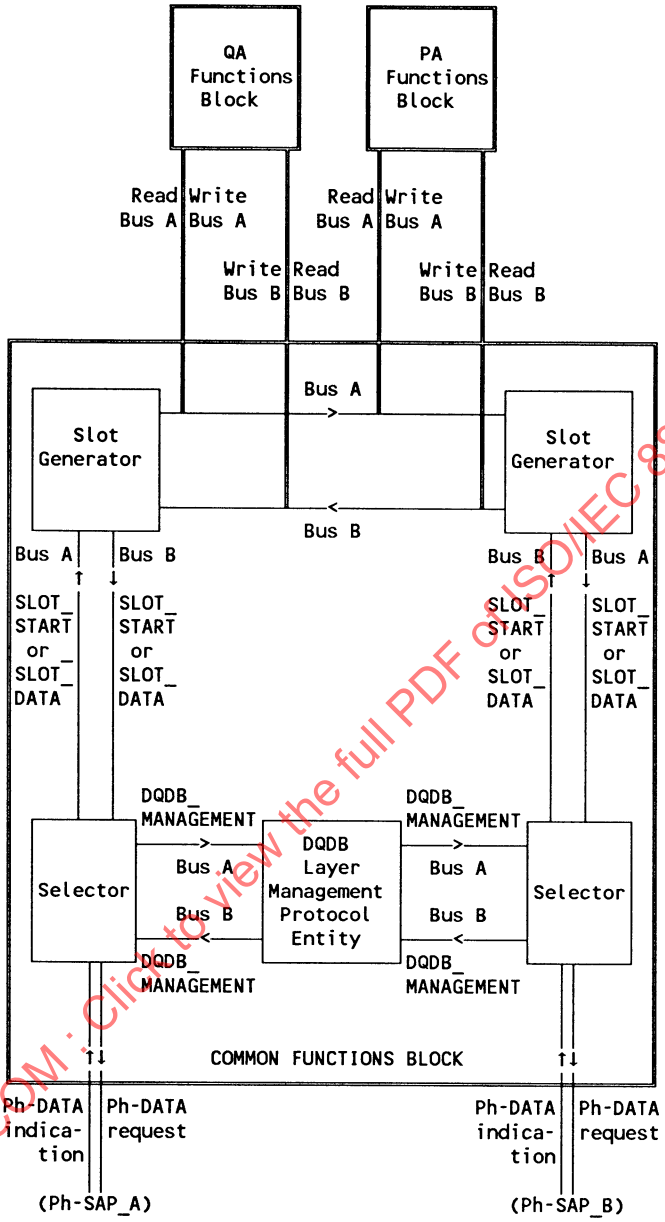


Figure 5-24—Functional data path for the Common Functions block

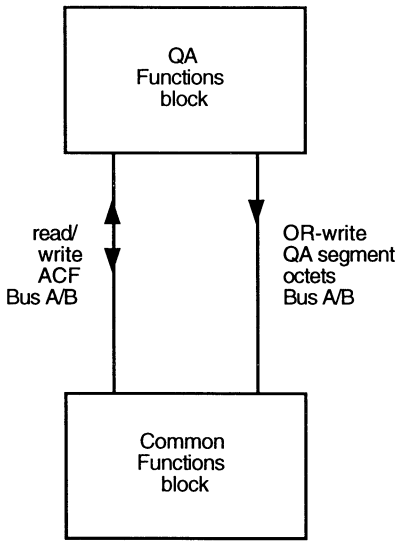


Figure 5-25—Transmit interaction between QA Functions block and Common Functions block

5.4.1.2 Receive interactions between Common Functions block and QA Functions block

The interaction between the Common Functions block and the QA Functions block for the transfer of the octets of a QA segment is depicted in figure 5-26.

When the Common Functions block receives a SLOT_START octet with the BUSY bit set to one and the SL_TYPE bit set to zero (i.e., the ACF of a busy QA slot), it then delivers a copy of the SLOT_DATA octets to the QA Functions block as they pass on the bus. If the QA Functions block is notified that the QA segment formed by the SLOT_DATA octets received on a bus was destined only to this node, then it shall set the PSR bit to one in the next SLOT_START octet on that bus, irrespective of the value of the SL_TYPE bit of the slot.

5.4.1.3 Transmit interactions between PA Functions block and Common Functions block

The transmit interaction between the Common Functions block and the PA Functions block for the transfer of the octets of a PA segment header and PA segment payload is depicted in figure 5-27.

When the Common Functions block receives a SLOT_START octet with the BUSY bit set to one and the SL_TYPE bit set to one (i.e., the ACF of a PA slot), it then delivers a copy of the first four SLOT_DATA octets (the PA segment header) to the PA Functions block as they pass on the bus. It also asserts the OFFSET_SIGNAL control (see 5.2.2) to the offset of the subsequent 48 SLOT_DATA octets (the PA segment payload) as each passes on the bus. If the PA Functions block is to write an octet at the offset asserted by OFFSET_SIGNAL, then it OR-writes it as the SLOT_DATA octet at that offset passes along the bus.

5.4.1.4 Receive Interactions between PA Functions block and Common Functions block

The receive interaction between the Common Functions block and the PA Functions block for the transfer of the octets of a PA segment header and PA segment payload is depicted in figure 5-28.

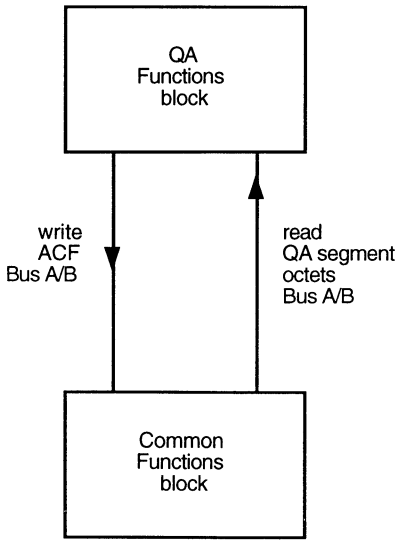


Figure 5-26—Receive interaction between Common Functions block and QA Functions block

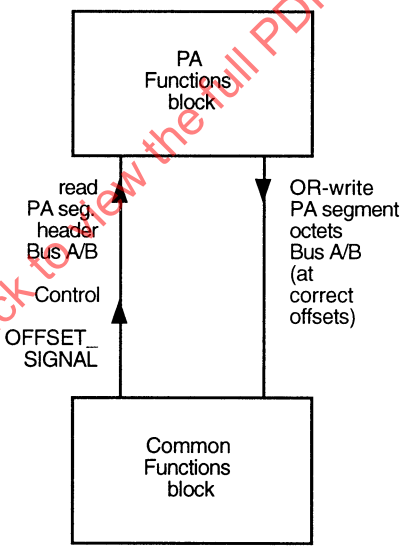


Figure 5-27—Transmit interaction between PA Functions block and Common Functions block

When the Common Functions block receives a SLOT_START octet with the BUSY bit set to one and the SL_TYPE bit set to one (i.e., the ACF of a PA slot), it then delivers a copy of the first four SLOT_DATA octets (the PA segment header) to the PA Functions block as they pass on the bus. It also delivers a copy of each of the subsequent 48 SLOT_DATA octets (the PA segment payload) to the PA Functions block and asserts the OFFSET_SIGNAL control (see 5.2.2) to the offset for each SLOT_DATA octet as it passes on the bus.

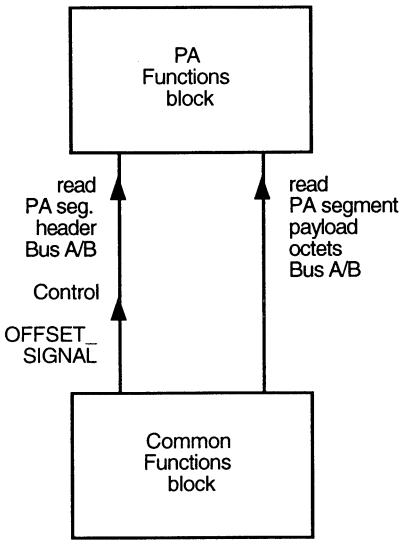


Figure 5-28—Receive interaction between Common Functions block and PA Functions block

5.4.2 Subnetwork Configuration Control function

The Subnetwork Configuration Control function is responsible for ensuring that the resources of all nodes of a subnetwork are configured into a correct dual bus topology. The resources that are managed support Head of Bus functions, Subnetwork Timing Reference functions, and Bus Identification functions. If these resources are not configured properly, the Configuration Control protocol, defined in 10.2, operates to cause the activation and deactivation of resources at appropriate nodes.

The Subnetwork Configuration Control function is based on a set of fundamental requirements, outlined in 5.4.2.1, and the Common Functions block architecture, which supports the node Configuration Control function, and is outlined in annex F.

5.4.2.1 Fundamental subnetwork requirements

The following fundamental requirements, FR1, FR2, and FR3, must be satisfied for any stable DQDB subnetwork to operate correctly. The requirements may be met by any suitable combination of resources on a subnetwork-wide basis. Explanatory notes in this clause are contained in paragraphs with a leading "•" symbol.

FR1. There shall be one Head of Bus A function and one Head of Bus B function operating in a stable subnetwork.

- Subclause 5.4.2.3 summarizes the allowed combinations of resources that meet this requirement on a subnetwork-wide basis. An active Head of Bus function is responsible for the Slot Marking function, described in 5.4.3.1, and the generation of appropriate values in the Bus Identification Field (BIF) and MID Page Allocation Field, described in 5.4.3.3 and 5.4.4.1, respectively.

FR2. There shall be one 125 µs timing reference for all nodes of a stable subnetwork.

- Subclause 5.4.2.4 summarizes the allowed combinations of resources that meet this requirement on a subnetwork-wide basis. FR2 is necessary because the slot rate must be the same on both buses, for the following reasons:

1) Stable operation of the Distributed Queue requires that the frequency of Request bits on each bus must be equal to or greater than the frequency of empty QA slots on the other bus. Since the number of Request bits at each level is equal to the number of slots, then the slot rates must be the same on the two buses.[28]

2) Support for some isochronous communications without the occurrence of slips requires that both ends of the isochronous communication are operating at the same rate.

FR3a. There shall be one node that provides the function of defining the identity of the buses for a subnetwork. After start-up of the subnetwork, the identity of each bus shall remain unchanged while the subnetwork remains operational.

- FR3a is necessary to ensure correct operation of the MID Page Allocation function, which operates differently on Bus A and Bus B, as described in 5.4.4. FR3a is supported by selecting one node prior to the start-up of each subnetwork to perform the function of defining bus identity. This node contains the active Default Slot Generator function, described architecturally in annex F.

FR3b. If the subnetwork is stable in a looped dual bus configuration, then there shall be one point at one node that provides both the Head of Bus A and Head of Bus B functions. Consequently, this shall also be the point where Bus A and Bus B terminate.

- FR3b is necessary to allow the Distributed Queue to operate properly and also to prevent formation of a ring configuration. For a subnetwork that will stabilize as a looped dual bus configuration, FR3b is supported by the same function in that subnetwork that supports FR3a, i.e., the active Default Slot Generator function.

FR3a and FR3b are incorporated in Fundamental Subnetwork Requirement 3, FR3:

FR3. All subnetworks shall contain exactly one node with an active Default Slot Generator (SG_D) function when started up. The mechanism by which this node is selected is outside the scope of this part of ISO/IEC 8802.[29] The SG_D function shall provide the function of defining the bus identity for a subnetwork. The bus that leaves the SG_D function and passes the QA Functions block and PA Functions block of the node is defined to be Bus A, as shown in figure 5-29. The SG_D function shall also provide the Head of Bus A and Head of Bus B functions for a subnetwork in a stable looped dual bus configuration.

5.4.2.2 Start-up of nodes not supporting default Slot Generator function

According to Fundamental Subnetwork Requirement FR3, bus identity is determined prior to subnetwork start-up. However, given that the Slot Generator Type 1 (SG_1) and Type 2 (SG_2) functions are not equivalent and that the MID Page Allocation function operates differently on the two buses, a node that contains the SG_1/SG_2 pair of functions (see 10.2.1) cannot associate the Slot Generator functions with a Physical Layer Service Access Point (Ph-SAP) until it receives notification of the identity of at least one bus.

Hence, all nodes that contain a Slot Generator Type 1 and Slot Generator Type 2 function shall start up without prior knowledge of bus identity, as shown in figure 5-30. Such a node shall only associate a label with the Ph-SAPs at the node, and thus the Slot Generator functions with the respective Ph-SAPs (as shown in figure 5-31), when the node receives a BIF containing a value of either Bus A or Bus B (see 10.1.1) from either Ph-SAP.

---

[28] The mechanism that would cause instability in the Distributed Queue for the case of unequal slot rates is described in annex D.

[29] An example of one mechanism would be a hardware switch set to the inactive position for all nodes except the selected node.

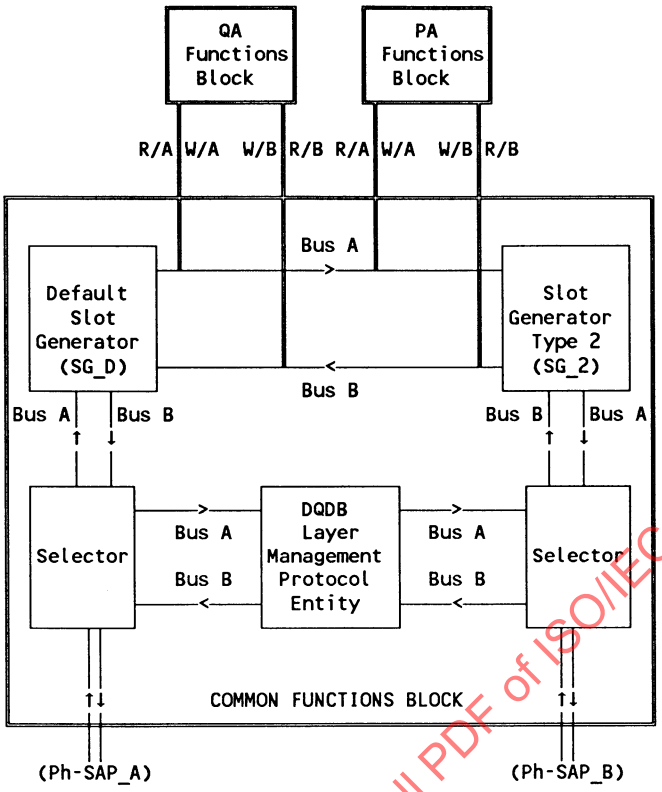


Figure 5-29—Slot generator designations for the node supporting the Default Slot Generator function

5.4.2.3 Summary of subnetwork head of bus requirements

If the subnetwork is in a stable looped Dual Bus configuration, the Head of Bus A function and Head of Bus B function are both performed by the Default Slot Generator (SG_D) function.

If the subnetwork is in a stable open dual bus configuration, the Head of Bus A function is provided by the node that has the capability and is most upstream on Bus A. If this node contains the SG_D function, the SG_D function provides the Head of Bus A function. If this node does not contain the SG_D function, the Slot Generator Type 1 (SG_1) function at the node provides the Head of Bus A function.

If the subnetwork is in a stable open dual bus configuration, the Head of Bus B function is provided by the node that has the capability and is most upstream on Bus B. The Slot Generator Type 2 (SG_2) function at this node provides the Head of Bus B function, irrespective of whether or not the node contains the SG_D function.

If a node is not providing the subnetwork 125 µs timing reference (as defined in 5.4.2.4.1), but is providing the function of head of a bus, then it shall use 125 µs timing derived according to 5.4.2.4.4.

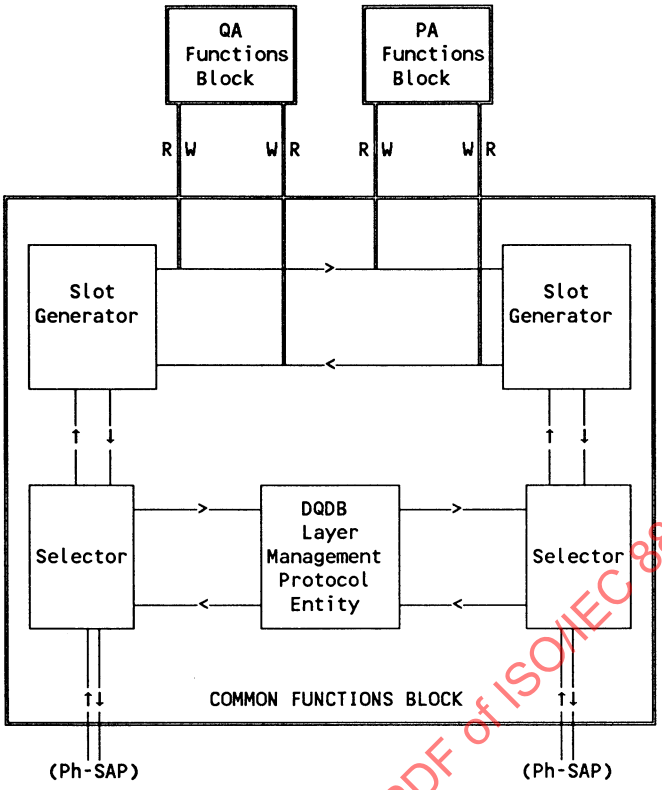


Figure 5-30—Slot generator designations for a node waiting to receive valid bus identification at startup

5.4.2.4 Subnetwork timing reference Configuration Control functions

5.4.2.4.1 Hierarchy for selection of primary subnetwork timing reference

In order to support Fundamental Subnetwork Requirement FR2, two types of 125 µs timing may be present in a subnetwork:

a) A timing reference provided externally to the DQDB subnetwork by a public network provider.
b) A timing reference provided by the node clock at one designated node in the subnetwork.

All subnetworks must possess a 125 µs node timing capability, as described in b) above. A subnetwork may possess an external timing capability, as described in a) above. This clause specifies the hierarchy used to select the primary subnetwork timing reference from the available timing sources. The rationale for this hierarchy is provided in annex G.

Hierarchy. The choice of primary 125 µs timing reference for the subnetwork consists of a three-level hierarchy, with the highest priority choice given first:

a) External timing; then
b) Node 125 µs timing reference at the node with the SG_D function; then
c) Node 125 µs timing reference at the node that is also providing the Head of Bus A function.

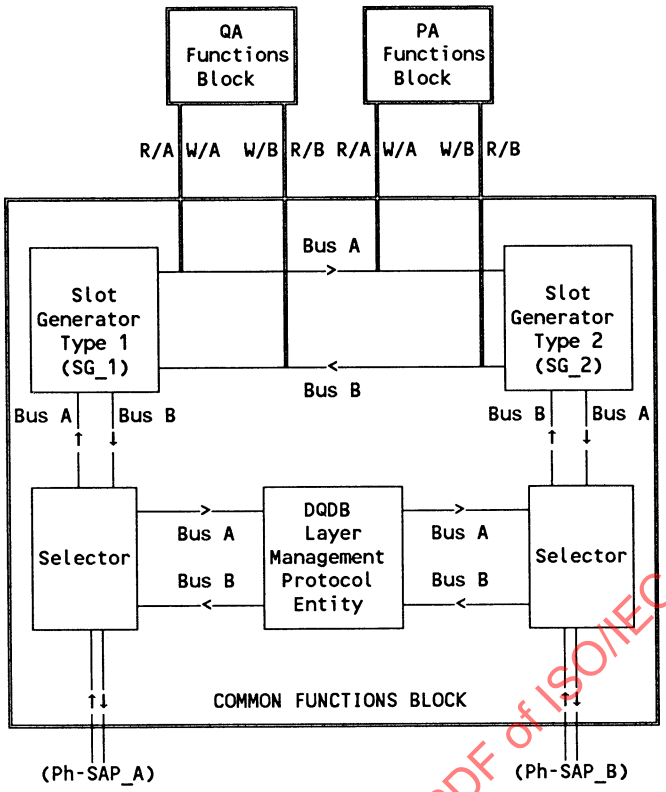


Figure 5-31—Slot generator designations for a node not supporting the Default Slot Generator function

NOTE—Multiple nodes can simultaneously provide external timing, provided the timing references are traceable to a single external timing source. The mechanism by which this is guaranteed is outside the scope of this part of ISO/IEC 8802.

5.4.2.4.2 Primary subnetwork timing reference functions for node not at head of bus

If a node is providing the primary timing reference function for the subnetwork and is not providing the Head of Bus function for either bus, then it shall provide 125 μs timing for one bus, and may provide 125 μs timing for both buses. If a node is providing the primary timing reference function for the subnetwork, and is neither providing the Head of Bus function for either bus nor contains the active Default Configuration Control function and, further, is providing 125 μs timing for one bus only in a looped subnetwork, then the timing shall be provided for Bus B. The node shall maintain data path continuity on each bus for which timing is being provided.

5.4.2.4.3 Primary subnetwork timing reference functions for node at head of bus

If a node is providing the primary timing reference function for the subnetwork and is also providing the Head of Bus x (x = A or B) function, then it shall provide 125 μs timing for Bus x.

5.4.2.4.4 Other timing reference functions for node at head of bus

If a node is providing the Head of Bus x function in a stable subnetwork, but is not providing the primary timing reference function for the subnetwork, then it shall generate slots for Bus x using the 125 µs timing it receives from the end of the other bus, Bus y (y = B or A).

If a node is undergoing Configuration Control arbitration to activate or deactivate the Head of Bus x function and is not providing the primary subnetwork timing reference function, then, during the arbitration period, the node shall use the node 125 µs timing reference for generation of slots for Bus x.

5.4.2.5 Nodes that cannot support Head of Bus functions

This part of ISO/IEC 8802 does not require all nodes to support the Head of Bus function. When a node that is not capable of supporting Head of Bus functions detects an incoming transmission link failure on Bus x (x = A or B), the node shall signal the detection of the link failure to nodes downstream on Bus x. (See 11.5.4.)

— This function aids in isolation of the transmission link fault by ensuring detection of apparent link failures due to the lack of an upstream head of bus capability.

5.4.2.6 Common Functions block support for the fundamental subnetwork requirements

The resources required to achieve the three fundamental requirements, FR1, FR2, and FR3, are managed by the Subnetwork Configuration Control function. As this function manages DQDB Layer objects, it is part of the DQDB Layer Management Entity. However, the Configuration Control functions at all nodes must communicate to coordinate their operation. This is done using the Configuration Control protocol, defined in 10.2.

The information required to operate the Configuration Control protocol is carried in the Subnetwork Configuration Field (SNCF), which is one of the fields of the DQDB Layer Management Information octets carried between nodes. (See 10.1.2.) The SNCF consists of three subfields: the Default Slot Generator Subfield (DSGS), the Head of Bus Subfield (HOBS), and the External Timing Source Subfield (ETSS).

In general, an SNCF subfield may be modified by the DQDB Layer Management Protocol Entity at any node where the Configuration Control functions have activated the associated resource (i.e., DSG, HOB, or ETS). At nodes where the Configuration Control functions have not activated the associated resource, the SNCF subfield value is relayed unchanged by the DQDB Layer Management Protocol Entity. The following clauses provide specific details of the individual SNCF subfield operations.

5.4.2.6.1 Default slot generator signalling

The valid values for the DSGS are PRESENT and NOT_PRESENT. (See 10.1.2.1.) If any of the invalid values for the DSGS are received at a node that does not contain the Default Slot Generator function, the DSGS shall be relayed unchanged by the DQDB Layer Management Protocol Entity and the value of DSGS shall be ignored. The default value of the DSGS is NOT_PRESENT. The DSGS shall be set on both buses to PRESENT by the DQDB Layer Management Protocol Entity at the node that contains the active Default Slot Generator function.

5.4.2.6.2 Head of bus signalling

The valid values for the HOBS are WAITING, STABLE, and NO_ACTIVE_HOB. (See 10.1.2.2.) If the invalid value (11) for the HOBS is received at a node, the HOBS shall be relayed unchanged by the DQDB Layer Management Protocol Entity and the value of HOBS shall be ignored.

The HOBS shall be set to STABLE on both Bus A and Bus B by the DQDB Layer Management Protocol Entity at the node that contains the Default Slot Generator (SG_D) function, if the SG_D is acting as the Head of Bus A and as the Head of Bus B in a looped dual bus subnetwork. The HOBS shall be set to STABLE on Bus A by the DQDB Layer Management Protocol Entity at the node that contains the SG_D function, if the SG_D is acting as the Head of Bus A only.

The HOBS shall be set to NO_ACTIVE_HOB on both Bus A and Bus B by the DQDB Layer Management Protocol Entity at a node that contains either a Slot Generator Type 1 (SG_1) function or Slot Generator Type 2 (SG_2) function during early phases of initialization at power-up.

The HOBS shall be set to WAITING on Bus A by the DQDB Layer Management Protocol Entity at a node that contains a Slot Generator Type 1 (SG_1) function, which is waiting to determine whether it will activate or deactivate the Head of Bus A function (i.e., Timer_H_1 is running, 7.1.2). When the Head of Bus A function is active at an SG_1 function without Timer_H_1 running, the DQDB Layer Management Protocol Entity shall set HOBS to STABLE on Bus A.

The HOBS shall be set to WAITING on Bus B by the DQDB Layer Management Protocol Entity at any node that contains a Slot Generator Type 2 (SG_2) function, which is waiting to determine whether it will activate or deactivate the Head of Bus B function (i.e., Timer_H_2 is running, 7.1.2). When the Head of Bus B function is active at an SG_2 function without Timer_H_2 running, the DQDB Layer Management Protocol Entity shall set HOBS to STABLE on Bus B.

5.4.2.6.3 External timing source signalling

The valid values for the External Timing Source Subfield (ETSS) are PRESENT and NOT_ PRESENT. (See 10.1.2.3.) The default value of the ETSS is NOT_PRESENT. The ETSS shall be set on both buses to PRESENT by the DQDB Layer Management Protocol Entity at each node which is providing the External Timing Source function.

5.4.3 Head of Bus functions

The Head of Bus function is activated and deactivated at a node under control of the Configuration Control function, subject to the requirements of 5.4.2. The Head of Bus function is provided by a combination of the Slot Marking function and parts of the DQDB Layer Management Protocol Entity function of the Common Functions block.

The Slot Marking function at an active Head of Bus function is the process of marking PA slots to be written at the head of bus, and is described in 5.4.3.1. The functional data path for the Common Functions block of a node is described in 5.4.3.2. The DQDB Layer Management Protocol Entity function at the head of bus includes the Bus Identification functions, described in 5.4.3.3, and the MID Page Allocation functions at the head of bus, described in 5.4.4.1.

5.4.3.1 Slot Marking function

The formats for Queued Arbitrated (QA) and Pre-Arbitrated (PA) slots are given in 6.3 and 6.4, respectively.

If a node is to support the Head of Bus functions, then it needs to support the managed objects and operations described in 9.2.6 and 9.2.7, which inform the node of the requirements for generating PA slots. The Slot Marking Generator function at an active Head of Bus function shall generate all slots as QA slots unless it has been directed to generate PA slots by DQDB Layer Management PA_VCI_ADD_HOB and PA_VCI_DELETE_HOB actions. (See 9.2.6 and 9.2.7.)

Empty QA slots are generated by the Slot Marking function at the active Head of Bus function with all bits in the SLOT_START octet and SLOT_DATA octets of the QA slot set to 0. The BUSY bit and SL_TYPE bit in the SLOT_START octet are both set to 0 to indicate an empty QA slot. (See 6.2.1.)

When the Slot Marking function at the active Head of Bus function determines that it needs to generate a PA slot for that bus to meet with the requirements invoked by the PA_VCI_ADD_HOB and PA_VCI_DELETE_HOB actions, then it sets the BUSY bit and SL_TYPE bit of the next SLOT_START octet to 1 to indicate a PA slot. (See 6.2.1.) It then performs the following actions on the first four SLOT_DATA octets of that slot:

a) Code the VCI value required into the VCI field into the PA segment header, according to 6.4.1.1.1.
b) Code the Payload_Type field in the PA segment header with binary 00, according to 6.4.1.1.2.
c) Code the Segment_Priority field in the PA segment header with binary 00, according to 6.4.1.1.3.
d) Calculate and code the Segment Header Check Sequence of the PA segment header, according to 6.4.1.1.4.

The Slot Marking function then sets all bits of the subsequent 48 SLOT_DATA octets to zero.

5.4.3.2 Functional data path for nodes

There is a relationship between the generation of Ph-DATA indication primitives at each Physical Layer service access point (Ph-SAP) and the receipt of Ph-DATA request primitives at both Ph-SAPs of the node. The relationship depends on whether or not the node is performing the function of head of a bus, and is described in 4.3. The Common Functions block provides the DQDB Layer function of relaying the slot octets and management information octets received in Ph-DATA indication primitives between the two Ph-SAPs in all cases described in 4.3.

Figure 5-32 gives the key for interpreting each of the diagrams in the series of figures 5-33 to 5-36.

KEY:
θ = start of data flow
▮ = end of data flow
▬ = timing and octet TYPE information relayed through node
EMPTY = All bits set to zero
HOB_A = Head of Bus A function
HOB_B = Head of Bus B function
HOB_AB = Head of Bus A and Head of Bus B function
SG_z = Slot Generator Type z function (z = 1, 2, D)

Figure 5-32—Key to functional data path diagrams

Figure 5-33 presents a model for describing the relay function for a node that is not performing Head of Bus functions, based on figure 5-24. For such a node, the Slot Generator function is a null function, and the DQDB Layer Management Protocol Entity performs the node MID Page Allocation functions defined in 5.4.4.2. All nodes conforming to this part of ISO/IEC 8802 shall support these capabilities.

Figures 5-34, 5-35, 5-36a), and 5-36b) present additions to the model of figure 5-33 for describing the relay function for nodes that are performing Head of Bus A, Head of Bus B, Head of Bus A and Bus B functions, and End of Bus A and Bus B functions, respectively. These figures provide the details of the DQDB Layer functions that use the facilities of the Physical Layer service defined in 4.3.2.1, 4.3.2.2, and 4.3.2.3, respectively.

The functions of a Slot Marking function at an active Head of Bus function are described in 5.4.3.1. The functions of the DQDB Layer Management Protocol Entity at a node with an active Head of Bus function are described in 5.4.3.3 and 5.4.4.1, for the BIF and MID Page Allocation Field, respectively.

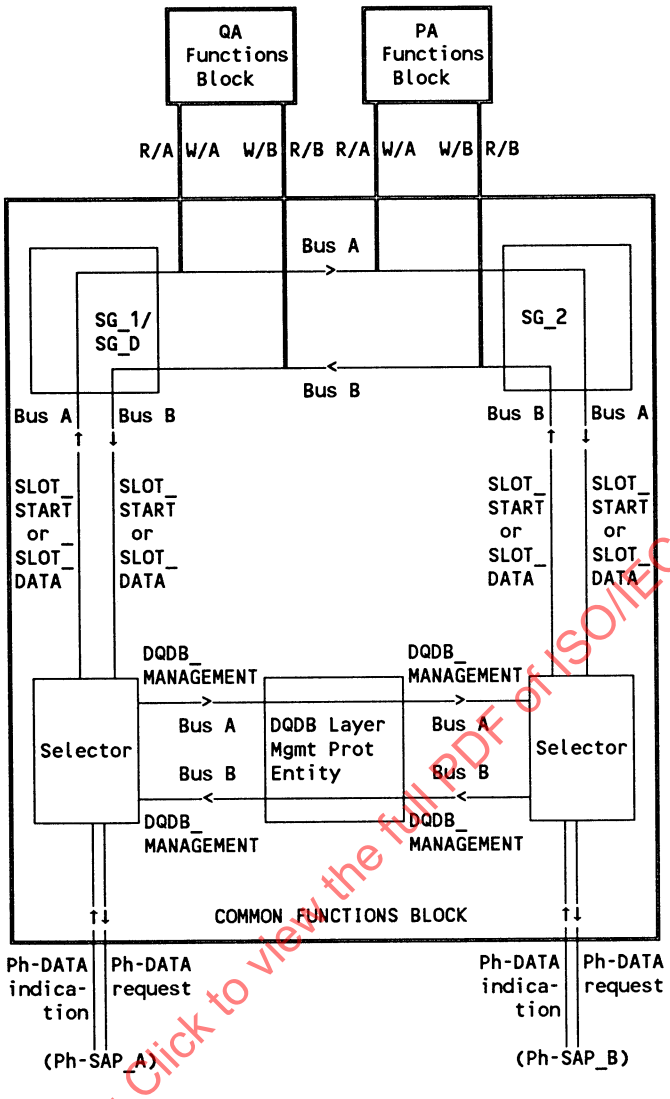


Figure 5-33—Functional data path for node not performing Head of Bus function

5.4.3.3 Bus Identification functions

The identity of each bus shall be signaled to all nodes by the DQDB Layer Management Protocol Entity at the node containing the active Head of Bus function for that bus. This is done using the Bus Identification Field (BIF), which is one of the fields of the DQDB Layer Management Information octets carried between nodes. (See 10.1.1.)

Where a node with a Default Slot Generator (SG_D) function is present in a subnetwork, the BIF for both Bus A and Bus B shall be generated by the DQDB Layer Management Protocol Entity at that node. If the function of either head of bus is active at any node without the SG_D function, then the DQDB Layer Management Protocol Entity function at that node shall generate the opposite of the BIF received at the end of bus.[30] If a node without the SG_D function has a Head of Bus function active and receives a value of the BIF that is neither Bus A nor Bus B, the DQDB Layer Management Protocol Entity at the node shall gener-

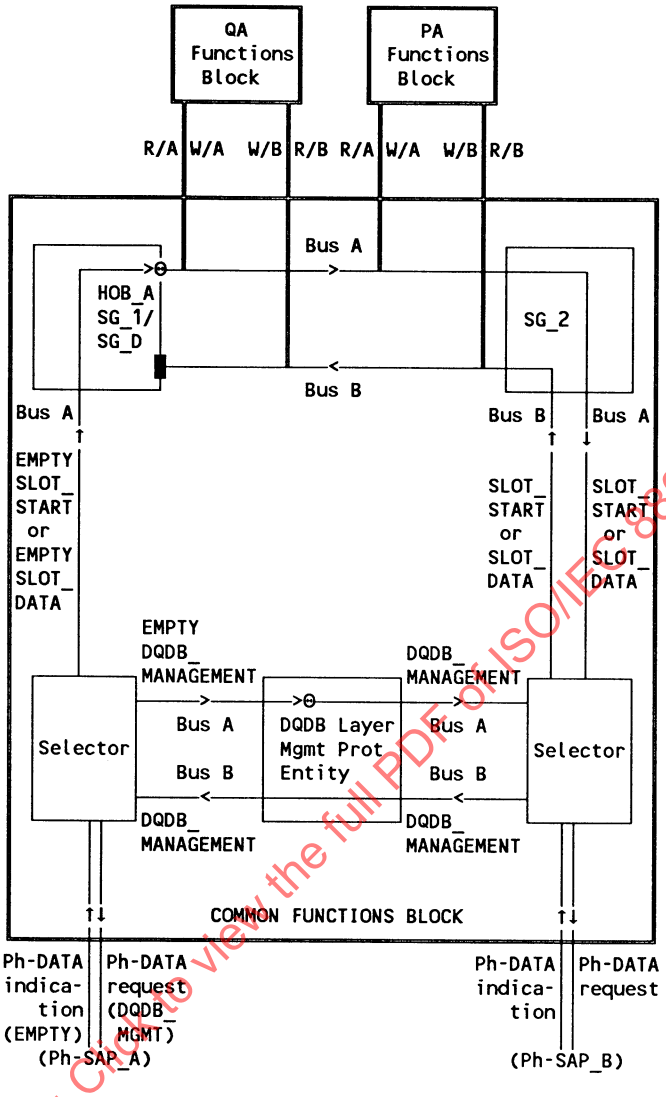


Figure 5-34—Functional data path for node performing Head of Bus A function

ate the opposite of the last Bus A or Bus B BIF received at the end of the bus while it is not receiving Bus A or Bus B BIF values.

In the case of an island[31] subnetwork formed from a subnetwork with an SG_D function, the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function shall generate the code for Bus A in the BIF. The DQDB Layer Management Protocol Entity at the node with the active Head of Bus B function shall generate the opposite of the BIF received at the end of Bus A. If the node with the active Head of Bus B receives a value of the BIF that is neither Bus A nor Bus B, the DQDB Layer Management Protocol Entity at the node shall ignore the BIF value and continue to generate the Bus B BIF value while it is not receiving Bus A or Bus B BIF values.

[30] This is the reason that all subnetworks must contain an SG_D function at subnetwork startup.

[31] An operating part of a DQDB subnetwork that is isolated from the node containing the SG_D function.

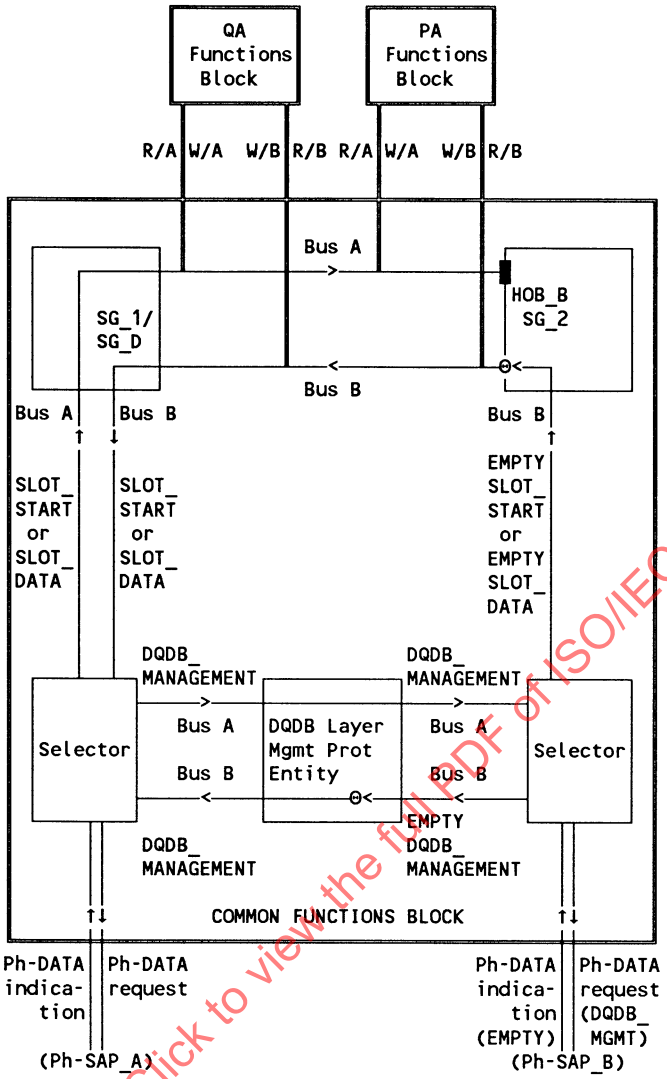
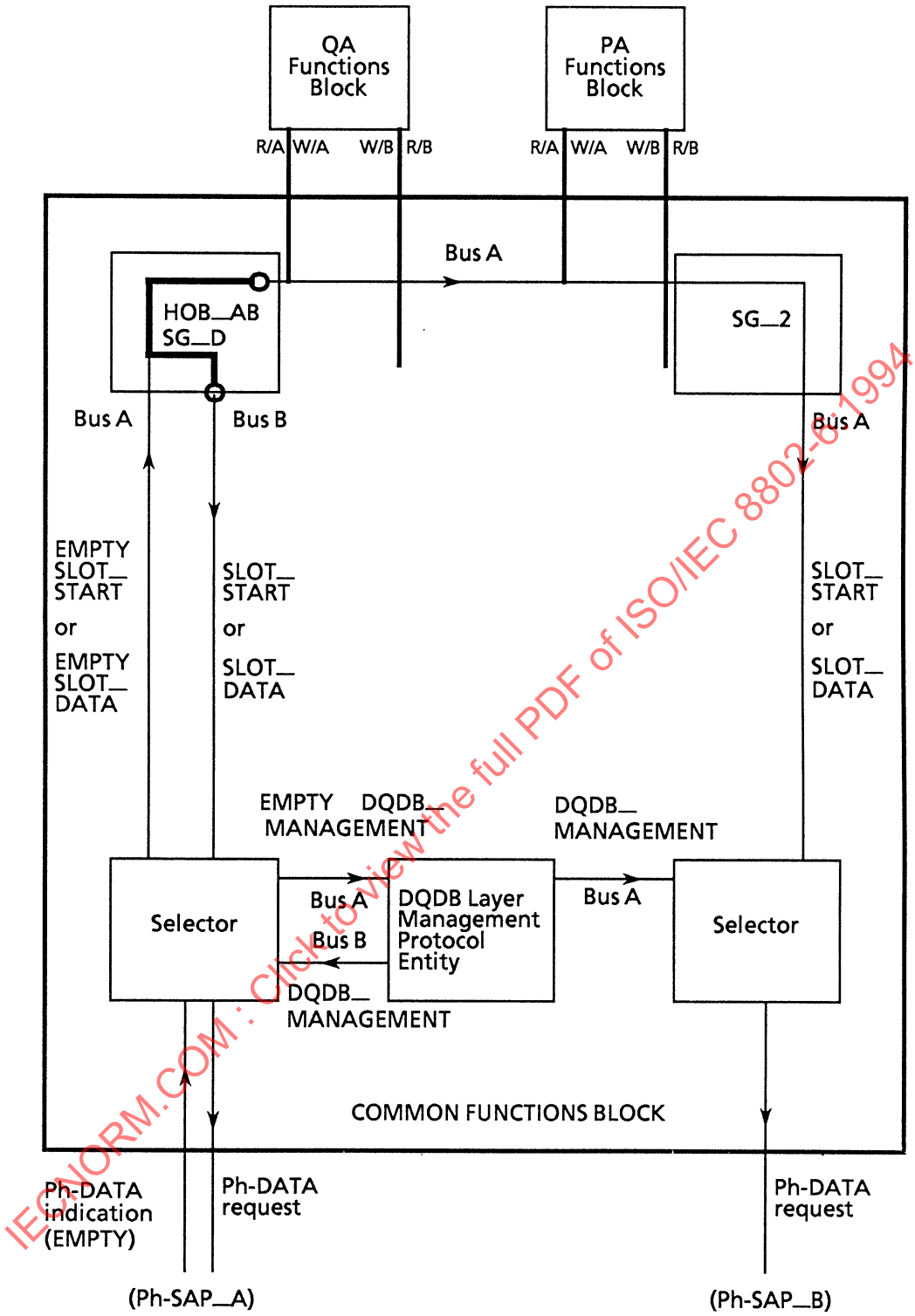


Figure 5-35—Functional data path for node performing Head of Bus B function

At nodes without the SG_D function and not containing an active Head of Bus function, the DQDB Layer Management Protocol Entity shall always relay the BIF value unchanged, irrespective of the value it contains.

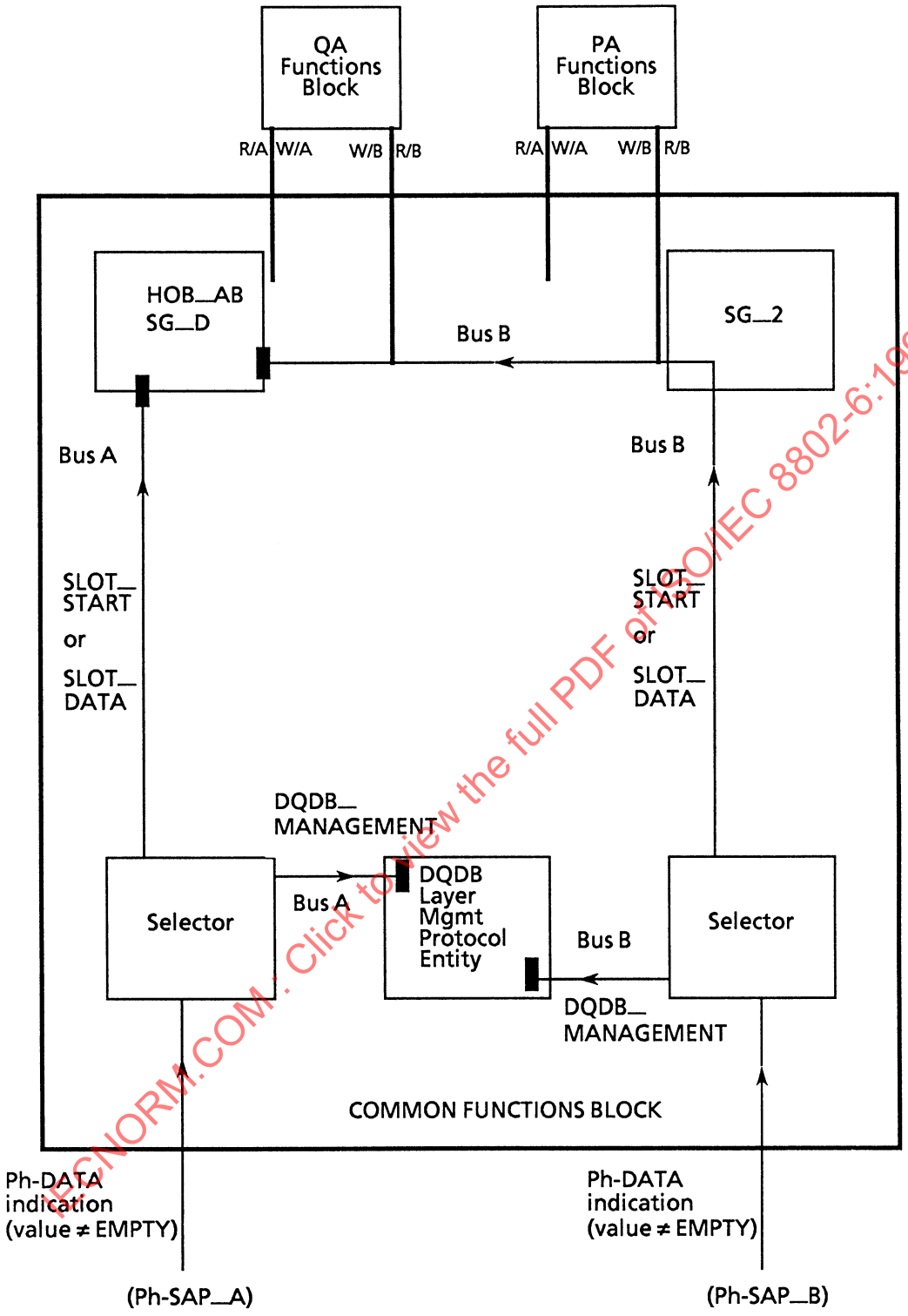
5.4.4 MID Page Allocation functions

The MID Page Allocation function controls the allocation of Message Identifiers (MIDs) to nodes along the dual bus. The unit of allocation of MIDs is an MID page of one MID value. The MID values are used for the transfer of IMPDUs that generate multiple segmentation units, as described in 5.1.1.1.3.2. The MID Page Allocation function supports each node gaining unique allocation of one or more MID page values, to ensure that the reassembly process described in 5.1.1.2.3 can operate correctly.



a) Head of Bus functions

Figure 5-36—Functional data path for node performing Head of Bus A and Head of Bus B Functions



b) End of Bus functions

Figure 5-36—Functional data path for node performing Head of Bus A and Head of Bus B functions

As the MID Page Allocation function manages DQDB Layer objects, it is part of the DQDB Layer Management Entity. However, the MID Page Allocation functions at all nodes must communicate to coordinate their operation. This is done using the MID Page Allocation protocol, defined in 10.3.

The information required to operate the MID Page Allocation protocol is carried in the MID Page Allocation Field (MPAF), which is one of the fields of the DQDB Layer Management Information octets carried between nodes. (See 10.1.3.)

The MID Page Allocation protocol operates by logically associating an MID page value with each MPAF and allowing nodes to deterministically arbitrate for allocation of this MID page value. The MPAF consists of three subfields: the Page Reservation (PR) subfield, the Page Counter Modulus (PCM) subfield, and the Page Counter Control (PCC) subfield. The PCC subfield is used to control the MID page value that will be logically associated with the PR subfield in the next MPAF received at the node. The PR subfield is used to indicate whether the associated MID page value is reserved or not reserved for use by a node. The PCM subfield is used as an additional check that all nodes are associating the correct MID page value with each MPAF.

The MID Page Allocation protocol is a two-pass protocol, with MID page values being obtained on Bus B and then maintained on Bus A. Therefore, the protocol operates differently on the two buses.

The MPAF values are generated by the DQDB Layer Management Protocol Entity at the node that contains the active Head of Bus A function. The MPAF passes along Bus A and is operated on by the DQDB Layer Management Protocol Entity of all nodes on the subnetwork to maintain knowledge of the current MID page value and, if required, to keep the MID page value, according to the MID Page Allocation protocol defined in 10.3.

The MPAF values received at the end of Bus A are echoed unchanged onto Bus B by the DQDB Layer Management Protocol Entity that contains the active Head of Bus B function. The MPAF then passes along Bus B and is operated on again by the DQDB Layer Management Protocol Entity of every node on the subnetwork to maintain knowledge of the current MID page value and, if required, to get the MID page value, according to the MID Page Allocation protocol.

The MPAF values received at the end of Bus B are discarded by the DQDB Layer Management Protocol Entity that contains the active Head of Bus A function.

Subclause 5.4.4.1 describes the functions required of the node with the active Head of Bus A function and the node with the active Head of Bus B function. Subclause 5.4.4.2 describes the functions performed at every node as a result of receiving each MPAF.

5.4.4.1 Head of Bus functions

In order to allocate MID pages to nodes on a DQDB subnetwork, the MPAFs are logically associated with a sequential numbering scheme[32] under the control of the node with the active Head of Bus A function. While the MID Page Allocation function is enabled for operation, the number associated with an MPAF is always in the range between MIN_PAGE = 1 and MAX_PAGE = $(2^{10} - 1)$. Otherwise, the number associated with an MPAF is always the reserved value of zero.

In order for the scheme to operate correctly, each node should associate the same number with a given MPAF. This number is maintained by each node using a separate page counter for each bus, PAGE_CNTR_x (x = A or B; see 7.2.4). The PAGE_CNTR_x holds the number to be associated with the PR subfield of the next MPAF to be received on that bus. This number is controlled within the range MIN_PAGE to MAX_PAGE by the PCC subfield, which is generated by the DQDB Layer Management Protocol Entity at

[32] The numbers are logically associated with the MPAF because they are not physically carried in the MPAF.

the node with the active Head of Bus A function. The value of the PCM subfield received in an MPAF on Bus x is used to check that the value of the PAGE_CNTR_x at the node is synchronized to the value associated with that MPAF by the Head of Bus A function.

5.4.4.1.1 Page Counter Control (PCC) subfield

The valid values for the PCC subfield are INCREMENT or RESET. (See 10.1.3.3.) The PCC subfield shall normally be generated by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function with a value of INCREMENT. Whenever the number that would be associated with the next MPAF is greater than MAX_PAGE = $(2^{10} - 1)$, the PCC shall be generated as RESET by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function.

The MID Page Allocation function is disabled upon power-up of the node with the active Head of Bus A function. The MID Page Allocation function stays disabled while the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function is either sending on Bus A or receiving from Bus B a Head of Bus Subfield (HOBS; see 10.1.2.2) value of WAITING. Each PCC subfield is generated with an INCREMENT value if the operation of the MID Page Allocation function has been disabled.

The MID Page Allocation function is enabled when the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function is both sending a HOBS value of STABLE on Bus A and receiving a HOBS value of STABLE from Bus B. When the MID Page Allocation function is enabled after having been disabled, the first PCC subfield is generated with a RESET value. Once enabled, the MID Page Allocation function is disabled again if the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function starts to either send on Bus A or receive from Bus B a HOBS value of WAITING.

The PCC values are generated under the control of the Page Counter State Machine (PCSM) for Head of Bus A (see 10.3.1), which maintains the value of the page counter for Head of Bus A, PAGE_CNTR_HOBA (see 7.2.4).

The DQDB Layer Management Protocol Entity at the node with the active Head of Bus B function shall pass a PCC subfield value received at the end of Bus A transparently to the next PCC subfield it generates for Bus B.

It should be noted that if the MID Page Allocation function is to operate correctly, then only one PCC subfield can be set to RESET on the DQDB subnetwork at any one time.[33]

5.4.4.1.2 Page Reservation (PR) subfield

The valid values for the PR subfield are RESERVED or NOT_RESERVED. (See 10.1.3.1.) The PR subfield shall normally be generated by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function with a value of NOT_RESERVED, which allows nodes to contend for allocation of the associated MID page value. However, to allow for centralized allocation of MID pages, the node with the active Head of Bus A function may be instructed by a DQDB Layer Management operation to mark the PR subfield associated with certain MID page values to RESERVED.

This is done by modifying the value of the RESERVED_MID_PAGES system parameter (see 7.3.4) at the node with the active Head of Bus A function to a nonzero value using a DQDB Layer Management set operation. (See 9.6.1.) If a node is to support the Head of Bus functions, then it needs to support the RESERVED_MID_PAGES parameter and the associated management operation.

[33] Hence, there is a limit on the maximum length of the DQDB subnetwork, which depends on the frequency with which MPAFs are sent. However, if the MPAFs are sent once per 125 μs, the maximum length is approximately 12 500 km.

Then, if the MID page value associated with a PR subfield is less than or equal to RESERVED_MID_PAGES, the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function shall generate the PR subfield with a value of RESERVED. If the MID page value associated with a PR subfield is greater than RESERVED_MID_PAGES, the PR subfield shall be generated at the Head of Bus A with a value of NOT_RESERVED.

The PR values are generated at the Head of Bus A under the control of the Page Reservation State Machine for Head of Bus A. (See 10.3.2.)

The DQDB Layer Management Protocol Entity at the node with the active Head of Bus B function shall pass a PR subfield value received at the end of Bus A transparently to the next PR subfield it generates for Bus B.

5.4.4.1.3 Page Counter Modulus (PCM) subfield

The valid values for the PCM subfield are the binary representation of the decimal values of 0, 1, 2, or 3. (See 10.1.3.2.) The PCM subfield shall be generated by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function with the modulo 4 value of the page counter for Head of Bus A, PAGE_CNTR_HOBA, as specified in 10.3.3.

The DQDB Layer Management Protocol Entity at the node with the active Head of Bus B function shall pass transparently a PCM subfield value received at the end of Bus A to the next PCM subfield it generates for Bus B.

5.4.4.2 Node functions

Each node must maintain a list of the MID page values that are available for use by the MAC Convergence Function block at the node. The operations required to do this are described in 5.4.4.2.1.

For a node to obtain and keep MID pages, it must maintain knowledge of the MID page value associated with the next PR subfield. This is done using page counters for each bus, as described in 5.4.4.2.2.

In order to ensure that a node can keep an MID page that it has been allocated, as recorded in the node MID page list, a two pass system is used. MID page values are kept using the PR subfields received on Bus A, as described in 5.4.4.2.3. An MID page value is released simply by not performing the function required to keep it. A node obtains a new MID page value using the PR subfields received on Bus B, as described in 5.4.4.2.4.

The strategy used for obtaining and releasing MIDs is a matter for implementation decision. One possible strategy is for the node to hold a pool of MID pages. The size of this pool could be adjusted as needed, provided that the node had not exceeded its maximum allowed allocation of MID pages, as defined by the MAX_MID_PAGES parameter. (See 7.3.5.)

5.4.4.2.1 MID page list maintenance

The node maintains a list of the MID page values that are available for use by the MAC Convergence Function block. The maximum number of MID page values that a node can maintain in the list is given by the MAX_MID_PAGES system parameter. (See 7.3.5.)

The list is maintained by the following DQDB Layer Management Interface (LMI) interactions described in 9.4:

— When the Get Page State Machine supporting the MID Page Allocation protocol (see 10.3.6) indicates that the node has successfully obtained a new MID page value, as reported in an LM-ACTION

reply (MID_PAGE_GET) with SUCCESSFUL status, the MID page value obtained is added to the MID page list.

— When the node is instructed to release an MID page value that is currently in its MID page list, by an LM-ACTION invoke (MID_PAGE_RELEASE), the MID page value released is deleted from the MID page list.

— When the Keep Page State Machine supporting the MID Page Allocation protocol (see 10.3.5) indicates that a remote node has signaled that it has allocation of an MID page value in the local node's MID page list, as indicated by an LM-EVENT notify (MID_PAGE_LOST), the MID page value is deleted from the MID page list.

5.4.4.2.2 Page counter operations

Each node maintains a page counter for each Bus x, PAGE_CNTR_x, to maintain knowledge of the MID page value associated with the next MPAF received on Bus x. The value of a PAGE_CNTR_x is controlled by the Page Counter State Machine for Bus x (see 10.3.4) using the values in the PCM subfields and PCC subfields received on Bus x.

At power-up, the value of PAGE_CNTR_x is set to zero, and remains at this value until a PCC subfield value of RESET is received on Bus x. This causes the node to reset PAGE_CNTR_x to MIN_PAGE = 1.

Thereafter, during normal operation, receipt of a PCC subfield value of INCREMENT on Bus x, with the value of PAGE_CNTR_x less than MAX_PAGE = $(2^{10} - 1)$, shall cause the node to increment the value of PAGE_CNTR_x by one. Receipt of a PCC subfield value of RESET on Bus x, with the value of PAGE_CNTR_x equal to MAX_PAGE shall cause the node to reset PAGE_CNTR_x to MIN_PAGE = 1.

Errors can occur that result in the PAGE_CNTR_x at the node temporarily losing synchronism with the MID page value associated with the MPAF. These conditions are as follows:

a) Receipt of a PCC subfield value of RESET on Bus x, with the value of PAGE_CNTR_x less than MAX_PAGE.
b) Receipt of a PCC subfield value of INCREMENT on Bus x, with the value of PAGE_CNTR_x equal to MAX_PAGE.
c) Receipt of a PCM subfield on Bus x, which contains a value that does not equal the modulo 4 value of the PAGE_CNTR_x.

In all of these cases operation of the Page Counter State Machine for Bus x is suspended until synchronism can be established again via receipt of a PCC subfield value of RESET. This causes the node to reset PAGE_CNTR_x to MIN_PAGE = 1, and recommence operation of the state machine. While the operation of PCSM_x is suspended, a node may continue to use any MID page value in its MID page list for transmission of IMPDUs. This is safe because the probability of another node being allocated the same MID page is very low and, in this event, the probability of an errored IMPDU being reassembled without detection of the error is negligible.

5.4.4.2.3 Keeping MID pages

Page Reservation (PR) subfields set to NOT_RESERVED originate from the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function and from there pass down Bus A. The MID page value associated with a PR subfield is indicated by the current value of the PAGE_CNTR_A at the node when the PR subfield is received.

If the MID page value associated with a PR subfield is in the node MID page list, the DQDB Layer Management Protocol Entity shall change the value of the PR subfield from NOT_RESERVED to RESERVED when it is received at the node. If the value of the PR subfield is already set to RESERVED when received, then another node has claimed the MID page value, and the node shall delete the MID page value from its

own MID page list. The MID Page Allocation protocol operation of keeping an MID page value currently in the node MID page list is controlled by the Keep Page State Machine at the node, as described in 10.3.5.

5.4.4.2.4 Getting MID pages

The DQDB Layer Management Protocol Entity at the node with the active Head of Bus B function shall pass a PR subfield value received at the end of Bus A transparently to the next PR subfield it generates for Bus B. The MID page value associated with a PR subfield is indicated by the current value of the PAGE_CNTR_B at the node when the PR subfield is received.

If the PR subfield is set to NOT_RESERVED when it is written to the head of Bus B, then the MID page value associated with the PR subfield may be claimed for use by a node. The DQDB Layer Management Protocol Entity of a node that wishes to obtain an MID page value does so by changing the value of a NOT_RESERVED PR subfield on Bus B to the value of RESERVED. The MID page value associated with the PR subfield that is changed in this manner is added to the node MID page list. The operation of getting an MID page value is controlled by the Get Page State Machine, as described in 10.3.6.

If the PR subfield is set to RESERVED when it reaches a node on Bus B, then the MID page value associated with the PR subfield is in use and may not be claimed.

6. DQDB Layer Protocol Data Unit (PDU) formats

6.1 Ordering principles

Each DQDB Layer PDU in this clause is described as a sequence of fields. Each figure depicts the fields in the order in which they are transferred within the DQDB Layer, with the left-most bit of the left-most field transferred first. This rule is applied to all fields represented in this part of ISO/IEC 8802.

The Physical Layer service data unit is an octet. Therefore, the sequence of fields for the PDU at each DQDB Layer entity forms an octet stream at the Physical Layer to DQDB Layer boundary. The left-most octet in each PDU is passed across the Physical Layer to DQDB Layer boundary first. The order of bits in each octet is as depicted in the relevant figure or table in this clause.

Binary formats of all fields in this document appear with the most significant bit in the left-most position of the field, except for the 16- and 48-bit addresses which follow the address representation as described in ISO/IEC 10039, 12.2.1.

6.2 Slot

A slot is the basic unit of data transfer. Every slot is a PDU transferred between adjacent nodes along a uni-directional bus. Each slot contains a 1-octet Access Control Field (ACF) and a 52-octet segment, as shown in figure 6-1.

ACF	Segment
(1 octet)	(52 octets)

Figure 6-1—Slot format

6.2.1 Access Control Field (ACF)

The ACF contains the bits that control access to slots. It contains the fields shown in figure 6-2. The length of each field is shown in bits. Reserved bits are set to 0.

Busy	SL_TYPE	PSR	Reserved	Request
(1 bit)	(1 bit)	(1 bit)	(2 bits)	(3 bits)

Figure 6-2—Access control field

The BUSY bit indicates whether the slot contains information (BUSY = 1) or does not contain information (BUSY = 0).

The SL_TYPE bit indicates whether the slot is a Queued Arbitrated (QA) slot (SL_TYPE = 0) or a Pre-Arbitrated (PA) slot (SL_TYPE = 1).

The combinations of BUSY and SL_TYPE are shown in table 6-1.

Table 6-1—Slot access control field codings

BUSY	SL_TYPE	Slot state
0	0	Empty QA slot
0	1	Reserved
1	0	Busy QA slot
1	1	PA slot

The PSR bit indicates whether the segment in the previous slot may be cleared (PSR = 1) or may not be cleared (PSR = 0).[34] The PSR bit shall be set or not set, as described in 5.1.2.2.2.

The REQUEST field contains three REQ bits, as shown in figure 6-3. The REQ bits are used in the operation of the three priority level distributed queue access mechanism.

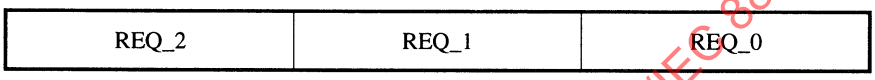


Figure 6-3—REQUEST field

Each of the REQ_I bits (I = 2,1,0) shall be set to zero by the Slot Marking function at the head of the bus and may be set to one by access units along the bus according to the rules for operation of the distributed queue. (See 8.1.2.) Each bit is used on Bus x (x = A or B) to request access to a Queued Arbitrated (QA) slot on Bus y (y = B or A, respectively). Requests for access to a QA slot are made at one of three priority levels. Priority queue level 2 is the highest priority queue and REQ_2 is the left-most bit of the REQUEST field. The priority level of the requested queue REQ_I, decreases with I (I = 2,1,0).

6.3 Queued Arbitrated (QA) slot

A QA slot is used to transfer a QA segment.

All bits of the QA slot shall be set to 0 by the Slot Marking function at the head of the bus. The BUSY bit of the ACF set to 0 and the SL_TYPE bit of the ACF set to 0 indicate an empty QA slot.

Access units access QA slots according to the rules for operation of the distributed queue. (See 8.1.) When an access unit gains access to a QA slot to transfer a QA segment, it shall mark the QA slot by setting the BUSY bit to 1, and shall write into the QA Segment field. Otherwise, the BUSY bit should remain unchanged.

6.3.1 QA segment

A QA segment is the PDU transferred in a QA slot. Each QA segment contains a header of 4 octets and a payload of 48 octets, as shown in figure 6-4.

[34] The exact operation of segment clearing is for further study.

QA segment header	QA segment payload
(4 octets)	(48 octets)

Figure 6-4—QA segment format

6.3.1.1 QA Segment Header fields

The QA segment header contains the fields shown in figure 6-5. The length of each field is shown in bits. These fields shall be inserted by the access unit that writes the QA segment into an empty QA slot.

VCI	Payload type	Segment priority	HCS
(20 bits)	(2 bits)	(2 bits)	(8 bits)

Figure 6-5—QA segment header fields

6.3.1.1.1 Virtual Channel Identifier (VCI)

The 20-bit VCI provides a means to identify the virtual channel to which the QA segment belongs. There is a single VCI space that is shared by all services.

The VCI value corresponding to all bits being set to zero is not available to refer to an active virtual channel.

6.3.1.1.1.1 Default connectionless VCI

The VCI value corresponding to all bits being set to one is the default value for the connectionless MAC service provided by the MAC Convergence function. All nodes shall be able to use the default connectionless VCI for both transmission and reception of QA segments.

An implementation may optionally support the use of additional connectionless VCI values. In such an implementation, the VCIs shall be constrained such that the range is expressed by the right-most bits, with all the left-most bits set to one.

6.3.1.1.2 Payload_Type

The 2-bit Payload_Type field indicates the nature of the data to be transferred. User data is indicated by the value Payload_Type = 00. All other values are reserved for future use.[35]

The Payload_Type field shall be set to the value 00 in all QA segments that transfer QA segment payloads for the MAC Convergence function using the default connectionless VCI (i.e., VCI = all ones).

The use of the Payload_Type field in QA segments with any other value of VCI is under study. The default value of Payload_Type to be used in QA segments with other values of VCI is 00.

[35] The Payload_Type field is intended for use in DQDB subnetworks interconnected by a Multiport Bridge. An example of its use would be to differentiate user and network information, thus allowing in-band control messages in the latter case. Another example would be to use this field for circuit tracing.

6.3.1.1.3 Segment_Priority

The 2-bit Segment_Priority field is reserved for future use with Multiport Bridging.[36] The field shall be set to the value 00.

6.3.1.1.4 Segment Header Check Sequence (HCS)

The 8-bit HCS field provides for detection of errors and correction of single-bit errors in the QA Segment Header. The HCS contains an 8-bit cyclic redundancy check (CRC) calculated on the QA Segment Header field, using the following standard generator polynomial of degree eight:

$$G(x) = x^8 + x^2 + x + 1$$

The HCS shall be encoded by the originator of the QA segment header. The contents (treated as a polynomial) of the QA segment header are multiplied by x^8 and then divided (modulo 2) by $G(x)$ to produce a remainder. This remainder is the HCS, where the coefficient of the highest term is the left-most bit.

Error detection using the HCS is mandatory at a receiver, whereas error correction using the HCS is optional. If a node supports error correction using the HCS, the procedure defined in 8.3 shall be used.

As a typical implementation, at the source node, the initial remainder of the division is preset to all zeros and is then modified by division of the QA segment header (excluding the HCS field) by the generator polynomial $G(x)$. The resulting remainder is inserted into the HCS field, with the most significant bit inserted first.

At all destination node(s), the initial remainder is preset to all zeros. The serial incoming bits of the received QA segment header (including the HCS bits), when divided by the generator polynomial $G(x)$, results, in the absence of transmission errors, in a remainder of all zeros.

6.3.1.2 QA segment payload

The QA segment payload is 48 octets long. The contents of the QA segment payload are not constrained in any way.

6.4 Pre-Arbitrated (PA) slot

A PA slot is used to transfer isochronous service octets.

6.4.1 PA segment

Each PA segment contains a header of 4 octets and a payload of 48 octets, as shown in figure 6-6.

PA segment header	PA segment payload
(4 octets)	(48 octets)

Figure 6-6—PA segment format

[36] An example of the use of the Segment_Priority field would be in congestion-control strategies within a segment-based Multiport Bridge.

A PA segment is carried in a PA slot. A PA slot shall be generated by the Slot Marking function at the head of the bus with the BUSY bit of the ACF set to 1, the SL_TYPE bit of the ACF set to 1, and all other bits of the ACF set to 0. The BUSY bit and SL_TYPE bit of the PA slot should remain unchanged at all times, but the REQ bits in the ACF may be operated on according to the rules of the distributed queue. The Slot Marking function at the head of the bus shall also write the PA segment header (see 6.4.1.1), which is carried by the PA slot. The Slot Marking function at the head of the bus shall set every bit in the PA segment payload to 0.

6.4.1.1 PA segment header fields

The PA segment header contains the fields shown in figure 6-7. The length of each field is show n in bits. These fields are written by the Slot Marking function at the head of the bus and should remain the same as the slot passes along the bus.

VCI	Payload type	Segment priority	HCS
(20 bits)	(2 bits)	(2 bits)	(8 bits)

Figure 6-7—PA segment header fields

6.4.1.1.1 Virtual Channel Identifier (VCI)

The 20-bit VCI provides a means to identify the virtual channel to which the PA segment belongs. There is a single VCI space that is shared by all services.

The VCI value corresponding to all bits being set to zero is not available to refer to an active virtual channel.

The VCI value with all bits set to one is reserved for use by the connectionless MAC service being supported by QA segments (see 6.3.1.1.1.1) and is not available for use with PA segments.

6.4.1.1.2 Payload_Type

The 2-bit Payload_Type field indicates the nature of the data to be transferred. User data is indicated by the value Payload_Type = 00. All other values are reserved for future use.[37]

The use of the Payload_Type field in PA segments is under study. The default value of Payload_Type to be used in PA segments is 00.

6.4.1.1.3 Segment_Priority

The 2-bit Segment_Priority field is reserved for future use with Multiport Bridging.[38] The field shall be set to the value 00.

---

[37] The Payload_Type field is intended for use in DQDB subnetworks interconnected by a Multiport Bridge. An example of its use would be to differentiate user and network information, thus allowing in-band control messages in the latter case. Another example would be to use this field for circuit tracing.

[38] An example of the use of the Segment_Priority field would be in congestion control strategies within a segment-based Multiport Bridge.

6.4.1.1.4 Segment HCS

The 8-bit HCS field provides for detection of errors and correction of single-bit errors in the PA segment header. The HCS contains an 8-bit cyclic redundancy check (CRC) calculated on the PA Segment Header field, using the following standard generator polynomial of degree eight:

$$G(x) = x^8 + x^2 + x + 1$$

The HCS shall be encoded by the Slot Marking function at the head of bus function. The contents (treated as a polynomial) of the PA segment header are multiplied by x^8 and then divided (modulo 2) by $G(x)$ to produce a remainder. This remainder is the HCS, where the coefficient of the highest term is the left-most bit.

Error detection using the HCS is mandatory at a node that supports the optional function of reading and/or writing the PA segment payload, whereas error correction using the HCS is optional at such a node. If a node supports error correction using the HCS, the procedure defined in 8.3 shall be used.

As a typical implementation, at the node with the active head of bus function, the initial remainder of the division is preset to all zeros and is then modified by division of the PA segment header (excluding the HCS field) by the generator polynomial $G(x)$. The resulting remainder is inserted into the HCS field, with the most significant bit inserted first.

At all destination node(s) that support the optional function of reading and/or writing the PA segment payload, the initial remainder is preset to all zeros. The serial incoming bits of the received PA segment header (including the HCS bits), when divided by the generator polynomial $G(x)$ results, in the absence of transmission errors, in a remainder of all zeros.

6.4.1.2 PA segment payload

The PA segment payload is 48 octets long. It consists of 48 isochronous service octets.

6.4.1.2.1 Isochronous service octet

Access to PA segment payloads may be shared among a number of Isochronous Service Users (ISUs). Isochronous service octets are transferred within the PA segment payload of a PA slot to support service to the ISUs.

6.5 Transfer of MAC Service Data Unit (MSDU)

A MSDU is transferred within an Initial MAC Protocol Data Unit (IMPDU). An IMPDU is transferred between peer MAC Convergence function protocol entities (MCFs) using Derived MAC Protocol Data Units (DMPDUs). IMPDUs and DMPDUs are described in the following subclauses.

The complete hierarchy of DQDB Layer PDUs required to support the transfer of an MSDU in an IMPDU is summarized in diagrammatic form in 6.6.

6.5.1 Initial MAC Protocol Data Unit (IMPDU)

The format of an IMPDU is shown in figure 6-8. The length of each field is shown in octets.

IMPDU header						
Common PDU header	MCP header	Header extension	INFO	PAD	CRC 32	Common PDU trailer
(4 octets)	(20 octets)	(+)	(*)	(#)	(!)	(4 octets)

(+) The length of the Header Extension is in steps of 4 octets in the range 0 to 20 octets, inclusive.
(*) The INFO field contains the MSDU. The length of the INFO field will vary between IMPDUs, but this part of ISO/IEC 8802 requires all nodes to receive MSDUs up to and including 9188 octets.*
(#) The PAD field consists of either 0, 1, 2, or 3 octets. The number of PAD octets is such that the total length of the INFO plus PAD fields is an integral multiple of 4 octets.
(!) CRC32 field may either be absent (length 0 octets) or present (length 4 octets).

Figure 6-8—IMPDU format

* The value of 9188 octets is such that, for a zero-length header extension, absent CRC32 field, and maximum-length INFO field, the length of the IMPDU is 9216 octets, which is 9 × 1024 octets. For a maximum-length header extension and CRC32 field present, the IMPDU occupies an integral number of segmentation units (210).

An IMPDU is constructed by adding Protocol Control Information (PCI) and zero to three PAD octets to the variable length MSDU. The PCI consists of a Common PDU header and an MCP header, which together form the IMPDU header, a Common PDU trailer, possibly a Header Extension, and possibly a CRC32 field. The MSDU is the INFO field of the IMPDU.

The IMPDU header contains the Common PDU header, which is carried in all DQDB Layer PDUs supporting frame based bursty data services, and the MAC Convergence Protocol (MCP) header, which is specific to the transfer of a MSDU. The Common PDU trailer is carried in all DQDB Layer PDUs supporting frame based bursty data services.

6.5.1.1 Common PDU Header fields

The Common PDU header contains the fields shown in figure 6-9. The length of each field is shown in octets.

Reserved	BEtag	BAsize
(1 octet)	(1 octet)	(2 octets)

Figure 6-9—Common PDU header format

6.5.1.1.1 Reserved field

The Reserved field shall be set by the MAC Convergence function to zero for the transfer of an IMPDU.[39]

6.5.1.1.2 Beginning-End tag (BEtag)

The value in the BEtag field is selected for each IMPDU by the MAC Convergence function, as described in 5.1.1.1.1. For a given IMPDU, the source node shall insert the same value in the BEtag field in the Common PDU header and the BEtag field in the Common PDU trailer (see 6.5.1.7.2). This is done independently of the length of the IMPDU, so that the BEtag fields in an SSM shall be sent with the same value. The BEtag allows the association of the Beginning of Message (BOM) DMPDU with the End of Message (EOM)

[39] The use of the Reserved field for other convergence functions is under study.

DMPDU derived from the same IMPDU. This association is required to detect the loss of certain DMPDUs over a sequence of IMPDUs.

6.5.1.1.3 Buffer Allocation size (BAsize)

The BAsize field shall be set by the MAC Convergence function to the length, in octets, of the MCP header, header extension, INFO field, PAD field, and CRC32 field (if present), for the transfer of an IMPDU. The source node shall insert the same value in the Length field in the Common PDU trailer of the IMPDU (see 6.5.1.7.3).[40]

This part of ISO/IEC 8802 requires all nodes to be able to receive INFO fields of length up to and including 9188 octets for PI = 1 (i.e., when the MAC service user is LLC).

6.5.1.2 MCP header fields

The MCP header contains information specific to the MAC Convergence function that is necessary to transfer an IMPDU. It contains the fields shown in figure 6-10. The length of each field is shown in octets.

DA	SA	PI/PL	QOS/CIB/HEL	BRIDGING
(8 octets)	(8 octets)	(1 octet)	(1 octet)	(2 octets)

Figure 6-10—MCP header format

6.5.1.2.1 Address fields

Each IMPDU contains two address fields: the destination address (DA) and the source address (SA). The DA and SA fields are of a fixed length of 64 bits. The address fields can support three MAC Service Access Point (MSAP) address types of length N bits, where N is 16, 48, or 60.

Support for 48-bit, universally administered MSAP addresses is mandatory. Support for 48-bit, locally administered, and 16- and 60-bit MSAP addresses is optional.

The Address field subfields are as shown in figure 6-11. The length of each subfield is shown in bits.

Address type	Padding	MSAP address
(4 bits)	(60 – N bits)	(N bits)

Figure 6-11—Address field

The ADDRESS_TYPE subfield indicates the structure of the remaining 60 bits, and is coded as shown in table 6-2.

All bits in the PADDING subfield shall be set to zero.

The representation of each MSAP Address field shall be as described below.

[40] The BAsize may be set by other convergence functions to a value that is equal to or larger than the Length field in the Common PDU trailer. This field is intended for buffer allocation purposes at the destination node.

Table 6-2—Coding of ADDRESS_TYPE subfield

ADDRESS_TYPE	MSAP address structure
0100	16 bit
1000	48 bit
1100	Individual 60 bit, publicly administered
1101	Individual 60 bit, privately administered
1110	Group 60 bit, publicly administered
1111	Group 60 bit, privately administered
All other codes	Reserved for future standardization

6.5.1.2.1.1 MSAP addresses of 16 and 48 bits

The representation of 16-bit and 48-bit MSAP addresses shall be as shown in figure 6-12. The left-most bit of both 16-bit and 48-bit addresses indicates whether the address is an individual or group address. Source addresses may not contain a group address. The bit second from the left in a 48-bit address indicates whether it is a universally or locally administered address.

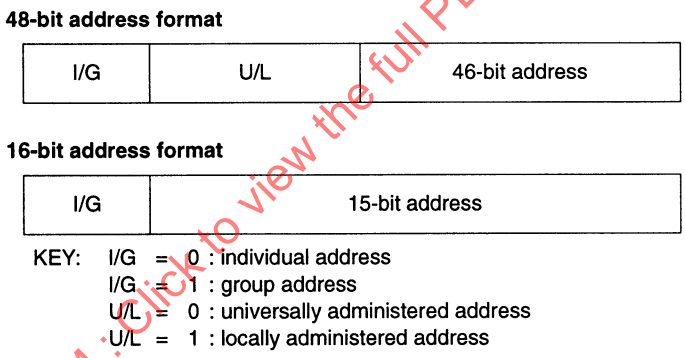


Figure 6-12—Sixteen-bit and 48-bit MSAP address format

Individual and Group Addresses. The left-most (I/G) bit of the destination address distinguishes individual from group addresses:

0 = individual address

1 = group address

Individual addresses identify a particular node on a DQDB subnetwork and shall be distinct from all other individual node addresses on the same DQDB subnetwork (in the case of local administration), or from the individual addresses of other DQDB nodes and LAN stations on a global basis (in the case of universal administration).

A group address shall be used to address an IMPDU to multiple destination MAC service access points.

Broadcast Address. The group address consisting of all 16 bits or all 48 bits being set to 1 shall constitute a broadcast address, denoting the set of all nodes on a DQDB subnetwork that employs 16- or 48-bit addressing, respectively.

Address Administration. There are two methods of administering the set of 48-bit addresses: locally or through a universal authority. The bit second from the left (U/L) indicates whether the address has been assigned by a universal or local administrator:

0 = universally administered
1 = locally administered

Universal Administration. With this method, all individual addresses are distinct from the individual addresses of all other DQDB nodes and LAN stations on a global basis.

NOTE—The administration of these addresses is performed by the Registration Authority, IEEE Standards Department, Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

The structure and format of universally administered 48-bit addresses shall be as described in ISO/IEC 10039, 12.2.1.

Local Administration. With this method, individual node addresses are locally administered. This is the only method allowed for 16-bit addresses.

6.5.1.2.1.2 MSAP addresses of 60 bits

In the case of publicly administered 60-bit addresses, the administration authority is the local metropolitan area network (MAN) operator.

When the publicly administered, individual 60-bit MSAP addresses are used to carry numbers allocated according to CCITT Recommendation E.164, Numbering Plan for the ISDN Era, then the administration of these numbers is according to clause 4 of CCITT Recommendation E.164, which states the following:

The assignment of country codes is administered by CCITT, while National Significant Number (National Destination Code plus Subscriber Number) code assignments are a national responsibility.

The E.164 number carried in the IMPDU shall always be the international significant number. The international significant number of E.164 can require up to 15 decimal digits. If the local MAN operator does not make use of the full number length (15 decimal digits), then the number assigned by the local MAN operator shall be padded on the right with bits set to one so that a full 60-bit address is obtained.

The encoding of the E.164 number shall be as follows. Each decimal digit shall be encoded using Binary Coded Decimal (BCD) into four bits, with the most significant bit occurring left-most. The most significant BCD encoded digit of the E.164 number shall occur first in the MSAP Address subfield. Therefore, the first bit of the 60-bit MSAP Address subfield is the most significant bit of the most significant E.164 digit.

6.5.1.2.2 Destination Address (DA)

The DA field identifies the node or nodes for which the INFO field of the IMPDU is intended. The DA field may take any of the formats outlined in 6.5.1.2.1.1 or 6.5.1.2.1.2.

6.5.1.2.3 Source Address (SA)

The SA field identifies the node that originated the IMPDU. The SA field shall not be coded as a group address, but otherwise shall have the same type as the DA in a given IMPDU.

6.5.1.2.4 Protocol Identification/PAD Length (PI/PL)

The PI/PL field contains the subfields shown in figure 6-13. The length of each subfield is shown in bits.

PI	PL
(6 bits)	(2 bits)

Figure 6-13—PI/PL field format

6.5.1.2.4.1 Protocol Identification (PI) field

The 6-bit PI field is used to identify the MAC service user to which the INFO field is to be sent at the node or nodes identified by the Destination Address field. Valid values in decimal form for this field are as shown in table 6-3.

Table 6-3—Coding of the PI field

PI range	Protocol entity
1	LLC
48–63	Available for use by local administration
Other values	Reserved for future standardization by IEEE 802.6

6.5.1.2.4.2 PAD Length (PL) field

The 2-bit PL field is used to indicate the length of the PAD field. (See 6.5.1.5.) The number of PAD octets is either 0, 1, 2, or 3, such that the total length of INFO field and PAD field is an integral multiple of four octets.

As a typical implementation, the number of PAD octets is derived by computing the difference between 3 and the modulo 4 result of the sum of 3 and the length of the INFO field:

Number_PAD_octets = 3 – [(Length of INFO field + 3)(mod 4)]

6.5.1.2.5 Quality of Service/CRC32 Indicator Bit/Header Extension Length (QOS/CIB/HEL)

The Quality of Service (QOS) part of this field indicates the requested quality of service for an IMPDU. The CRC32 Indicator Bit (CIB) part of this field indicates the presence or absence of the CRC32 field of the IMPDU. The Header Extension Length (HEL) part of this field indicates the length of the Header Extension field of the IMPDU. The QOS/CIB/HEL field contains the subfields shown in figure 6-14. The length of each subfield is shown in bits.

QOS_DELAY	QOS_LOSS	CIB	HEL
(3 bits)	(1 bit)	(1 bit)	(3 bits)

Figure 6-14—QOS/CIB/HEL field format

6.5.1.2.5.1 Quality of Service: Delay (QOS_DELAY)

The 3-bit QOS_DELAY subfield indicates the requested quality of service for an IMPDU with respect to delay in accessing the subnetwork. The value coded into the QOS_DELAY subfield is determined from the priority parameter received in the corresponding MA-UNITDATA request primitive.

The coding of the QOS_DELAY subfield shall be as shown in table 6-4.

Table 6-4—Coding of QOS_DELAY

Priority requested	QOS_DELAY subfield	Relative delay requested
7	1 1 1	Shortest
6	1 1 0	
5	1 0 1	
4	1 0 0	
3	0 1 1	
2	0 1 0	
1	0 0 1	
0	0 0 0	Longest

6.5.1.2.5.2 Quality of Service: Loss (QOS_LOSS)

The 1-bit QOS_LOSS subfield is currently reserved, and shall be coded as QOS_LOSS = 0.[41]

6.5.1.2.5.3 CRC32 Indicator Bit (CIB)

The CIB subfield indicates the presence or absence of the CRC32 field. (See 6.5.1.6.) A value of zero indicates that there is no CRC32 field in the IMPDU. A value of one indicates that a CRC32 field is present in the IMPDU.

6.5.1.2.5.4 Header Extension Length (HEL)

The 3-bit HEL subfield gives the length of the Header Extension field in the IMPDU. The length of the Header Extension in octets is given by multiplying the numerical value of the HEL subfield by four. The decimal value of the HEL subfield is in the range 0 to 5, inclusive. Therefore, the Header Extension field is of length 0 to 20 octets, inclusive, in steps of four octets.

A length of zero indicates that there is no Header Extension field in the IMPDU. The HEL subfield values of 6 and 7 are invalid.

---

[41] The QOS_LOSS subfield is intended for use in DQDB subnetworks interconnected by a bridge. An example of its use would be to indicate the requested quality of service for an IMPDU with respect to preferential discard due to resource congestion at a MAC Convergence function block, such as at a MAC Sublayer bridge.

6.5.1.2.6 Bridging field (BRIDGING)

The 2-octet BRIDGING field is reserved for future use with MAC Sublayer bridging.[42] All bits shall be set to zero.

6.5.1.3 Header Extension field

The Header Extension field provides the capability to convey additional IMPDU PCI that may be standardized in the future. All values for the encoding of the Header Extension are reserved for future use.[43]

All nodes shall recognize the existence of the Header Extension field, as indicated by a nonzero value of the HEL subfield. Nodes are not required to interpret the contents of the Header Extension field.

6.5.1.4 Information (INFO) field

The INFO field contains the MAC Service Data Unit (MSDU). The length of the INFO field will vary, but this part of ISO/IEC 8802 requires all nodes to receive MSDUs up to and including 9188 octets, for PI = 1. The INFO field is intended at the node or nodes identified by the Destination Address field for the MAC service user indicated by the Protocol Identification field.

6.5.1.5 PAD field

The PAD field contains sufficient octets such that the total length of INFO field plus PAD field is an integral multiple of four octets. The number of PAD octets is either 0, 1, 2, or 3, as indicated by the PAD Length (PL) field (see 6.5.1.2.4.2). Each PAD octet shall be coded as zeros.[44]

6.5.1.6 CRC32 field

The CRC32 field provides the capability for including a 32-bit Cyclic Redundancy Check (CRC), calculated over all the fields of the MCP Header (see 6.5.1.2), the Header Extension field (see 6.5.1.3), the INFO field (see 6.5.1.4), and the PAD field (see 6.5.1.5). These are referred to below as the calculation fields.

For the purposes of CRC calculation, the BRIDGING field is assumed to be all zeros.[45] Therefore, the 32-bit CRC does not provide for detection of errors in the BRIDGING field, but does provide for detection of errors in all of the other calculation fields mentioned above. These are referred to as the covered fields.

The CRC32 is calculated using the following standard generator polynomial of degree 32:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

See [B6].[46]

---

[42] The BRIDGING field may be standardized for use in carrying the maximum number of bridges that the IMPDU is allowed to pass through, and to maintain a count of the number of bridges through which the IMPDU has actually passed. This would enable the IMPDU to be discarded when a set number of bridges had been traversed.

[43] An example of the use of the Header Extension field would be to convey service-provider-specific information in cases where this protocol is used to access the services of a public network.

[44] Whether the CRC32 field is present or absent, this PAD field is considered to be part of a MAC Convergence Protocol (MCP) trailer. The (MCP) PAD ensures that the data present between the Common PDU header and Common PDU trailer is always 32-bit aligned. In addition, a PAD associated with the Common PDU trailer is defined. However, this PAD field always has a length of zero octets for the MCF and is therefore not shown in the relevant figures (e.g., figure 6-20). For other convergence functions, e.g., Connection-Oriented Convergence function, the PAD field associated with the Common PDU trailer may be non-zero length in order to achieve 32-bit alignment.

[45] It must be assumed that the BRIDGING field changes in transit from the value of all zeros generated at the transmitter.

[46] The numbers in brackets correspond to those of the bibliographical references in annex J.

The CRC32 shall be the one's complement of the sum (modulo 2) of the following:

a) The remainder of $x^k * (x^{31} + x^{30} + x^{29} + ... + x^2 + x + 1)$ divided (modulo 2) by G(x), where k is the number of bits in the calculation fields; with

b) The remainder after multiplication of the contents (treated as a polynomial) of the calculation fields by x^{32} and then division (modulo 2) by G(x).

The CRC32 shall contain the coefficient of the highest term in the left-most bit.

As a typical implementation, at the source node, the initial remainder of the division is preset to all ones and is then modified by division of the calculation fields by the generator polynomial, G(x). The one's complement of this remainder is inserted in the CRC32 field, with the most significant bit inserted first.

At the destination node or nodes, the initial remainder is preset to all ones. The division of the received calculation fields by the generator polynomial, G(x), results, in the absence of transmission errors, in a unique nonzero remainder value, which is the following polynomial:

$$x^{31} + x^{30} + x^{26} + x^{25} + x^{24} + x^{18} + x^{15} + x^{14} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x + 1$$

All nodes shall recognize the existence of the CRC32 field, as indicated by a value of one in the CIB subfield. Nodes are not required to interpret the contents of the CRC32 field.[47]

6.5.1.7 Common PDU trailer fields

The Common PDU trailer contains the fields shown in figure 6-15. The length of each field is shown in octets.

Reserved	BEtag	Length
(1 octet)	(1 octet)	(2 octets)

Figure 6-15—Common PDU trailer format

6.5.1.7.1 Reserved field

The Reserved field shall be set by the MAC Convergence function to zero for the transfer of an IMPDU.[48]

6.5.1.7.2 Beginning-End tag (BEtag)

For a given IMPDU, the source node shall insert the same value in the BEtag field in the Common PDU trailer as was inserted in the BEtag field in the Common PDU header. (See 6.5.1.1.2.) The BEtag allows the association of the End of Message (EOM) DMPDU with the Beginning of Message (BOM) DMPDU derived from the same IMPDU. This association is required to detect the loss of certain DMPDUs over a sequence of IMPDUs.

[47] The CRC32 field is provided as an optional field with the understanding that for many applications the various other error checking capabilities such as the Segment Header Check Sequence, and the Payload_CRC will be sufficient.

[48] The use of the Reserved field for other convergence functions is under study.

6.5.1.7.3 Length

The Length field shall be set by the MAC Convergence function to the length, in octets, of the MCP header, header extension, INFO field, PAD field, and CRC32 field (if present), for the transfer of an IMPDU. The source node shall insert the same value in the BAsize field in the Common PDU header of the IMPDU. (See 6.5.1.1.3.)

This part of ISO/IEC 8802 requires all nodes to be able to receive INFO fields of length up to and including 9188 octets for PI = 1 (i.e., when the MAC service user is LLC).

6.5.2 Derived MAC Protocol Data Unit (DMPDU)

The variable-length IMPDU is passed to the segmentation process in the MAC Convergence function (MCF) block, which produces one or more segmentation units that are of fixed length of 44 octets. Protocol Control Information is added to each segmentation unit to form a DMPDU. There are four types of DMPDUs, as described in 6.5.2.1.1. This process is shown in figure 6-16 and described in 5.1.1.1.2 and 5.1.1.1.3.

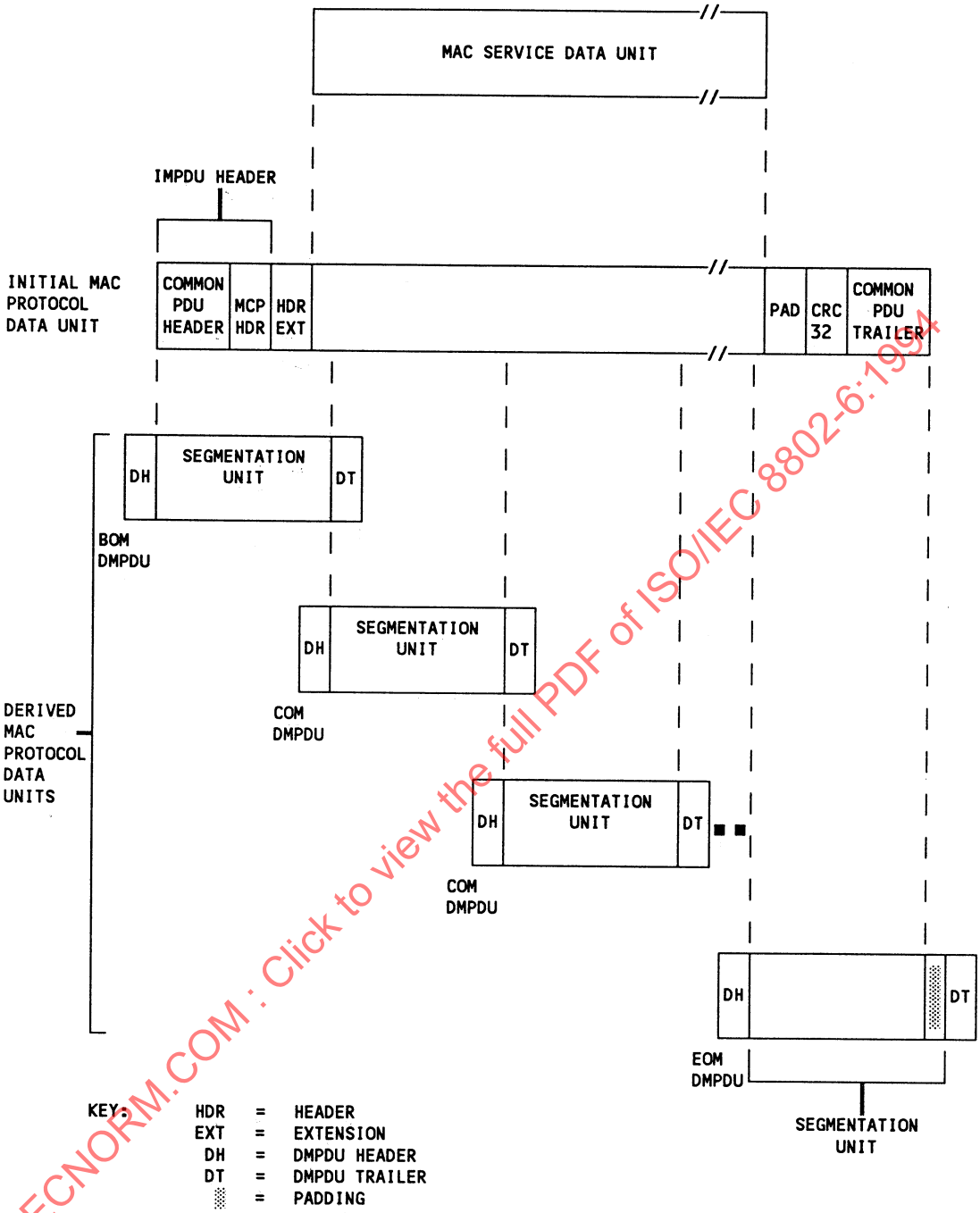


Figure 6-16—Segmentation of an IMPDU into DMPDUs

The DMPDUs derived from a single IMPDU are transferred as QA segment payloads to one or more peer MCF entities, which reassemble the segmentation units in the received DMPDUs to form the IMPDU. All of the DMPDUs derived from an IMPDU shall be transferred using the same Distributed Queue access priority level and using the same bus(es).

The DMPDU contains the fields shown in figure 6-17. The length of each field is shown in octets.

DMPDU header	Segmentation unit	DMPDU trailer
(2 octets)	(44 octets)	(2 octets)

Figure 6-17—DMPDU format

6.5.2.1 DMPDU header subfields

The DMPDU header contains information for reassembling the IMPDU. It contains the subfields shown in figure 6-18. The length of each subfield is shown in bits.

Segment_Type	Sequence_Number	MID
(2 bits)	(4 bits)	(10 bits)

Figure 6-18—DMPDU header format

6.5.2.1.1 Segment_Type subfield

The 2-bit Segment_Type subfield indicates how the segmentation unit should be processed by the receiving MCF block. The codes for the Segment_Type field are shown in table 6-5.

Table 6-5—DMPDU type

Segment_Type	DMPDU type
00	Continuation of Message (COM)
01	End of Message (EOM)
10	Beginning of Message (BOM)
11	Single Segment Message (SSM)

6.5.2.1.1.1 Beginning of Message (BOM) DMPDU

The first 24 octets of the BOM segmentation unit contain the Common PDU header and the MCP header.

The next 20 octets of the IMPDU fill the remainder of the BOM segmentation unit.

6.5.2.1.1.2 Continuation of Message (COM) DMPDU

The first octet of a COM segmentation unit is the next octet of the IMPDU to be transmitted. The COM segmentation unit contains the next 44 octets of the IMPDU.

6.5.2.1.1.3 End of Message (EOM) DMPDU

The first octet of an EOM segmentation unit is the next octet of the IMPDU to be transmitted. Depending on the length of the original IMPDU, an EOM segmentation unit contains the last four or more octets of the IMPDU, up to a maximum of 44 octets. Any trailing octets in the EOM segmentation unit that are not used to carry octets of the IMPDU are coded as zeros.

6.5.2.1.1.4 Single Segment Message (SSM) DMPDU

The first 24 octets of the SSM segmentation unit contain the Common PDU header and the MCP header.

The remaining 20 octets of the SSM segmentation unit are used to carry all of the remaining octets of the IMPDU. Any trailing octets in the SSM segmentation unit that are not used to carry octets of the IMPDU are coded as zeros.

6.5.2.1.2 Sequence_Number subfield

The 4-bit Sequence_Number subfield is used in reassembling an IMPDU to verify that all of the DMPDU segmentation units of the IMPDU have been received and concatenated in the correct sequence. The value of the Sequence_Number subfield is set in the BOM DMPDU to the current value of the TX_SEQUENCE_NUM counter (see 7.2.6) associated with both the MID value used for the BOM DMPDU and the VCI value used in the header of the segment that carries the BOM DMPDU.[49] The value of the Sequence_Number subfield is incremented by one (modulo 16) for each successive DMPDU derived from the IMPDU.

The 4-bit Sequence_Number subfield is also present in SSM DMPDUs, although it is not needed for reassembly. The value of the Sequence_Number subfield is set in the SSM DMPDU to the current value of the TX_SEQUENCE_NUM counter (see 7.2.6) associated with both the reserved MID value used for SSM DMPDUs and the VCI value used in the header of the segment that carries the SSM DMPDU.

6.5.2.1.3 Message Identifier (MID) subfield

The 10-bit MID subfield is used to reassemble the DMPDU segmentation units into an IMPDU. The source shall send the same MID in all DMPDUs derived from a given IMPDU.

The MID subfield contains one of the MID page values currently available to the node via the MID Page Allocation function described in 5.4.4.2.

The MID value represented by all 10 bits of the MID subfield being set to zero is reserved for use with SSM DMPDUs. All SSM DMPDUs shall be generated with all 10 bits of the MID subfield set to zero.

6.5.2.2 DMPDU trailer subfields

The DMPDU trailer contains information for detecting errors in the DMPDU and for identifying the fill in the payload. It contains the subfields shown in figure 6-19. The length of each subfield is shown in bits.

---

[49] The values assigned to the BOM and SSM Sequence_Number subfields are not verified at reassembly, and are only recommended values.

Payload_Length	Payload_CRC
(6 bits)	(10 bits)

Figure 6-19—DMPDU trailer format

6.5.2.2.1 Payload_Length subfield

The 6-bit Payload_Length subfield indicates the number of octets from the IMPDU that occupy the segmentation unit of the DMPDU. This number has a value that is any multiple of four between 4 and 44, inclusive. All other values are invalid.

The Payload_Length subfield shall be set by the MAC Convergence function to a value of 44 for all BOM and COM DMPDUs. The Payload_Length value shall be set by the MAC Convergence function to the appropriate value for EOM and SSM DMPDUs. For EOM DMPDUs, the value can be any multiple of 4 in the range 4 to 44, inclusive. For SSM DMPDUs, the value can be any multiple of 4 in the range 28 to 44, inclusive.

6.5.2.2.2 Payload_CRC subfield

The 10-bit Payload_CRC subfield provides for detection of errors and correction of single-bit errors in the DMPDU, including the DMPDU header, the segmentation unit, and the Payload_Length subfield of the DMPDU trailer. The Payload_CRC contains a 10-bit cyclic redundancy check (CRC) calculated over the entire contents of the DMPDU, using the following standard generator polynomial of degree ten:

$$G(x) = x^{10} + x^9 + x^5 + x^4 + x + 1$$

The Payload_CRC shall be encoded at the source node. The contents (treated as a polynomial) of the DMPDU are multiplied by x^{10} and then divided (modulo 2) by $G(x)$ to produce a remainder. This remainder is the Payload_CRC, where the coefficient of the highest term is the left-most bit.

Use of the Payload_CRC at a receiver is mandatory for error detection. Use of the Payload_CRC for error correction of the DMPDU header is optional. Use of the Payload_CRC for error correction of the segmentation unit or DMPDU Trailer is not allowed.[50]

As a typical implementation, at the source node, the initial remainder of the division is preset to all zeros and is then modified by division of the DMPDU (excluding the Payload_CRC subfield) by the generator polynomial $G(x)$. The resulting remainder is inserted into the Payload_CRC subfield, with the most significant bit inserted first.

At the destination node or nodes, the initial remainder is preset to all zeros. The serial incoming bits of the received DMPDU (including the Payload_CRC bits), when divided by the generator polynomial $G(x)$, results, in the absence of transmission errors, in a remainder of all zeros.

6.6 Protocol Data Unit (PDU) Hierarchy for MAC Service

Figure 6-20 contains a diagrammatic representation of the relationship between the entire set of DQDB Layer PDUs required to transfer a MSDU. Note that all numbers in brackets are in octets. Field lengths that are given in bits are indicated specifically.

[50] The use of the Payload_CRC for error correction of the segmentation unit is under study for other convergence functions.

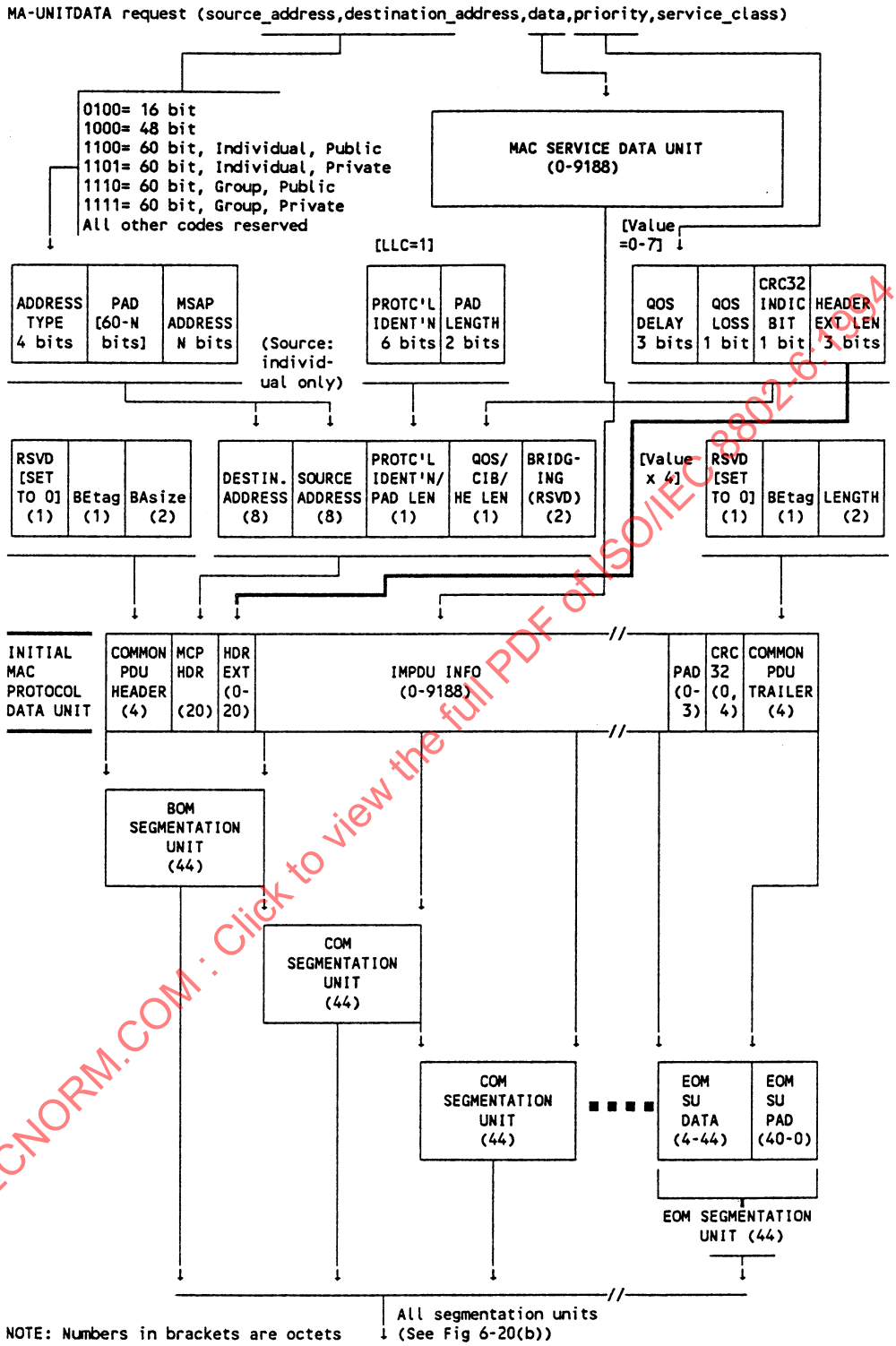


Figure 6-20—MAC connectionless service PDU hierarchy

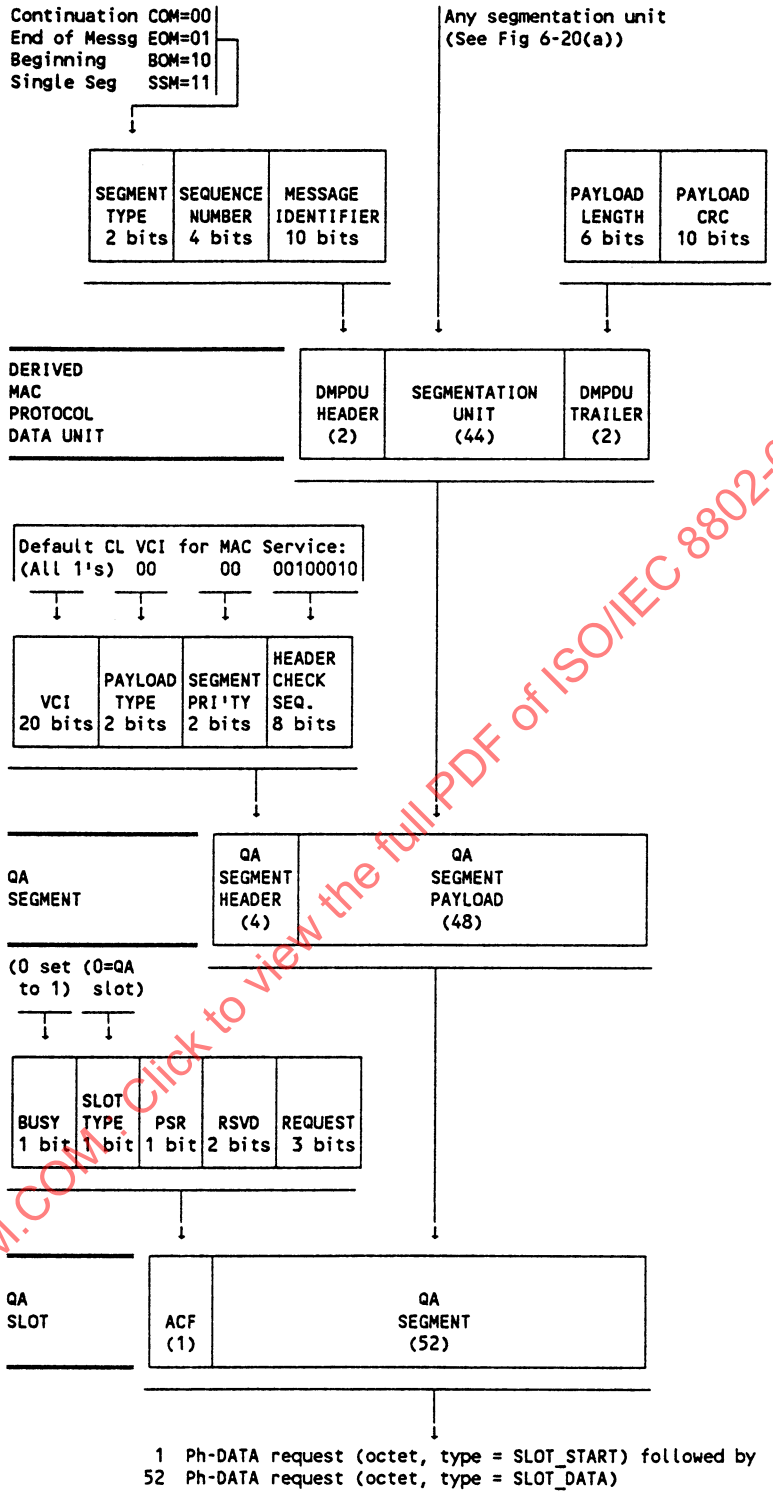


Figure 6-20—MAC connectionless service PDU hierarchy *(continued)*

6.7 Node conformance requirements

Tables 6-6 through 6-10 contain a summary of all of the Protocol Control Information (PCI) elements described in the previous subclauses of clause 6. The tables also indicate the operations that a node that supports MAC service to the LLC Sublayer shall be able to perform on each of these PCI elements. In the case of conflict between these tables and the previous subclauses, the previous subclauses of clause 6 shall take precedence.

The key to the symbols used in the Set and Interpret columns in tables 6-6 through 6-10 is as follows:

KEY: M = Mandatory; i.e., node must perform operation
O = Optional; i.e., node may perform operation
N/A= Not applicable

Table 6-6—Access control field conformance

Access Control Field (See 6.2.1)	Set	Interpret
BUSY bit	M	M
SL_TYPE bit = 1 (QA)	N/A(1)	M
SL_TYPE bit = 1 (QA)	N/A(1)	M
PSR bit	M	O
REQUEST field	M	M

(1)=M for Head of Bus function.

Table 6-7—Queued arbitrated segment conformance

Queued arbitrated segment (See 6.3.1)	Set	Interpret
Default Connectionless VCI for MAC Service (VCI = All 1's)		
VCI (All 20 bits set to 1)	M	M
Payload_Type (Set to 00)	M	N/A
Segment_Priority (Set to 00)	M	N/A
HCS: Setting at Source	M	N/A
HCS: Detection at Destination	N/A	M
HCS: Correction at Destination	N/A	O
QA Segment Payload	M	M
Other VCI excluding VCI with all 20 bits set to zero	O(1)	O(1)*

(1) = If an implementation is capable of recognizing connectionless VCIs additional to the default value, then the VCIs shall be constrained such that the range is expressed by the right-most bits, with all the left-most bits set to one.

* It is strongly recommended that manufacturers support the ability to recognize more than one value of VCI from the entire range.

Table 6-8—Pre-Arbitrated segment conformance

Pre-Arbitrated Segment (See 6.4.1)	Set	Interpret
VCI	N/A(1)	O(2)
Payload_Type	N/A(1)	O
Segment_Priority	N/A(1)	O
HCS: Setting at Source	N/A(1)	N/A
HCS: Detection at Destination	N/A	O(2)
HCS: Correction at Destination	N/A	O
PA Segment Payload	O(1)	O

(1) = M for head of bus function.
(2) = M if node supports optional function of setting and/or interpreting PA segment payload.

Table 6-9—IMPDU conformance

Initial MAC PDU (See 6.5.1)	Set	Interpret
Common PDU Header (6.5.1.1)		
Reserved (all bits set to 0)	M	O
BEtag	M	M
BAsize	M	O
MCP Header (6.5.1.2)		
Destination Address (6.5.1.2.2)		
48 bit, universally administered	M	M
48 bit, locally administered	O	O
16 bit	O	O
60 bit, all forms	O	O
Source address (6.5.1.2.3)		
48 bit, universally administered	M	M
48 bit, locally administered	O	O
16 bit	O	O
60 bit, all forms	O	O
Protocol Identification	M	M
PAD Length	M	M
Quality of Service: Delay	M	M
Quality of Service: Loss (Set to 0)	M	N/A
CRC32 Indicator Bit = 0	M	M

Table 6-9—IMPDU conformance *(Continued)*

Initial MAC PDU (See 6.5.1)	Set	Interpret
CRC32 Indicator bit = 1	O	M
Header Extension Length = 0	M	M
Header Extension Length > 0	O	M
BRIDGING (all 16 bits set to 0)	M	N/A
Header Extension (6.5.1.3)	O	O(1)
INFO (6.5.1.4)		
Length of 0–9188 for PI = 1	M	M
PAD (6.5.1.5)	M	O(2)
CRC32 (6.5.1.6)	O	O(3)
Common PDU trailer (6.5.1.7)		
Reserved (all bits set to 0)	M	O
BEtag	M	M
Length	M	M

(1) = All nodes shall recognize the existence of the Header Extension field, as indicated by a nonzero value of the Header Extension Length subfield. Nodes are not required to interpret the contents of the Header Extension field.

(2) = All nodes shall recognize the existence of a nonzero length PAD field. Nodes are not required to interpret the contents of the PAD field.

(3) = All nodes shall recognize the existence of the CRC32 field, as indicated by a one value of the CRC32 Indicator Bit. Nodes are not required to interpret the contents of the CRC32 field.

Table 6-10—DMPDU conformance

Derived MAC PDU (See 6.5.2)	Set	Interpret
DMPDU Header (6.5.2.1)		
Segment_Type	M	M
Sequence_Number	M	M
Message Identifier	M	M
DMPDU trailer (6.5.2.2)		
Payload_Length	M	M
Payload_CRC: Setting at Source	M	N/A
Payload_CRC: Detection at Destination	N/A	M
Payload_CRC: Correction of DMPDU Header at Destination	N/A	O

7. DQDB Layer facilities

All of the DQDB Layer facilities described may be operated upon using certain DQDB Layer Management Interface (LMI) operations described in clause 9.

The following conventions apply to the suffixes used for facilities described in this clause. If the suffix is not used, then all facilities in the node that have that name are being referenced.

x, where x = A or B, refers to the forward bus at the node for the facility being described.
y, where y = B or A, is used in combination with x to refer to the opposite bus for the facility being described (e.g., if the forward bus is Bus A, then the opposite bus is Bus B).
z, where z = 1, 2, or Default, is used to refer to one of the Slot Generator (SG) functions or associated Configuration Control (CC) functions in a node, as described in 5.4.2.3, 10.2.1, and 10.2.2.
w, where w = 1 or 2, refers to the specified SG or Configuration Control (CC) functions.
I, where I = 0, 1, 2, is used to refer to a distributed queue priority level.

7.1 Timers

7.1.1 Reassembly IMPDU Timer (RIT)

A Reassembly IMPDU Timer (RIT) is associated by the MAC Convergence Function (MCF) protocol entity with each Initial MAC Protocol Data Unit (IMPDU) being reassembled from Derived MAC Protocol Data Units (DMPDUs). The period of each RIT is a system parameter, RIT_PERIOD. (See 7.3.1.)

The RIT is initialized and started upon the arrival at a VCI, which the node is programmed to receive, of a valid Beginning of Message (BOM) DMPDU destined for the node. This arrival generates a reassembly process controlled by the Reassembly State Machine, associated with the VCI at which the DMPDU was received and the Message Identifier (MID) contained in the BOM DMPDU header. (See 8.2.1.)

The RIT is cleared upon arrival from the same VCI of a valid End of Message (EOM) DMPDU containing the same MID.

If the RIT expires before arrival of the valid EOM DMPDU, the reassembly process discards all state information relating to the associated IMPDU without generating an MA-UNITDATA indication primitive.

7.1.2 Head of Bus Arbitration Timer (Timer_H)

A Head of Bus Arbitration Timer (Timer_H_w, w = 1 or 2) is associated with each Configuration Control (CC) function of Type 1 or 2 that can support a Head of Bus A (HOB_A) function or a Head of Bus B (HOB_B) function, respectively, at a node. The period of each Timer_H_w is a system parameter, Timer_H_PERIOD. (See 7.3.2.) The Timer_H_w runs during the arbitration period to determine which CC_1 or CC_2 function should activate the Head of Bus A or Head of Bus B function when the current head of bus function for Bus A or Bus B becomes unavailable. This function is described in 10.2.3.4.

7.2 Counters

7.2.1 Request counter (REQ_I_CNTR)

The Queued Arbitrated functions block maintains two independent request counters at each priority level I (I = 0,1,2): one for Bus A (REQ_I_CNTR_A) and one for Bus B (REQ_I_CNTR_B). The REQ_I_CNTR_x (x = A or B) counters are used by the Distributed Queue State Machine (DQSM) (see 8.1.1), which performs

the access control function for QA segments. Each REQ_I_CNTR_x is associated with a CD_I_CNTR_x and a REQ_I_Q_y.

Each REQ_I_CNTR_x counter operates at all times, and is used by the access unit in queueing a new QA segment at priority level I in the distributed queue for Bus x. If the access unit does not have a QA segment queued at priority level I for access to Bus x, the counter indicates the number of currently outstanding requests from downstream for access to Bus x at priority level I or higher. If the access unit does have a QA segment queued at priority level I for access to Bus x, the counter indicates the number of requests from downstream for access to Bus x at priority level I that arrived at the access unit after the QA segment was queued.

Each REQ_I_CNTR_x counter has a minimum value of zero, and a maximum value of $(2^{16} - 1)$. Changes that would take the REQ_I_CNTR_x counter value beyond the minimum or maximum value leave the counter at its minimum or maximum value, respectively. The REQ_I_CNTR_x counter is initialized to zero when the node is powered up.

7.2.2 Countdown counter (CD_I_CNTR)

The Queued Arbitrated Functions block maintains two independent countdown counters at each priority level I (I = 0,1,2): one for Bus A (CD_I_CNTR_A) and one for Bus B (CD_I_CNTR_B). The CD_I_CNTR_x (x = A or B) counters are used by the DQSM (see 8.1.1), which performs the access control function for QA segments. Each CD_I_CNTR_x is associated with a REQ_I_CNTR_x and a REQ_I_Q_y.

Each CD_I_CNTR_x counter operates when the access unit has a QA segment queued at priority level I for access to Bus x. The counter indicates the number of outstanding requests made for access to Bus x at priority level I or higher that have to be satisfied on Bus x before the QA segment can be transmitted.

Each CD_I_CNTR_x counter has a minimum value of zero, and a maximum value of $(2^{16} - 1)$. Changes that would take the CD_I_CNTR_x counter value beyond the minimum or maximum value leave the counter at its minimum or maximum value, respectively. The CD_I_CNTR_x counter is initialized to zero when the node is powered up.

7.2.3 Local request queue counter (REQ_I_Q)

The Queued Arbitrated Functions block maintains two independent local request queue counters at each priority level I (I = 0,1,2): one for Bus B (REQ_I_Q_B) and one for Bus A (REQ_I_Q_A). The REQ_I_Q_y (y = B or A) counters are used by the REQ Queue Machine (see 8.1.2), which controls the writing of requests at priority level I into the REQ_I subfield of the Access Control Field (ACF) of slots passing on Bus y. Each REQ_I_Q_y is associated with a REQ_I_CNTR_x and a CD_I_CNTR_x.

Each REQ_I_Q_y counter holds the number of local requests that the access unit has queued to send at priority level I, REQ_I, on Bus y, but has not yet sent.

Each REQ_I_Q_y counter has a minimum value of zero, and a maximum value of $(2^8 - 1)$. Changes that would take the REQ_I_Q_y counter value beyond the minimum or maximum value leave the counter at its minimum or maximum value, respectively. The REQ_I_Q_y counter is initialized to zero when the node is powered up.

7.2.4 Page counter (PAGE_CNTR)

Message Identifiers (MIDs) are obtained by the DQDB Layer Management Protocol Entity through the MID Page Allocation function for use by the MAC Convergence Function block. (See 5.4.4.) The unit of allocation of MIDs is an MID page of one MID value.

The DQDB Layer Management Protocol Entity in the node containing the active Head of Bus A function maintains a page counter (PAGE_CNTR_HOBA) in order to generate the appropriate MID Page Allocation field values for Bus A. (See 10.3.1–3.)

When the MID Page Allocation function is enabled, the PAGE_CNTR_HOBA counter has a minimum value of MIN_PAGE = 1 and a maximum value of MAX_PAGE = $(2^{10} - 1)$. The PAGE_CNTR_HOBA counter is initialized to zero when the node is powered up and is also set to zero when the MID Page Allocation function is disabled.

The DQDB Layer Management Protocol Entity for each node maintains two separate Page Counters (PAGE_CNTR_x), one for Bus A (PAGE_CNTR_A) and one for Bus B (PAGE_CNTR_B). Each of these counters holds the page number of the page of MIDs that is associated with the next Page Reservation (PR) subfield to be received on the related bus. The operation of the PAGE_CNTR_x counters is controlled by the Page Counter State Machine for that bus. (See 10.3.4.)

When the MID Page Allocation function is enabled and the PAGE_CNTR_x counter is synchronized to the MID page values associated with the PR subfields on Bus x, then the PAGE_CNTR_x counter has a minimum value of MIN_PAGE = 1 and a maximum value of MAX_PAGE = $(2^{10} - 1)$. Each PAGE_CNTR_x counter shall be initialized to have a value of zero when the node is powered up. A PAGE_CNTR_x counter shall also be set to zero when the node loses synchronism with the MID page values associated with the PR subfields on Bus x.

7.2.5 Bandwidth balancing counter (BWB_CNTR)

The Queued Arbitrated functions block maintains two independent bandwidth balancing counters: one for Bus A (BWB_CNTR_A) and one for Bus B (BWB_CNTR_B). The BWB_CNTR_x (x = A or B) counters are used by the Bandwidth Balancing Machine (see 8.1.3).

Each BWB_CNTR_x counter operates when the node transmits a QA segment on Bus x. The counter is used to determine when the node should skip the use of an empty QA slot in order to facilitate effective sharing of the bandwidth on Bus x.

Each BWB_CNTR_x counter has a minimum value of zero, and a maximum value of $(2^6 - 1)$. The BWB_CNTR_x counter is initialized to zero when the node is powered up.

7.2.6 Transmit sequence number counter (TX_SEQUENCE_NUM)

The MAC Convergence Function (MCF) block maintains one transmit sequence number counter (TX_SEQUENCE_NUM) for each combination of MID value which is contained in the node MID page list (see 5.4.4.2.1) and VCI value which the node is programmed to receive on behalf of the MCF block (see 9.2.1). The TX_SEQUENCE_NUM counter associated with an MID/VCI pair contains the value to be inserted in the Sequence_Number subfield of the next DMPDU sent by the node that contains the associated MID and VCI values.

Each TX_SEQUENCE_NUM counter has a minimum value of zero, and a maximum value of $(2^4 - 1)$. The value of the TX_SEQUENCE_NUM counter associated with an MID is set to zero (for all VCIs which the node is programmed to receive on behalf of the MCF block) when that MID is obtained by the Get Page State Machine. (See 10.3.6.) The TX_SEQUENCE_NUM counter is incremented by one (modulo 16) whenever a DMPDU containing the associated MID value is sent by the node.

7.3 System parameters

System parameters shall be initialized on all nodes before installation. They may be operated upon remotely by network management procedures if the node supports the DQDB LMI. (See 9.6.)

7.3.1 Reassembly IMPDU timer period (RIT_PERIOD)

The system parameter RIT_PERIOD sets the period of operation for each Reassembly IMPDU Timer associated with an active reassembly process under the control of an RSM. (See 7.1.1.) The default value of RIT_PERIOD when the node is powered up shall be at least 0.7 s.[51]

7.3.2 Head of Bus Arbitration Timer period (Timer_H_PERIOD)

The system parameter Timer_H_PERIOD sets the period of operation for each Head of Bus Arbitration Timer associated with a Configuration Control function of Type 1 or 2 (Timer_H_w, w = 1 or 2). (See 7.1.2.) The value of Timer_H_PERIOD is a function of both the underlying transmission system and the requirements of the network administrator. The default value of Timer_H_PERIOD when the node is powered up shall be 5 s.[52]

7.3.3 Quality of service map (QOS_MAP)

The MCF protocol entity in each access unit maintains a quality of service map (QOS_MAP), which describes the mapping to be used between the priority parameter received in MA-UNITDATA request primitives and the access queue priority level I (I = 0,1,2) to be used by the Queued Arbitrated Functions block to access the medium at the requested priority. (See 5.1.1.1.4.)

The QOS_MAP element is a multivalued attribute that contains eight single-valued elements, QOS_MAP_J (J = 0,1,2,3,4,5,6,7). Each QOS_MAP_J corresponds to one of the priority levels that may be requested in an MA-UNITDATA request, and contains the value of the priority level I to be used to access the medium corresponding to the requested priority level J. For each QOS_MAP_J in the sequence from J = 0 to 7, inclusive, the corresponding values of QOS_MAP_J shall form a monotonically nondecreasing sequence (i.e., when comparing QOS_MAP_(J+1) with QOS_MAP_J, the value of QOS_MAP_(J+1) shall be greater than or equal to the value of QOS_MAP_J). The value of each QOS_MAP_J shall be 0 for conforming nodes.

7.3.4 Reserved number of MID pages (RESERVED_MID_PAGES)

The DQDB Layer Management Entity at the node with the active Head of Bus A function maintains the single-valued RESERVED_MID_PAGES attribute, which contains the number of MID pages that are reserved by network management for centralized allocation. The PR subfields associated with these MID page values are to be marked as RESERVED by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function. (See 10.3.2.)

The minimum and default value of RESERVED_MID_PAGES is zero. The maximum value of RESERVED_MID_PAGES is $(2^{10} - 1)$.

---

[51]This is the worst-case value for RIT_PERIOD, which assumes maximum-length IMPDUs being sent on an overloaded subnetwork of 512 nodes, operating at 150 Mbit/s over a 160 km loop length, with 10% transmission overhead. The value of RIT_PERIOD could be modified to different values for different nodes on the subnetwork using the DQDB Layer Management operation defined in 9.6.1. It could also be tuned for subnetworks different from the example quoted. In particular, it could be less for smaller subnetworks, subnetworks operating at lower speed, or subnetworks with less nodes.

[52]The value of Time_H_Period should be the same for all nodes with Configuration Control Type 1 or 2 functions capable of supporting Head of Bus functions on the subnetwork. However, it could be tuned once the subnetwork is fully operational.

7.3.5 Maximum number of MID pages (MAX_MID_PAGES)

The DQDB Layer Management Entity for each node maintains the single-valued MAX_MID_PAGES attribute, which contains the maximum number of MID pages that can be obtained by the node using the MID Page Allocation function.

The default value of MAX_MID_PAGES is one.[53]

7.3.6 Bandwidth balancing modulus (BWB_MOD)

The system parameter BWB_MOD sets the modulus for each BWB_CNTR_x (x = A or B). The BWB_MOD parameter is associated with the control of access via the Distributed Queueing protocol. The value of the parameter shall be in the range 0 to 64, inclusive. The value of 0 implies that the bandwidth balancing mechanism is disabled. A nonzero value of BWB_MOD, n (where n is in the range 1 to 64, inclusive), means that the maximum fraction of QA slots that the node can use on either bus is $n / (n + 1)$. The default value of BWB_MOD shall be 8.

7.4 Flags

7.4.1 Configuration Control Flag (CC_12_CONTROL)

The Configuration Control Flag (CC_12_CONTROL) is used to control the operation of the Configuration Control functions of Type 1 and 2 in nodes that do not contain the Default Configuration Control function. The Configuration Control Flag shall contain one of the values:

NORMAL
DISABLED

The value NORMAL directs the Configuration Control functions of Type 1 and 2 to operate normally. The value DISABLED directs the Configuration Control functions of Type 1 and 2 to take no action.

The default value of CC_12_CONTROL at node power-up is DISABLED. According to 5.4.2.2, the value of CC_12_CONTROL shall not be changed locally until a valid Bus Identification field (see 10.1.1) is received at the node, which indicates to the node the identity of the Physical Layer service access points (PhSAPs).

CC_12_CONTROL may be set by the DQDB Layer Management Interface operation defined in 9.7.1 to NORMAL only if the node is capable of supporting Head of Bus functions.

7.4.2 Default Configuration Control Flag (CC_D2_CONTROL)

The Default Configuration Control Flag (CC_D2_CONTROL) for the node containing the Default Configuration Control function (CC_D) is used to control the operation of the Default Configuration Control (CC_D) function and Configuration Control Type 2 (CC_2) function at the node. The Default Configuration Control Flag shall contain one of the values:

NORMAL
DISABLED

[53]End user nodes would normally require only one MID page. However, the value of MAX_MID_PAGES could be set to a value greater than one for some nodes on the subnetwork, such as bridges. Such a change should only be made by action of the network administrator.

The value NORMAL directs the CC_D function and CC_2 function to operate normally. The value DISABLED directs the CC_D function and CC_2 function to take no action.

The default value of CC_D2_CONTROL at node power-up is NORMAL. CC_D2_CONTROL may subsequently be set to either of these values by the DQDB LMI operation defined in 9.7.1.

7.4.3 CRC32 Generation Flag (CRC32_GEN_CONTROL)

The CRC32 Generation Flag (CRC32_GEN_CONTROL) is used to control the operation of the CRC32 Generation function at the node. The CRC32 Generation Flag shall contain one of the values:

ON
OFF

The value ON directs the MAC Convergence function to generate the CRC32 field in IMPDUs, as described in 5.1.1.1.1. The value OFF directs the MCF to not create a CRC32 field in IMPDUs.

The default value of CRC32_GEN_CONTROL at node power-up is OFF. CRC32_GEN_CONTROL may subsequently be set to ON by the DQDB LMI operation defined in 9.8.1 only if the node is capable of supporting CRC32 generation.

7.4.4 CRC32 Checking Flag (CRC32_CHECK_CONTROL)

The CRC32 Checking Flag (CRC32_CHECK__CONTROL) is used to control the operation of the CRC32 Checking function at the node. The CRC32 Checking Flag shall contain one of the values:

ON
OFF

The value ON directs the MAC Convergence function to check the CRC32 field, if present, in an IMPDU, as described in 5.1.1.2.4. The value OFF directs the MCF to not check a CRC32 field in an IMPDU, even if present.

The default value of CRC32_CHECK_CONTROL at node power-up is OFF. CRC32_CHECK_CONTROL may subsequently be set to ON by the DQDB LMI operation defined in 9.8.1 only if the node is capable of supporting CRC32 checking.

7.5 Resource status indicators

The allowed combinations of Configuration Control functions in a node are specified in 10.2.1 as follows:

a) A single node in the subnetwork contains a Default Configuration Control (CC_D) function and a Configuration Control Type 2 (CC_2) function.

b) All other nodes in the subnetwork contain a Configuration Control Type 1 (CC_1) function and a Configuration Control Type 2 (CC_2) function.

The resource status indicators described in this clause are used to record the status of resources that are under the control of the Configuration Control functions at the node.

7.5.1 Configuration Control Status Indicator (CC_STATUS)

There is one Configuration Control Status Indicator (CC_STATUS_z, z = 1, 2, or Default) associated with each Configuration Control (CC) function present in a node. The status indicator records whether any of the resources under the control of the CC function are active or inactive and shall contain one of the values:

ACTIVE
INACTIVE

If the CC_STATUS is ACTIVE for either Configuration Control function at a node, the DQDB Layer Management Protocol Entity at the node shall operate on the appropriate subfield of the SubNetwork Configuration field, as defined in 10.2.3 to 10.2.6.

The default value of CC_STATUS_D at power-up is ACTIVE. The default value of CC_STATUS_2 for the CC_2 function in the node that contains the CC_D function is ACTIVE at power-up. The default value of CC_STATUS_2 for all other CC_2 functions and of CC_STATUS_1 for all CC_1 functions is INACTIVE at power-up.

7.5.2 Head of Bus Operation Indicator (HOB_OPERATION)

There is one Head of Bus Operation Indicator (HOB_OPERATION_z, z = 1, 2, or Default) associated with each Configuration Control function in a node. The Head of Bus Operation Indicator records which functions the Head of Bus function is performing, as set by the Configuration Control protocol described in 10.2.4 to 10.2.6.

The Head of Bus Operation Indicator for a node that is not performing any Head of Bus function contains the value NULL.

The Head of Bus Operation Indicator for a node that is performing Head of Bus functions shall contain one or more of the values:

DEFAULT_SLOT_GENERATOR
HEAD_OF_BUS_A
HEAD_OF_BUS_B

The valid combinations of these values are described in 10.2.4 to 10.2.6.

The default values of HOB_OPERATION_D at power-up are DEFAULT_SLOT_GENERATOR and HEAD_OF_BUS_A. The default value of HOB_OPERATION_2 for the CC_2 function in the node that contains the CC_D function is HEAD_OF_BUS_B at power-up. The default value of HOB_OPERATION_2 for all other CC_2 functions and of HOB_OPERATION_1 for CC_1 functions is NULL.

7.5.3 Link Status Indicator (LINK_STATUS)

The Link Status Indicator (LINK_STATUS_z, z = 1, 2, or Default) records the status of the duplex transmission link for the PhSAP associated with the Configuration Control (CC) function of Type z. PhSAP_A is associated with either the CC_D function at the node containing the Default Slot Generator function or the CC_1 function at other nodes. PhSAP_B is associated with a CC_2 function.

The Link Status Indicator is set as a result of the status parameter received in a PhSTATUS indication (see 4.6) at the appropriate PhSAP, and is used in the Configuration Control protocol. (See 10.2.4 to 10.2.6.) The Link Status Indicator shall contain one of the values:

UP
DOWN

The default value of LINK_STATUS_z at power-up is DOWN.

7.5.4 External Timing Source Status Indicator (ETS_STATUS)

The External Timing Source Status Indicator (ETS_STATUS) records the status of the External Timing Source function at the node. It is used in the Configuration Control protocol. (See 10.2.4 to 10.2.6.) The External Timing Source Status Indicator shall contain one of the values:

UP
DOWN

The conditions under which the ETS is declared to be UP (available and to be used by the node to provide external timing) or DOWN (not available or not to be used by the node) shall be specified by the network administrator and, hence, are outside the scope of this part of ISO/IEC 8802.

8. DQDB Layer operation

This clause provides the description of the protocol machines used by the DQDB Layer functions to support the DQDB Layer services. The functional description of a DQDB node is given in clause 5.

8.1 Distributed Queue operation

The Queued Arbitrated (QA) Functions block includes the functions required to support operation of the Distributed Queue.

The Distributed Queue operates the request, REQ_I_CNTR_x, and countdown, CD_I_CNTR_x, counters for each combination of priority level I (I = 0,1,2) and Bus x (x = A or B). It also operates the bandwidth balancing counter, BWB_CNTR_x for each Bus x. The counters (see 7.2.1, 7.2.2, and 7.2.5) are used to control the write access of Queued Arbitrated (QA) segments to empty QA slots on the respective bus.

Operation of the request and countdown counters for Bus x involves read operations on the BUSY bit and SL_TYPE bit in the Access Control Field (ACF) (see 6.2.1) for all slots on Bus x and read operations on the REQUEST field in the ACF of all slots on the opposite bus, Bus y (y = B or A, respectively). This operation is controlled by the Distributed Queue State Machine (DQSM) and is described in 8.1.1.

A QA segment is placed in the queue(s) indicated by the combination of the TX_BUS_SIGNAL and ACCESS_QUEUE_SIGNAL values asserted when the QA segment payload is received by the QA Functions block. (See 5.1.2.1.3 and 5.1.2.1.4.) Request for access to Bus x is signaled to other nodes by a write operation performed on the REQUEST field of slots passing on the opposite bus, Bus y. This operation is controlled by the REQ Queue Machine (RQM), which operates the local request queue counter, REQ_I_Q_y, and is described in 8.1.2.

Operation of the bandwidth balancing counter for Bus x is related to the sending of QA segments on Bus x and effects the value of the request and countdown counters for Bus x. This operation is controlled by the Bandwidth Balancing Machine (BWBM) and is described in 8.1.3.

8.1.1 Distributed Queue State Machine (DQSM)

There are six instances of the DQSM at each node: one for each possible combination of Bus x (x = A or B) and priority level I (I = 0, 1, 2), that can be requested by the TX_BUS_SIGNAL and ACCESS_QUEUE_SIGNAL values asserted when the QA Functions block receives the QA segment payload. (Priority level 2 is the highest priority level.) All instances of the DQSM for a particular bus pass information between themselves about requests for access made by the node itself, called self-requests.

The generic DQSM for Queued Arbitrated access control to Bus x at priority level I is specified in figure 8-1. It describes the control of access for a QA segment to one Bus x (x = A or B), for one priority level I (I=0,1,2). The other bus is defined as Bus y (y = B or A, respectively). There is a request counter, REQ_I_CNTR_x, and a countdown counter, CD_I_CNTR_x, controlled by each DQSM.

There can only be a maximum of six QA segments queued within the Distributed Queue for access at each node, one at each instance of the DQSM. However, the QA Functions block can accept multiple QA segment payloads waiting for access to Bus x at priority level I, by maintaining a local queue of QA segments that cannot be placed in the Distributed Queue. The order of QA segments waiting for access to Bus x at priority level I shall be maintained by the QA Functions block.

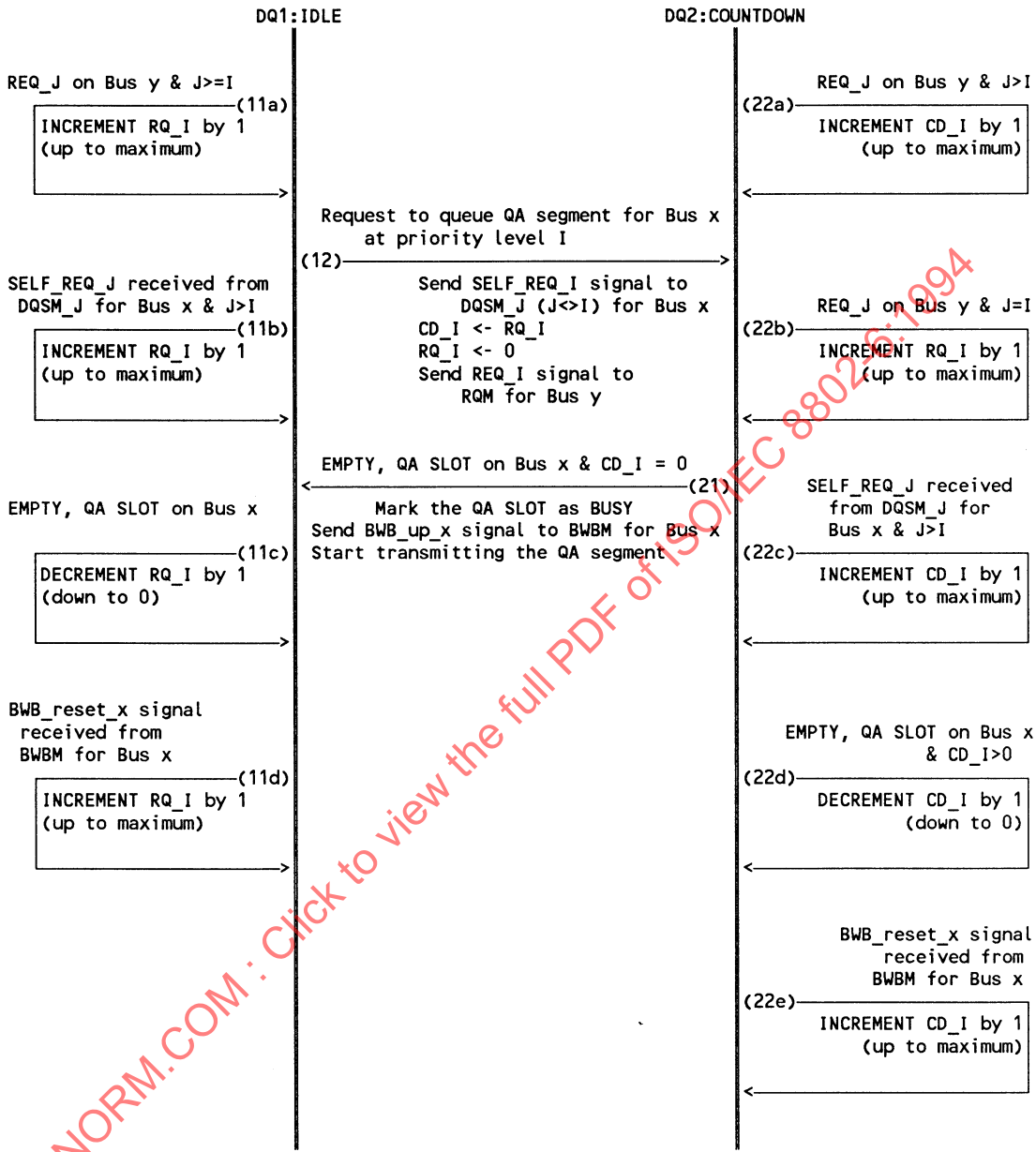


Figure 8-1—DQSM for Bus x at Priority Level I

Operation of the DQSM

The DQSM for access at priority level I (I = 0,1,2) to Bus x (x = A or B) can be in one of two states: Idle or Countdown.

In both states, the DQSM shall observe Bus y for requests generated by nodes downstream on Bus x. A request at priority level J is indicated by the REQ_J bit in the Access Control Field (ACF) being set to one. The DQSM shall also observe Bus x for empty QA slots, which are indicated by the BUSY bit and SL_TYPE bit of the ACF both being set to zero.

NOTE—Slots are considered to be empty QA slots if they are *received* at the node with the BUSY bit and SL_TYPE bit being set to zero. This ensures that the correct request and countdown counter values are maintained if the highest priority segment queued at the node gains access to the empty QA slot and the node marks the slot as busy.

The DQSM is in the Idle state when it has no QA segment to be transferred at priority level I on Bus x. While in this state the DQSM maintains a count of the requests generated by nodes downstream on Bus x at priority level I or higher that are still unsatisfied. The count also includes requests from the node itself for access to Bus x at higher priorities than priority I. This count is maintained in REQ_I_CNTR_x, the value of which is denoted as RQ_I in figure 8-1.

The DQSM is in the Countdown state when it has a QA segment queued for transfer at priority level I on Bus x, but is not yet permitted to access empty QA slots received on Bus x. The permission for access is controlled by a counter called CD_I_CNTR_x, the value of which is denoted as CD_I in figure 8-1.

State DQ1: Idle

The DQSM remains in the Idle state until it is requested to queue a QA segment for transfer on Bus x at priority level I (Transition 12). While in the Idle state the value of the REQ_I_CNTR_x shall be maintained by actions in the Transitions 11a, 11b, 11c, and 11d.

Transition 11a

The DQSM shall read all of the REQ bits in the ACF of every slot on Bus y for requests generated by nodes downstream on Bus x. Whenever the DQSM reads a request at priority level J, REQ_J, on Bus y, where (J is greater than or equal to I), and REQ_I_CNTR_x is not equal to its maximum value, then REQ_I_CNTR_x shall be incremented by 1.

Transition 11b

If other DQSMs within the node associated with Bus x generate a self-request for a priority level J where (J is greater than I), and REQ_I_CNTR_x is not equal to its maximum value, then REQ_I_CNTR_x shall be incremented by 1.

Transition 11c

If an empty QA slot is received on Bus x and REQ_I_CNTR_x is not equal to 0, then REQ_I_CNTR_x shall be decremented by 1.

Transition 11d

If a BWB_reset_x signal is received from the Bandwidth Balancing Machine (BWBM) for Bus x, and REQ_I_CNTR_x is not equal to its maximum value, then REQ_I_CNTR_x shall be incremented by 1.

Transition 12

When the DQSM is in the Idle state and receives a request to queue a QA segment for transfer on Bus x at priority level I, then it shall

a) Inform the other DQSMs within the node associated with Bus x of the self-request.
b) Transfer the value of REQ_I_CNTR_x to CD_I_CNTR_x.
c) Set the value of REQ_I_CNTR_x to 0.
d) Indicate to the associated REQ Queue Machine that a request is to be sent at priority level I on Bus y. (See 8.1.2.)
e) Enter the Countdown state.

State DQ2: Countdown

The DQSM remains in the Countdown state until the QA segment that is queued for transfer on Bus x is written into an empty QA slot (Transition 21). While in the Countdown state, the value of the CD_I_CNTR_x shall be maintained by actions in Transitions 22a, 22c, 22d, and 22e and the value of the REQ_I_CNTR_x shall be maintained by the action in Transition 22b.

Transition 21

If an empty QA slot is received on Bus x and CD_I_CNTR_x equals 0 then the DQSM shall

a) Mark the QA slot as busy by setting the BUSY bit to one.
b) Send a BWB_up_x signal to the BWBM for Bus x.
c) Start writing the QA segment into the QA slot.[54]
d) Enter the Idle state.

NOTE—If the CD_I_CNTR_x equals 0 for more than one value of I, then only the segment for the higher value of I shall be sent. This situation can only occur under exceptional conditions.

Transition 22a

The DQSM shall read all of the REQ bits in the ACF of every slot on Bus y for requests generated by nodes downstream on Bus x. Whenever the DQSM reads a request at priority level J, REQ_J, on Bus y, where (J is greater than I), and CD_I_CNTR_x is not equal to its maximum value, then CD_I_CNTR_x shall be incremented by 1.

Transition 22b

The DQSM shall read all of the REQ bits in the ACF of every slot on Bus y for requests generated by nodes downstream on Bus x. Whenever the DQSM reads a request at priority level J, REQ_J, on Bus y, where (J equals I), and REQ_I_CNTR_x is not equal to its maximum value, then REQ_I_CNTR_x shall be incremented by 1.

Transition 22c

If other DQSMs within the node associated with Bus x generate a self-request for a priority level J where (J is greater than I), and CD_I_CNTR_x is not equal to its maximum value, then CD_I_CNTR_x shall be incremented by 1.

[54]A node is allowed to queue another QA segment for transfer on Bus x at priority level I (Transition 12) as soon as it has marked the QA slot as Busy. The node shall send the BWB_up_x signal when it has set the BUSY bit to one.

Transition 22d

If an empty QA slot is received on Bus x and CD_I_CNTR_x is greater than 0 then CD_I_CNTR_x shall be decremented by 1.

Transition 22e

If a BWB_reset_x signal is received from the BWBM for Bus x, and CD_I_CNTR_x is not equal to its maximum value, then CD_I_CNTR_x shall be incremented by 1.

8.1.2 REQ Queue Machine (RQM)

The DQSM for Bus x at priority level I (figure 8-1) shows the generation of a request at priority level I, REQ_I, to be sent on the opposite bus, Bus y, as action d) arising from Transition 12. The sending of REQ_I's is controlled by the RQM.

Each DQSM on Bus x (x = A or B) is uniquely associated with an RQM on Bus y (y = B or A, respectively). Therefore, each instance of the RQM queues requests to be sent on Bus y for access of QA segments from the associated DQSM onto Bus x.

An RQM operates an REQ_I_Q_y counter. The REQ_I_Q_y counter holds the number of requests at priority level I, REQ_I, which are to be sent on Bus y. The counter is incremented for each indication received from the associated DQSM that a request is to be sent at priority level I on Bus y. The counter is only decremented when the node reads the REQ_I bit in the Access Control Field (ACF) of a slot on Bus y which is zero and it sets that bit to one.

Operation of the RQM

When the node generates a self-request at priority level I for access to Bus x, then the value of REQ_I_Q_y shall be incremented by one.

Whenever the REQ_I_Q_y counter is greater than zero, the RQM shall attempt to send a request by setting the REQ_I bit on Bus y to one in the ACF of the first slot in which the REQ_I bit is set to zero when received.

When a request is sent by setting an REQ_I bit that was zero to one, then the value of REQ_I_Q_y is decremented by one.

NOTE—Transfer of the QA segment by the DQSM on Bus x does not require that the RQM has sent the request on Bus y. That is, the operation of writing REQ bits and sending QA segments is independent. However, the REQ_I_Q_y counter shall not be decremented when the QA segment with which the REQ_I is associated is transmitted on Bus x. The REQ_I shall be recorded in the REQ_I_Q_y counter until it is transmitted on Bus y. The relationship between the DQSM and RQM is described in annex E.

8.1.3 Bandwidth Balancing Machine (BWBM)

The DQSM for Bus x at priority level I (figure 8-1) shows the generation of a BWB_up_x signal as action (2) arising from Transition 21, and the reception of a BWB_reset_x signal triggering Transitions 11d and 22e. These signals are used for communication between the BWBM for Bus x and each of the three DQSMs for Bus x with which the BWBM is associated.

The BWBM for Bus x operates the bandwidth balancing counter for Bus x, BWB_CNTR_x. There is a related system parameter called the Bandwidth Balancing Modulus (BWB_MOD). (See 7.3.6.) If BWB_MOD equals zero, the bandwidth balancing mechanism is disabled. If BWB_MOD is greater than zero, it indicates when the value of BWB_CNTR_x should roll over to zero, as described below.

Whenever any of the three DQSMs for Bus x sends a BWB_up_x signal to the BWBM for Bus x, indicating that the DQSM has transmitted a segment on Bus x, the BWB_CNTR_x is incremented by one, provided that it does not have a value of (BWB_MOD–1). If BWB_CNTR_x does have a value of (BWB_MOD–1) when the BWBM for Bus x receives a BWB_up_x signal, the BWBM resets BWB_CNTR_x to zero and sends a BWB_reset_x signal to each of the three DQSMs for Bus x. This signal informs the DQSMs to skip the use of one empty QA slot on Bus x.

NOTE—If the value of BWB_MOD is one, BWB_CNTR_x always has the value zero. In this case the BWBM for Bus x sends a BWB_reset_x signal to each of the three DQSMs for Bus x every time the BWBM receives a BWB_up_x signal from any of the DQSMs for Bus x.

Operation of the BWBM

If the value of BWB_MOD equals zero, the BWBM for Bus x is disabled: it does not respond to BWB_up_x signals from the DQSMs for Bus x and does not send BWB_reset_x signals to the DQSMs for Bus x.

If the value of BWB_MOD is greater than zero and the value of BWB_CNTR_x is less than (BWB_MOD-1), then receipt of a BWB_up_x signal from any of the DQSMs for Bus x shall cause the BWBM for Bus x to increment the value of BWB_CNTR_x by one.

If the value of BWB_MOD is greater than zero and the value of BWB_CNTR_x equals (BWB_MOD–1), then receipt of a BWB_up_x signal from any of the DQSMs for Bus x shall cause the BWBM for Bus x to reset the value of BWB_CNTR_x to zero and send a BWB_reset_x signal to each of the three DQSMs for Bus x. (If the value of BWB_MOD equals one, a BWB_reset_x signal is sent every time a BWB_up_x signal is received.)

8.2 Reassembly operation

The process of reassembly is used to reconstruct an Initial MAC Protocol Data Unit (IMPDU) from the segmentation units contained in Derived MAC Protocol Data Units (DMPDUs) received by the node that have passed the DMPDU validity checks. Any IMPDU that has more than one DMPDU derived from it, and is currently being received and reassembled by the node, has an instance of the Reassembly State Machine (RSM) associated with it to control the reassembly process.

As the DMPDUs from different IMPDUs destined to the node may arrive in an interspersed manner, the MAC Convergence Function (MCF) block includes the functions to support the simultaneous operation of multiple reassembly processes, each controlled by an active instance of the RSM.

For the purposes of reassembly at a receiver, two DMPDUs received with the same Message Identifier (MID) but from a different VCI, or with a different MID from the same VCI, shall be considered part of two different IMPDUs. Hence, in the abstract model of the MCF reassembly function used for this part of ISO/IEC 8802, there is an instance of the RSM for all of the $(2^{10} - 1)$ MIDs of every VCI that the MCF block is programmed to receive. Thus each RSM is uniquely associated with a VCI/MID pair.

However, the number of reassembly processes that can be supported simultaneously in a given node is an implementation choice. Therefore, the abstract model does not imply any particular implementation strategy, as it is recognized that the number of physical reassembly processes may be limited at a node to less than the maximum number of abstract RSMs.

Subclause 5.1.1.2.2, a)1), describes the operation of the MCF to select the appropriate RSM, and hence reassembly process, for a received BOM, COM, or EOM DMPDU. If the DMPDU is an SSM DMPDU, then it contains a complete IMPDU and does not need to be forwarded to a reassembly process. The function of processing an SSM DMPDU is described in 5.1.1.2.2, a)2). Subclause 5.1.1.2.3 describes the functional

operation of reassembly of an IMPDU from BOM, COM, and EOM DMPDUs. Subclause 8.2.1 describes the operation of a single instance of the RSM to control a reassembly process.

NOTE—Conforming nodes need not implement the reassembly process as given by the RSM in 8.2.1, provided that the node can extract and deliver MAC Service Data Units (MSDUs) correctly from a received stream of DMPDUs.

8.2.1 Reassembly State Machine (RSM)

The MCF block contains an instance of a RSM for all of the $(2^{10} - 1)$ MIDs of every VCI that the MCF block is programmed to receive. The operation of an instance of the RSM is specified in figure 8-2. Each instance of the RSM is uniquely associated with a VCI/MID pair and may be associated with a reassembly process if the node is currently receiving an IMPDU for that VCI/MID pair.

Individual/Group Flag (IG_FLAG)

The abstract description of the RSM in figure 8-2 assumes that the MCF block associates an Individual/Group Flag (IG_FLAG) with each IMPDU being reassembled from DMPDUs.

The IG_FLAG is set upon arrival, at a VCI, which the node is programmed to receive, of a valid Beginning of Message DMPDU destined for the node. The IG_FLAG is set to INDIVIDUAL if the value of the Destination Address (DA) field in the MCP header is an individual address. (See 6.5.1.2.2.) Otherwise, the IG_FLAG is set to GROUP if the value of the DA field is a group address.

The IG_FLAG is used to determine whether to cause the PSR bit to be set upon receipt of the Beginning of Message DMPDU and upon receipt of Continuation of Message and End of Message DMPDUs derived from the same IMPDU, as described functionally in 5.1.1.2.2 and formally below.

Receive Sequence Number Counter (RX_SEQUENCE_NUM)

The abstract description of the RSM in figure 8-2 assumes that the MCF block associates a receive sequence number counter (RX_SEQUENCE_NUM) with each IMPDU being reassembled from DMPDUs.

Each RX_SEQUENCE_NUM counter has a minimum value of zero, and a maximum value of $(2^4 - 1)$.

The value of the RX_SEQUENCE_NUM counter is initialized on receipt of a valid Beginning of Message DMPDU to (one plus the value of the Sequence_Number subfield received in the BOM DMPDU header) (modulo 16). Subsequently, the value of the RX_SEQUENCE_NUM counter is compared with the value of the Sequence_Number subfield in each valid DMPDU received by the RSM. If the two values match, the RX_SEQUENCE_NUM counter is incremented by one (modulo 16) (in the case of COM DMPDUs). Hence, the value of RX_SEQUENCE_NUM associated with an active RSM always contains the expected value of the next DMPDU to be received by the RSM.

If the comparison between the value of RX_SEQUENCE_NUM counter and the value of the Sequence_Number subfield received by the RSM in a valid DMPDU fails, then the reassembly process is terminated early.

Operation of the RSM

An instance of the RSM can be in one of two states: Idle or Active.

The RSM is in the Idle state when the node is not receiving an IMPDU for the associated VCI/MID pair.

The RSM is in the Active state when it is controlling a process that is reassembling an IMPDU for the associated VCI/MID pair.

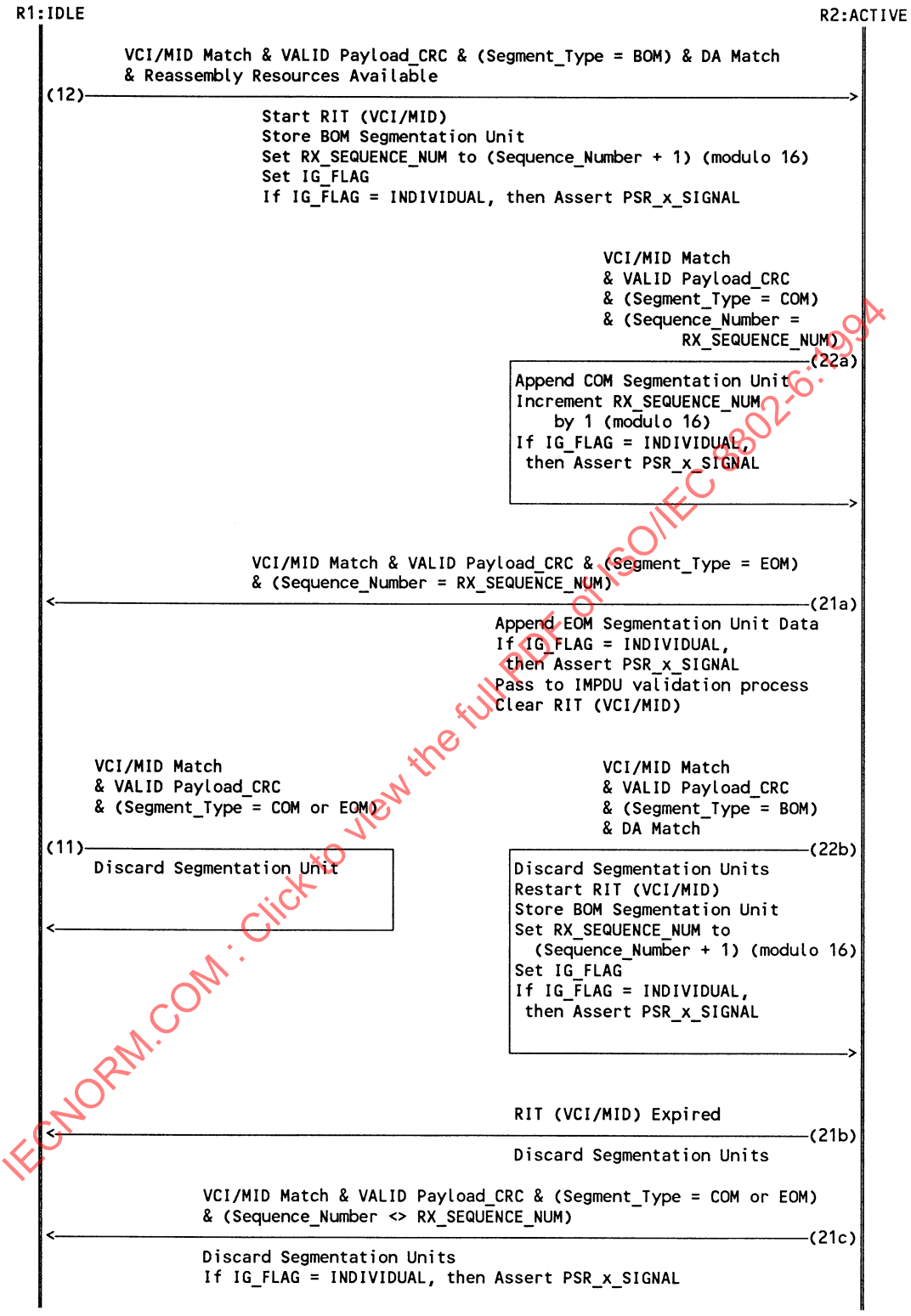


Figure 8-2—Instance of Reassembly State Machine (RSM)

State R1: Idle

The RSM remains in the Idle state until the node receives a BOM DMPDU destined to it at a VCI which the MCF block is programmed to receive, and both the MID in the BOM DMPDU header and the VCI match those associated with the RSM (Transition 12).

Transition 11

Whenever the RSM associated with a VCI/MID pair is in the Idle state and receives a DMPDU with matching VCI/MID, which has a valid Payload_CRC subfield and which contains either a COM or EOM Segment_Type subfield, then the RSM shall discard the segmentation unit.

Transition 12

Whenever the RSM associated with a VCI/MID pair is in the Idle state and receives a DMPDU with matching VCI/MID that has a valid Payload_CRC subfield, contains a BOM Segment_Type subfield, and contains a value in the Destination Address (DA) field of the MCP Header that matches an MSAP address which the node is programmed to receive, then it shall determine whether there are sufficient reassembly resources available at the node to receive the IMPDU.

If there are not sufficient reassembly resources, then the RSM shall discard the BOM segmentation unit. The MCF block may (but is not required to) assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block. Further, the MCF block may (but is not required to) assert the PSR_x_SIGNAL upon receipt of the COM DMPDUs and EOM DMPDU derived from the same IMPDU.

Otherwise, if there are sufficient reassembly resources, the RSM shall

a) Start a Reassembly IMPDU Timer (RIT) and associate it with the VCI/MID pair of the RSM.
b) Store the BOM segmentation unit.
c) Initialize the RX_SEQUENCE_NUM to (the value of the Sequence_Number subfield plus one) (modulo 16).
d) Enter the Active state.

If the value of the DA field is an individual address, the RSM shall set the IG_FLAG to INDIVIDUAL and the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block. Otherwise, the RSM shall set the IG_FLAG to GROUP.

If the IG_FLAG is set to INDIVIDUAL, the MCF block shall also assert the PSR_x_SIGNAL upon receipt of the COM DMPDUs and EOM DMPDU derived from the same IMPDU, as occurs during Transitions 22a and 21a, respectively. If Transition 22b occurs, then the value of IG_FLAG may be changed.

NOTE—As per 5.1.1.2.2b), Condition A, the MCF block shall assert the PSR_x_SIGNAL if the DA match is made, if the value of the DA is an individual MSAP address and if the Payload_CRC fails.

State R2: Active

The RSM remains in the Active state until the node receives a valid EOM DMPDU at a VCI which the MCF block is programmed to receive, and both the MID in the EOM DMPDU header and the VCI match those associated with the RSM (Transition 21a). While waiting for the EOM DMPDU, the RSM also processes any COM DMPDUs received for the associated VCI/MID pair (Transition 22a).

Certain error conditions cause the reassembly process to be terminated without reconstruction of a complete or valid IMPDU. One error condition occurs if the RSM receives a BOM DMPDU with matching VCI/MID pair and matching DA field before receiving the EOM DMPDU (Transition 22b). The other error condition occurs if the RIT associated with the VCI/MID pair expires before arrival of the EOM DMPDU (Transition 21b).

Transition 21a

Whenever the RSM associated with a VCI/MID pair is in the Active state and receives a DMPDU with matching VCI/MID that has a valid Payload_CRC subfield, contains an EOM Segment_Type subfield, and contains a value of the Sequence_Number subfield that matches the current value of RX_SEQUENCE_-NUM, then the RSM shall

a) Append the first N octets (where N is the value of the Payload_Length subfield in the DMPDU trailer) of the EOM segmentation unit to the previously accumulated BOM segmentation unit and COM segmentation unit(s) for the VCI/MID pair of the RSM.
b) Pass the reassembled IMPDU to the validation process specified in 5.1.1.2.4.
c) Clear the Reassembly IMPDU Timer (RIT) associated with the VCI/MID pair of the RSM.
d) Enter the Idle state.

If the value of the IG_FLAG is INDIVIDUAL, then the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF.

NOTE—As per 5.1.1.2.2b), Condition C, if the node is not supporting single-bit error correction of the DMPDU header via the Payload_CRC subfield in the DMPDU trailer, then the MCF block shall assert the PSR_x_SIGNAL if the value of the IG_FLAG is INDIVIDUAL, and if the MID match is made and the Payload_CRC fails.

Transition 21b

Whenever the Reassembly IMPDU Timer (RIT) associated with the VCI/MID pair of the RSM expires, then the RSM shall

a) Discard all accumulated segmentation units for the VCI/MID pair of the RSM.
b) Enter the Idle state.

Transition 21c

Whenever the RSM associated with a VCI/MID pair is in the Active state and receives a DMPDU with matching VCI/MID that has a valid Payload_CRC subfield, contains a COM or EOM Segment_Type sub-field, and contains a value of the Sequence_Number subfield that does not match the current value of RX_SEQUENCE_NUM, then the RSM shall

a) Discard all accumulated segmentation units for the VCI/MID pair of the RSM.
b) Enter the Idle state.

If the value of the IG_FLAG is INDIVIDUAL, then the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF.

NOTE—As per 5.1.1.2.2b), Condition C, if the node is not supporting single-bit error correction of the DMPDU header via the Payload_CRC subfield in the DMPDU trailer, then the MCF block shall assert the PSR_x_SIGNAL if the value of the IG_FLAG is INDIVIDUAL, and if the MID match is made and the Payload_CRC fails.

Transition 22a

Whenever the RSM associated with a VCI/MID pair is in the Active state and receives a DMPDU with matching VCI/MID that has a valid Payload_CRC subfield, contains a COM Segment_Type subfield, and contains a value of the Sequence_Number subfield that matches the current value of RX_SEQUENCE_NUM, then the RSM shall append the COM segmentation unit to the previously accumulated BOM segmentation unit and COM segmentation unit(s) for the VCI/MID pair of the RSM, while maintaining the order of received segmentation units. The RSM shall increment the value of RX_SEQUENCE_NUM by 1 (modulo 16).

If the value of the IG_FLAG is INDIVIDUAL, then the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF.

NOTE—As per 5.1.1.2.2b), Condition C, if the node is not supporting single-bit error correction of the DMPDU header via the Payload_CRC subfield in the DMPDU trailer, then the MCF block shall assert the PSR_x_SIGNAL if the value of the IG_FLAG is INDIVIDUAL, and if the MID match is made and the Payload_CRC fails.

Transition 22b

Whenever the RSM associated with a VCI/MID pair is in the Active state and receives a DMPDU with matching VCI/MID that has a valid Payload_CRC subfield, contains a BOM Segment_Type subfield, and contains a value in the Destination Address (DA) field of the MCP Header that matches an MSAP address which the node is programmed to receive, then it shall

a) Discard all accumulated segmentation units for the VCI/MID pair of the RSM.
b) Initialize and restart the Reassembly IMPDU Timer (RIT) associated with the VCI/MID pair of the RSM.
c) Store the BOM segmentation unit.
d) Initialize the RX_SEQUENCE_NUM to (the value of the Sequence_Number subfield plus one) (modulo 16).

If the value of the DA field is an individual address, the RSM shall set the IG_FLAG to INDIVIDUAL and the MCF block shall assert the PSR_x_SIGNAL for the Bus x (x = A or B) which equals the value of RX_BUS_SIGNAL that was asserted when the DMPDU was received by the MCF block. Otherwise the RSM shall set the IG_FLAG to GROUP.

If the IG_FLAG is set to INDIVIDUAL, the MCF block shall also assert the PSR_x_SIGNAL upon receipt of the COM DMPDUs and EOM DMPDU derived from the same IMPDU, as occurs during Transitions 22a and 21a, respectively. If Transition 22b occurs again, then the value of IG_FLAG may be changed.

NOTE—As per 5.1.1.2.2b), Condition A, the MCF block shall assert the PSR_x_SIGNAL if the DA match is made, if the value of the DA is an individual MSAP address and if the Payload_CRC fails.

8.3 Segment Header Check Sequence (HCS) processing

The 8-bit HCS sent in QA segment headers (see 6.3.1.1.4) and PA segment headers (see 6.4.1.1.4) provides the following capabilities:

a) Multiple-bit error detection
b) Single-bit error correction

Error detection using the HCS is mandatory at a node for QA segment header validation (5.1.2.2.3) or PA segment header validation (5.2.2.1.2 and 5.2.2.2.2). Error correction using the HCS is optional at a node. If a node supports error correction, then the functional representation of a decoder is as shown in figure 8-3. A state diagram for control of the decoder is shown in figure 8-4.

The decoder can be in one of two states: Error Correction (EC) or Error Detection (ED).[55]

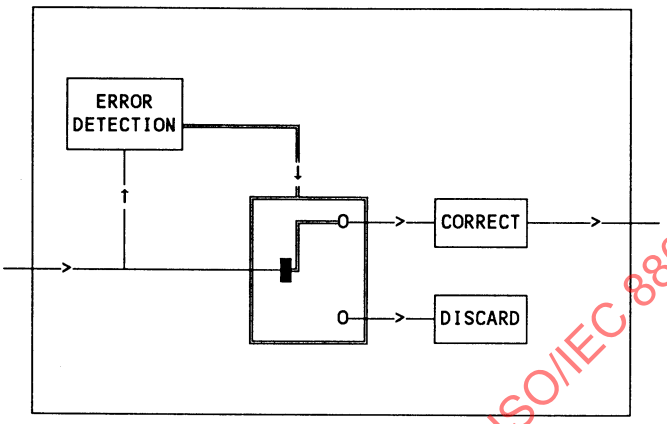


Figure 8-3—Functional diagram for HCS decoder

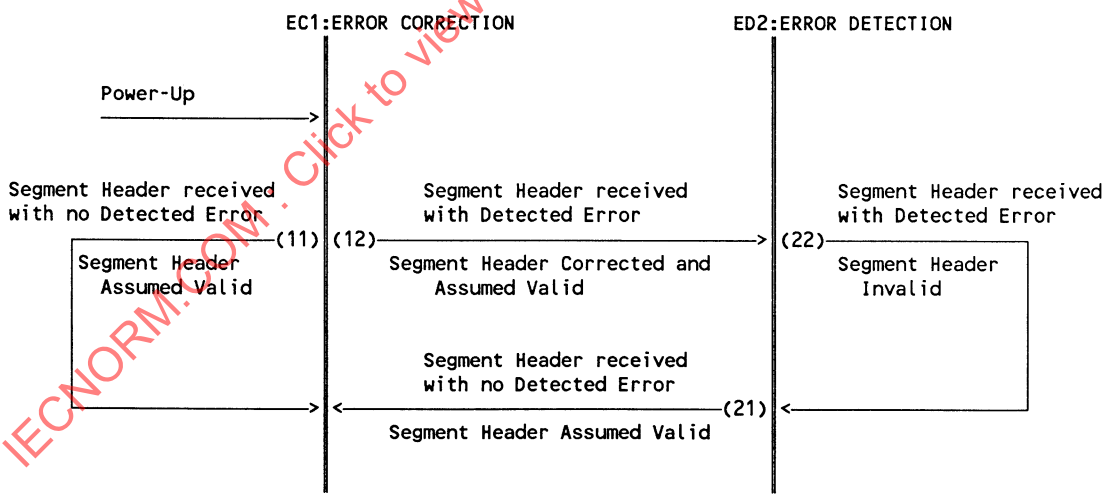


Figure 8-4—HCS Decoder State Machine

[55]The decoder still supports an error detection capability in the Error Correction state.

Operation of the HCS Decoder

State EC1: Error Correction

The decoder is initially set to be in the Error Correction state, which allows it to correct errors in the current segment header. When the decoder is in the Error Correction state, there is a probability that multiple-bit errors will be corrected as if they were single-bit errors. In such a case, since correction will be invoked, no error indication will be given and the HCS will be passed as valid.

Transition 11

If no errors are detected or corrected in the current segment header, the segment header is assumed to be valid and the decoder remains in the Error Correction state.

Transition 12

When a segment header is received with an error, the segment header is corrected, the corrected segment header is assumed to be valid and the decoder changes to the Error Detection state.

State ED2: Error Detection

While in the Error Detection state, the decoder may still act to detect segment header errors, but shall not correct errors.

Transition 21

If no errors are detected in the current segment header, the segment header is assumed to be valid and the decoder changes to the Error Correction state.

Transition 22

If an error is detected in the current segment header, the decoder remains in the Error Detection state. The errored segment header is invalid, according to a) of 5.1.2.2.3, 5.2.2.1.2, and 5.2.2.2.2.

9. DQDB Layer Management Interface (LMI)

Management is related to activities that control or monitor the use of resources. Management of the DQDB subsystem at a node involves three complementary processes:

a) *Local management within the same node.* The flow of control and data from people and/or software acting as administrative agents local to a management process occurs entirely within the local node environment and, as such, *is outside the scope of OSI management standardization.*

b) *Remote management via DQDB Layer Management,* which provides mechanisms for monitoring, control, and coordination of those managed objects used to accomplish communication activities within the DQDB Layer. DQDB Layer Management is required to enable management of the DQDB Layer when the DQDB Layer service to higher layers cannot be guaranteed to be fully operable. An example of this is when the network is being initialized.

   The DQDB Layer Management Entities (LMEs) communicate with one another via the DQDB Layer Management Protocol to support the management control of these managed objects. *The DQDB Layer Management Protocol is described in clause 10.*

c) *Remote management via network management services and protocols or system management services and protocols.* Network management provides mechanisms for the monitoring, control, and coordination of all managed objects within the Physical Layer and Data Link Layer of a node. Systems management provides mechanisms for the monitoring, control, and coordination of all managed objects within open systems. Systems management is effected through application layer protocols.

   The network and/or systems management processes at a node communicate with their peer management processes at other nodes using appropriate management protocols. *The management processes at a node access the DQDB Layer at that node via the DQDB Layer Management Interface (LMI), which is defined in this clause.*

9.1 DQDB LMI model

In the context of this part of ISO/IEC 8802, the model is used to specify the communication between the DQDB LME and the local node's Network Management Process (NMP) via the local DQDB LMI.[56]

In order to perform remote management of DQDB subnetwork node components using the model it is necessary to specify the following:

a) The services and protocols used to communicate management commands and information between the managing open system and the managed open system
b) The behavior of the node components that will be the subject of management, in terms of the operations that may be performed upon them and the way they affect operation of the node components
c) The nature of any particular protocol encodings that are required to express how the protocol is used to manage particular node components

Two general-purpose services/protocols available to satisfy a) for DQDB subnetwork management are described in ISO/IEC 9595 (CMIS) and ISO/IEC 9596 (CMIP). However, no management protocol can be used in isolation, as it is necessary to specify the elements described in b) and c) above.

[56]It should be noted that Layer Management is the subject of ongoing study and resolution within ISO/IEC JTC1, IEEE 802, and other standards organizations.

The first of these, b), may be defined in a manner that is independent of the management protocol that will be used. This description takes the form of statements about the management view of the component, and is described in terms of the following:

— Managed objects upon which operations may be performed
— The attributes (or other managed objects) that they contain
— How the definition of the managed objects relates to the behavior of the protocol machine described in this part of ISO/IEC 8802

The second of these, c), describes how the specific facilities that are available in a particular management service/protocol standard are used in order to manage the objects described in b). This will involve specification of the following:

— Particular encodings used to convey object or attribute identifiers and their values.
— The relationship between the operations described in b) and the service provided by the protocol, including any encoding requirements.

9.1.1 DQDB LMI primitives

The details of b) above are contained in the remainder of this clause. The information is represented using a set of generic LMI primitives.

The generic LMI provides the following primitives:

LM-SET-VALUE invoke
LM-SET-VALUE reply
LM-COMPARE-AND-SET invoke
LM-COMPARE-AND-SET reply
LM-GET-VALUE invoke
LM-GET-VALUE reply
LM-ACTION invoke
LM-ACTION reply
LM-EVENT notify

From the point of view of the user of the DQDB LMI service, primitives are of three types:

INVOKE	The invoke primitive is passed from the NMP to the DQDB LME to ask for the provision of a service via some DQDB Layer Management operation.
REPLY	The reply primitive is passed from the DQDB LME to the NMP to convey the result of a previous service invocation.
NOTIFY	The notify primitive is passed from the DQDB LME to the NMP to notify it of the occurrence of an event in the DQDB Layer.

The SET operation is used to set a parameter to a given value.

The COMPARE_AND_SET operation is used to set a parameter to a given value, provided that a designated test parameter is already set to match a given test value.

The GET operation is used to get the value of a given parameter.

The ACTION operation is used to force the DQDB Layer protocol machine to an identified state or to cause the initiation of a sequence of events.

The EVENT operation is not directly triggered by a request from the NMP; it is initiated locally by the DQDB LME to report the occurrence of an event in the DQDB Layer.

The primitives defined represent interactions across the DQDB LMI to the entire DQDB Layer subsystem in a particular node. Thus, the information conveyed in a primitive is accessible to any function in the DQDB Layer that requires it. A diagrammatic summary of the primitives defined in this clause and their relationship to each of the DQDB Layer functions is contained in 9.10 at the end of this clause. The management functions defined in this clause make reference to other internal functions of a DQDB node described in clause 5, and to the DQDB Layer facilities defined in clause 7, and indicate how the functions and facilities are managed.

It should be noted that the DQDB LMI primitives described in this clause currently represent a minimal set required by a node in order to support all of the services and functions offered by the DQDB Layer.

9.1.1.1 General LM-ACTION reply ( )

Many of the LMACTION invoke primitives defined in this clause lead to a general LM-ACTION reply ( ). The general LM-ACTION reply is defined as follows.

Function:

This primitive indicates the success or failure of an LMACTION invoke ( ).

Semantics of the Service Primitive:

```
LM-ACTION reply (
                    status,
                    [reason]
                    )
```

The value of status is either SUCCESSFUL or UNSUCCESSFUL. If the value of status is UNSUCCESSFUL, the reason parameter indicates the cause.

When Generated:

This primitive is generated after the local DQDB Layer subsystem has attempted to perform the action invoked by a previous LMACTION invoke.

Effect on Receipt:

The effect of receipt of this primitive by the NMP is unspecified.

Additional Comments:

None.

9.2 Virtual Channel Identifier (VCI) Management functions

Convergence functions are used to adapt the service provided by the access control functions (either Queued Arbitrated or Pre-Arbitrated) to the service defined as the DQDB Layer service. The objects and operations defined in 9.2.1 to 9.2.5 represent interactions to manage the relationship between a convergence function and its two associated functions, namely the user of the service provided by the convergence function and the associated Queued Arbitrated or Pre-Arbitrated Access function.

If an implementation is to support connectionless VCIs other than the default connectionless VCI for MAC service to LLC, then it needs to support the managed objects and operations described in 9.2.1 and 9.2.2.

If an implementation is to support isochronous service, then it needs to support the managed objects and operations described in 9.2.3 and 9.2.5.

If an implementation is to support connection-oriented data service, then it needs to support the managed objects and operations described in 9.2.4 and 9.2.5.

The Slot Marking function in the Head of Bus function needs to be informed of the requirements for generating PA slots. The primitives defined in 9.2.6 and 9.2.7 represent interactions to inform the Slot Marking function of these requirements. If an implementation is to support the Head of Bus functions, then it needs to support the managed objects and operations described in these clauses.

9.2.1 LM-ACTION invoke (CL_VCI_ADD)

The requirements for managing a connectionless convergence function differ from those required for a Connection-Oriented Convergence Function (COCF). By way of example, the connectionless function needs to perform bus selection and access_queue selection on a per frame (Initial MAC Protocol Data Unit (IMPDU) in the case of the MCF) basis. It also needs to generate Previous Segment Received bit control signals on a per segment basis, if this function is implemented by the node.

Therefore, the operations for enabling communication over a new connectionless VCI from the Queued Arbitrated Functions block are separated from those for performing the same functions for connection-oriented services. They are, however, essentially the same set of operations, with certain parameters of the OPEN_CE_COCF action set to default values for the CL_VCI_ADD action, namely tx_bus, rx_bus (set to both Bus A and Bus B), and tx_access_queue_priority (any value in the range 0–2). Connectionless VCIs may be multi-point (shared) VCIs, so that the connectionless convergence function may transmit or receive or both for any VCI, provided that it has been directed to do so by a CL_VCI_ADD action. Also, connectionless convergence functions are referred to by name, whereas COCFs are referred to via the connection end-point identifier.

Managed Object:

Relationship between Queued Arbitrated Functions block and the identified connectionless convergence function.

Function:

This primitive instructs the Queued Arbitrated function to enable communications using the specified connectionless VCI and associates the VCI with a particular convergence function (such as the MCF). It also identifies a mapping criterion that is used by the connectionless convergence function to determine when to use the VCI for the transfer of data units.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   CL_VCI_ADD,
                   convergence_function_type,
                   vci,
                   type,
                   [mapping_criteria]
                   )
```

The convergence_function_type identifies the convergence function[57] that is allowed to use the specified connectionless VCI. The type parameter specifies whether the VCI is to be used for transmission or reception. The values that can be associated with the type parameter are:

TRANSMIT, or
RECEIVE

If the type is TRANSMIT, then the mapping_criteria parameter identifies the criteria that are used by the convergence function to determine when the VCI is to be used for transfer of data. (For example, the mapping criteria could specify that the VCI be used for all data transfers, for the transfer of data to a particular set of destination addresses, for transfers when a particular quality of service is required, etc.)

When Generated:

This primitive is generated to enable communications using a specified connectionless VCI.

Effect on Receipt:

The Queued Arbitrated functions block and specified convergence function enable communications using the identified VCI. If the specified convergence function is the MCF block, then the MCF block shall initialize a new TX_SEQUENCE_NUM counter to zero for the combination of VCI and each MID value which is represented by the MID page values in the MID page list.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

A given VCI must be uniquely associated with one connectionless convergence function at the node. The VCI may be used for both transmission and reception, for transmission only, or for reception only. If the VCI is to be used both for transmission and reception, then this compound action is represented by two LM-ACTION invoke primitives for the VCI, one with TRANSMIT type and the other with RECEIVE type.

A convergence function may have more than one connectionless VCI enabled for transmission or reception or both. However, there must be different mapping criteria for each transmit VCI, such that a unique VCI can be selected for each request to transfer a convergence function service data unit. For example, the mapping criteria established for the MCF must allow it to select a unique VCI for transfer of a MAC Service Data Unit (MSDU) received in an MAUNITDATA request. There is no corresponding uniqueness requirement for the receive VCIs.

9.2.2 LM-ACTION invoke (CL_VCI_DELETE)

Managed Object:

Relationship between Queued Arbitrated Functions block and the identified connectionless convergence function.

Function:

This primitive instructs the Queued Arbitrated function to disassociate a VCI from a particular convergence function (such as the MCF) and to cease communications using the specified connectionless VCI.

[57] It is expected that, in the future, other connectionless convergence functions will be defined in addition to the MCF.

Semantics of the Service Primitive:

LM-ACTION invoke (
CL_VCI_DELETE,
vci,
type
)

The vci parameter specifies the connectionless VCI that is being removed from use by the node. The type parameter specifies whether the VCI was being used for transmission or reception. The values that can be associated with the type parameter are as follows:

TRANSMIT, or
RECEIVE

When Generated:

This primitive is generated to disable communications using a specified connectionless VCI.

Effect on Receipt:

The Queued Arbitrated Functions block and specified convergence function cease communications using the identified VCI.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

If the VCI was being used for both transmission and reception, and is to no longer be used for either, then this compound action is represented by two LM-ACTION invoke primitives for the VCI, one with TRANSMIT type and the other with RECEIVE type.

If the mapping criteria for a transmit VCI are to be modified, then this compound action is represented by an LMACTION invoke (CL_VCI_DELETE) for the VCI, followed by an LM-ACTION invoke (CL_VCI_ADD) with the new mapping criteria.

9.2.3 LM-ACTION invoke (OPEN_CE_ICF)

Note that there is always one Isochronous Convergence Function (ICF) per Isochronous Service User (ISU). Therefore for each ISU, there is a single connection end-point which is represented at the ICF to ISU boundary. This is mapped to the relationship between the Pre-Arbitrated Functions block and the ICF. Note also that the ICF could be performing a null function.

Managed Object:

The identified connection end-point between an ICF and an ISU, the relationship between the ICF and the Pre-Arbitrated Functions block for that connection end-point and the list of bus and [VCI,offset] pairs used by the Pre-Arbitrated Functions block to transmit and receive isochronous service octets for the node.

Function:

This primitive informs the Pre-Arbitrated function and the ICF of a new isochronous connection, and associates a connection end-point between the ICF and an ISU with the parameters needed to communicate on the connection.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   OPEN_CE_ICF,
                   cep_id,
                   TRANSMIT        (
                                   tx_bus,
                                   [tx_vci,offset],
                                   [tx_vci,offset],
                                   ...
                                   ),
                   RECEIVE         (
                                   rx_bus,
                                   [rx_vci,offset],
                                   [rx_vci,offset],
                                   ...
                                   )
                   )
```

The cep_id parameter identifies the connection end-point for the opened connection. The set of TRANSMIT parameters include tx_bus and the [tx_vci,offset] pairs which specify the bus and the VCI and octet offset used to send isochronous service octets. The set of RECEIVE parameters include rx_bus and the [rx_vci,offset] pairs which specify the bus and the VCI and octet offset used to accept isochronous service octets. The values that can be associated with either tx_bus or rx_bus are as follows:

BUS_A, or
BUS_B

Each offset parameter represents the octet position of the isochronous service octet within the PA segment payload and is an integer in the range 1 to 48.

When Generated:

This primitive is generated when an isochronous connection is opened.

Effect on Receipt:

The DQDB Layer functions associate the connection end-point with the specified parameters and enable information flow on the opened connection.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.2.4 LM-ACTION invoke (OPEN_CE_COCF)

Managed Object:

The identified connection end-point between a Connection-Oriented Convergence Function (COCF) and a Connection-Oriented Data Service User, the relationship between the COCF and the Queued Arbitrated Functions block for that connection end-point and the list of bus and VCIs used by the Queued Arbitrated Functions block to transmit and receive QA segments for COCFs at the node.

Function:

This primitive informs the Queued Arbitrated function and the COCF of a new connection, and associates a connection end-point between the COCF and a Connection-Oriented Data Service User with the parameters needed to communicate on the connection.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   OPEN_CE_COCF,
                   cep_id,
                   tx_bus,
                   tx_vci,
                   rx_bus,
                   rx_vci,
                   tx_segment_priority,
                   tx_access_queue_priority
                   )
```

The cep_id parameter identifies the connection end-point for the opened connection. The tx_bus and tx_vci parameters identify the bus and VCI to be used to send data for this connection. The rx_bus and rx_vci parameters identify the bus and VCI to be used to receive data for this connection. The values that can be associated with either tx_bus or rx_bus are as follows:

BUS_A, or
BUS_B

The tx_segment_priority parameter indicates the value of segment_priority to be used for each transfer over this connection. A single distributed queue priority, as specified by the tx_access_queue_priority parameter, is used for each data transfer over this connection.

NOTE—The use of a single access queue priority prevents segment misordering for the connection, which would result if multiple access priorities were allowed.

When Generated:

This primitive is generated when a connection to support the connection-oriented data service is opened.

Effect on Receipt:

The DQDB Layer functions associate the connection end-point with the specified parameters and enable information flow on the opened connection.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.2.5 LM-ACTION invoke (CLOSE_CE)

Managed Object:

The identified connection end-point between a DQDB Layer Convergence function and the user of the service provided by that convergence function.

Function:

This primitive instructs a convergence function (e.g., the ICF or COCF) to disassociate a connection end-point from a VCI as a result of a connection being closed.

Semantics of the Service Primitive:

```
LM-ACTION invoke  (
                  CLOSE_CE,
                  cep_id
                  )
```

The cep_id parameter identifies the connection end-point associated with the connection that has been closed.

When Generated:

This primitive is generated when an isochronous connection or connection-oriented data service connection is closed.

Effect on Receipt:

The convergence function that receives this primitive disassociates the specified connection end-point from the VCI previously assigned to it.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.2.6 LM-ACTION invoke (PA_VCI_ADD_HOB)

Managed Object:

The list of Pre-Arbitrated VCIs and associated attributes that the Slot Marking function uses to mark slots at the head of a bus.

Function:

This primitive directs the Slot Marking function in the node with an active Head of Bus function to mark slots as Pre-Arbitrated (PA) in accordance with the given parameters.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   PA_VCI_ADD_HOB,
                   VCI,
                   bus,
                   frequency,
                   variability
                   )
```

The VCI parameter specifies the value of the VCI field to be inserted in the PA segment header at the head of the specified bus (the default values of the Payload_Type and Segment_Priority fields of a PA segment header are both 00). The value of the bus parameter is either Bus A or Bus B. The frequency parameter specifies the average rate required for generation of PA slots with this VCI. The variability specifies the maximum variation allowed in the frequency of generation to enable the ICF to support the isochronous service.

When Generated:

This primitive is generated when the Network Management Process (NMP) in the node containing the active Head of Bus function has been instructed to start generating PA slots containing the specified VCI.

Effect on Receipt:

Receipt of this primitive causes the Slot Marking function in the node containing the active Head of Bus function to start generating PA slots in accordance with the parameters given.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

The effect of the variability parameter on the operation of the ICF is to be defined.

9.2.7 LM-ACTION invoke (PA_VCI_DELETE_HOB)

Managed Object:

The list of Pre-Arbitrated VCIs and associated attributes that the Slot Marking function uses to mark slots at the head of a bus.

Function:

This primitive directs the Slot Marking function in the node with an active Head of Bus function to cease generating Pre-Arbitrated (PA) slots for the specified VCI.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   PA_VCI_DELETE_HOB,
                   VCI,
                   bus
                   )
```

The VCI parameter specifies the VCI for which PA slots on the specified bus should no longer be generated.

When Generated:

This primitive is generated when the NMP in the node containing the active Head of Bus function has been instructed to cease generating PA slots for a specified VCI.

Effect on Receipt:

Receipt of this primitive causes the Slot Marking function in the node containing the active Head of Bus function to cease generating PA slots on the specified bus with the specified VCI.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.3 Header Extension Management functions

When the Logical Link Control (LLC) Sublayer requests the transfer of a MAC Service Data Unit (MSDU) via an MA-UNITDATA request, the MAC Convergence Function (MCF) must determine whether the Initial MAC Protocol Data Unit (IMPDU) used to transfer the MSDU also needs to carry a Header Extension field. The primitives defined in this clause represent interactions to manage the information required by all connectionless convergence functions to determine what information, if any, is carried in the Header Extension field of each IMPDU.

9.3.1 LM-ACTION invoke (HEXT_INSTAL)

Managed Object:

The list of header extension values and selection information for the identified convergence function.

Function:

This primitive installs a new header extension value at a connectionless convergence function and provides information to the convergence function to allow it to determine when to use this value for a given data transfer.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   HEXT_INSTAL,
                   convergence_function_type,
                   HE_value,
                   HE_selection_info
                   )
```

The HE_value parameter specifies the header extension value to be installed at the identified convergence function. The HE_selection_info parameter specifies the conditions under which the header extension value is to be used.

When Generated:

This primitive is generated when a new header extension is to be made available for use by a connectionless convergence function.

Effect on Receipt:

The named convergence function adds this header extension value to the set of values available for use when a data transfer is requested of the convergence function and uses the value when a request is made that matches the selection information.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

A convergence function may have more than one header extension value available for use. However, there must be different selection information for each value, such that a unique header extension can be selected for each re-quest to transfer a convergence function service data unit. For example, the selection information established for the MCF must allow it to select a unique header extension value for transfer of a MSDU received in an MA-UNITDATA request.

If none of the sets of selection information match the parameters received in the request to transfer the convergence function service data unit, then the convergence function inserts a header extension of zero length.

9.3.2 LM-ACTION invoke (HEXT_PURGE)

Managed Object:

The list of header extension values and selection information for the identified convergence function.

Function:

This primitive purges a header extension value from the set available for use at a connectionless convergence function.

Semantics of the Service Primitive:

```
LM-ACTION.invoke   (
                   HEXT_PURGE,
                   convergence_function_type,
                   HE_value
                   )
```

The HE_value parameter specifies the header extension value to be purged from the set available at the identified convergence function.

When Generated:

This primitive is generated when a header extension is no longer to be used by a connectionless convergence function.

Effect on Receipt:

The identified convergence function deletes this header extension value from the set of values available for use.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

If the selection information for a header extension value is to be modified, then this compound action is represented by an LM-ACTION invoke (HEXT_PURGE) for the header extension value, followed by an LM-ACTION invoke (HEXT_INSTAL) with the new selection information.

9.4 Message Identifier (MID) Management functions

The operations defined in this clause represent interactions to manage the Message Identifiers (MIDs) that are used by the MAC Convergence Function (MCF) block. All nodes conforming to this part of ISO/IEC 8802 shall support the managed objects and operations described in this clause.

The strategy used for obtaining and releasing MIDs is a matter for implementation decision. One possible strategy is for the node to hold a pool of MID pages. The size of this pool could be adjusted as needed, provided that the node had not exceeded its maximum allowed allocation of MID pages. End user nodes would normally only require one MID page.

Irrespective of the local MID management strategy, this part of ISO/IEC 8802 uses a management model in which knowledge of the MIDs allocated to a particular node is held in a list. This list is represented as part of the local Management Information Base under control of the local Network Management Process (NMP). This means that when, according to the implementation, the node decides it needs a MID page, the Management Information Base, which is a conceptual composite of the management information within the node, requests the DQDB Layer to get an MID page. When a page is allocated, the list of MID pages available is modified and the MCF block is informed of this. Similarly, when the implementation determines that the node should release a MID page, the Management Information Base requests the DQDB Layer to delete the value from the list of available MID pages and inform the MCF block.

9.4.1 LM-ACTION invoke (MID_PAGE_GET)

Managed Object:

The list of MID pages available for use at a node.

Function:

This primitive directs the DQDB Layer to get a MID page.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   MID_PAGE_GET
                   )
```

When Generated:

This primitive is generated when a MID page must be obtained.

Effect on Receipt:

The receipt of this primitive causes the DQDB Layer Management Entity (LME) to try to obtain the allocation of a MID page via the MID Page Allocation function of the Common Functions block. (See 5.4.4.2.4.)

Additional Comments:

The strategy for obtaining MID pages (e.g., obtaining all MIDs at system start-up, obtaining MIDs as needed, etc.) is outside the scope of this part of ISO/IEC 8802.

9.4.2 LM-ACTION reply (MID_PAGE_GET)

Function:

This primitive indicates whether or not the DQDB Layer has obtained a MID page, as requested by an LM-ACTION invoke (MID_PAGE_GET).

Semantics of the Service Primitive:

```
LM-ACTION reply  (
                 MID_PAGE_GET,
                 status,
                 [mid_page_id],
                 [reason]
                 )
```

The value of the status is either SUCCESSFUL or UNSUCCESSFUL. If the value of status is SUCCESSFUL, the mid_page_id parameter specifies which MID page has been allocated. If the value of status is UNSUCCESSFUL, the reason parameter indicates the cause.

When Generated:

This primitive is generated after the DQDB LME has tried to obtain a MID page, as requested by receipt of an LMACTION invoke (MID_PAGE_GET). If the status is SUCCESSFUL, it

a) Provides notification that the appropriate change has been made to the list of MID pages, as required by 5.4.4.2.1, and
b) Indicates that the MCF has been informed that it may use the MID page.

Effect on Receipt:

This primitive informs the NMP of the response of the DQDB Layer.

Additional Comments:

None.

9.4.3 LM-ACTION invoke (MID_PAGE_RELEASE)

Managed Object:

The list of MID pages available for use at a node.

Function:

This primitive directs the DQDB Layer to release a MID page.

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   MID_PAGE_RELEASE,
                   mid_page_id
                   )
```

The mid_page_id parameter identifies the MID page to be released.

When Generated:

This primitive is generated when a MID page must be released.

Effect on Receipt:

The receipt of this primitive causes the DQDB LME to release the specified MID page via the MID Page Allocation function of the Common Functions block. (See 5.4.4.2.1.)

Additional Comments:

The strategy for releasing MID pages (e.g., after the MID page has been used for a certain time, when the number of inactive MIDs at a node exceeds a threshold, etc.) is outside the scope of this part of ISO/IEC 8802.

9.4.4 LM-ACTION reply (MID_PAGE_RELEASE)

Function:

This primitive indicates whether or not the DQDB Layer has released a MID page, as requested by an LMACTION invoke (MID_PAGE_RELEASE) primitive.

Semantics of the Service Primitive:

```
LM-ACTION reply    (
                   MID_PAGE_RELEASE,
                   status,
                   [mid_page_id]
                   [reason]
                   )
```

The value of the status is either SUCCESSFUL or UNSUCCESSFUL. If the value of the status is SUCCESSFUL, the mid_page_id parameter specifies which MID page was released. If the value of status is UNSUCCESSFUL, the reason parameter indicates the cause.

When Generated:

This primitive is generated after the DQDB LME has tried to release a MID page, as requested by receipt of an LM-ACTION invoke (MID_PAGE_RELEASE). If the status is SUCCESSFUL, it

a) Provides notification that the appropriate change has been made to the list of MID pages, as required by 5.4.4.2.1, and
b) Indicates that the MCF has been informed that it may no longer use the MID page.

Effect on Receipt:

This primitive informs the NMP of the response of the DQDB Layer.

Additional Comments:

None.

9.4.5 LM-EVENT notify (MID_PAGE_LOST)

Managed Object:

The list of MID pages available for use at a node.

Function:

This primitive notifies the NMP that the allocation of a MID page has been lost.

Semantics of the Service Primitive:

```
LM-EVENT notify    (
                   MID_PAGE_LOST,
                   mid_page_id,
                   reason
                   )
```

The mid_page_id parameter specifies which MID page has been lost. The reason parameter indicates the reason why allocation of the MID page was lost, such as contention in the MID Page Allocation protocol.

When Generated:

This primitive is generated when allocation of a MID page is lost. It does the following:

a) Provides notification that the appropriate change has been made to the list of MID pages, as required by 5.4.4.2.1, and
b) Indicates that the MCF has been informed that it may no longer use the MID page.

Effect on Receipt:

The receipt of this primitive informs the NMP why the allocation of a MID page has been lost.

Additional Comments:

The notification of this event does not require the MAC Convergence Function block to abort the sending of an IMPDU that has already been segmented and for which the BOM DMPDU has been sent using one of the MID values included in the MID page value that has been lost.

9.5 Address Management functions

If an implementation is to support the recognition of MSAP addresses other than the 48-bit, universally administered address, then it needs to support the managed objects and operations described in this clause.

9.5.1 LM-ACTION invoke (ADDRESS_ADD)

Managed Object:

The list of addresses the node is required to recognize in the Destination Address (DA) field of IMPDUs.

Function:

This primitive instructs the DQDB Layer in a node to add an address to the set of those recognized. The address may be an individual or multicast address of one of the three valid address types (i.e., 16 bit, 48 bit, or 60 bit).

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   ADDRESS_ADD,
                   address_type,
                   address
                   )
```

The address_type parameter specifies the type of the address parameter. The address parameter specifies the address to be added to the set of those recognized.

When Generated:

This primitive is generated to instruct the DQDB Layer in a node to recognize an address. Since a node may recognize more than one address, this primitive may be generated several times to inform the node of several addresses to be recognized.

Effect on Receipt:

The DQDB Layer subsystem in the node will add the specified address to the set of recognized addresses.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.5.2 LM-ACTION invoke (ADDRESS_DELETE)

Managed Object:

The list of addresses the node is required to recognize in the DA field of IMPDUs.

Function:

This primitive instructs the DQDB Layer in a node to delete an address from the set of those recognized. The address may be an individual or multicast address of one of the three valid address types (i.e., 16 bit, 48 bit, or 60 bit).

Semantics of the Service Primitive:

```
LM-ACTION invoke   (
                   ADDRESS_DELETE,
                   address_type,
                   address
                   )
```

The address_type parameter specifies the type of the address parameter. The address parameter specifies the address to be deleted from the set of those recognized.

When Generated:

This primitive is generated to instruct the DQDB Layer in a node to stop recognizing an address.

Effect on Receipt:

The DQDB Layer subsystem in the node will delete the specified address from the set of recognized addresses.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

None.

9.6 System Parameter Management functions

The operations defined in this clause represent interactions to manage the subnetwork system parameters.

If an implementation is to support setting the period of the Reassembly IMPDU Timer (see 7.1.1) to a value other than the default value, then it needs to support the RIT_PERIOD system parameter (see 7.3.1) and be able to set it using the operation described in this clause.

If an implementation is to support access to priority level queues other than the default level, then it needs to support the QOS_MAP system parameter (see 7.3.3) and be able to set it using the operation described in this clause.

An implementation must be capable of setting the BWB_MOD to values other than the default value. It must support the BWB_MOD system parameter (see 7.3.6) and be able to set it to the full range of values specified in 7.3.6 using the operations described in this clause.

If an implementation is to support the allocation of more than the default number of MID page values, then it needs to support the MAX_MID_PAGES system parameter (see 7.3.5) and be able to set it using the operation described in this clause.

If an implementation is to be able to support the Head of Bus functions using Configuration Control (CC) Type 1 or 2 functions, then it needs to support the Head of Bus Arbitration Timer (see 7.1.2), support the Timer_H_PERIOD system parameter (see 7.3.2), and be able to set it using the operation described in this clause. Timer_H_PERIOD should be set to the same value for all such nodes on the subnetwork.

If an implementation is to be able to support the Head of Bus A functions, then it needs to support the RESERVED_MID_PAGES system parameter (see 7.3.4) and be able to set it using the operation described in this clause.

9.6.1 LM-SET invoke (SYSTEM_PARAMETER)

Managed Object:

The identified system parameter as described in 7.3.

Function:

This primitive directs the DQDB LME to set the value of the identified system parameter.

Semantics of the Service Primitive:

```
LM-SET invoke (
                SYSTEM_PARAMETER,
                parameter_type,
                parameter_value
                )
```

The parameter_type parameter specifies which of the different system parameters is to be set to the specified value. The values that can be associated with the parameter_type parameter are as follows:

RIT_PERIOD, or
Timer_H_PERIOD, or
QOS_MAP, or
RESERVED_MID_PAGES, or
MAX_MID_PAGES, or
BWB_MOD

The values that can be associated with the parameter_value parameter are specified in 7.3.1 to 7.3.5, respectively.

When Generated:

This primitive is generated implicitly at node initialization to set the values of the system parameters. It may also be generated subsequently to tune the operation of the DQDB Layer functions.

Effect on Receipt:

The specified system parameter is set to the specified value.

Additional Comments:

None.

9.7 Configuration Control function management functions

The operations defined in this clause represent interactions to manage the operation of the Configuration Control functions defined in 10.2.4 to 10.2.6.

All implementations shall support the appropriate Configuration Control Flag, depending on whether the node contains the Default Configuration Control function (CC_D2_CONTROL, 7.4.2) or not (CC_12_CONTROL, 7.4.1). An implementation may not support all possible values for CC_12_CONTROL, depending on whether the node is capable or not of supporting Head of Bus functions.

9.7.1 LM-SET invoke (CC_FLAG)

Managed Object:

The identified Configuration Control Flag.

Function:

This primitive directs the DQDB LME to set the value of the identified Configuration Control Flag.

Semantics of the Service Primitive:

```
LM-SET invoke (
                CC_FLAG,
                flag_type,
                flag_value
                )
```

The flag_type parameter specifies which of the different configuration control flags is to be set to the specified value. The values that can be associated with the flag_type parameter are as follows:

CC_12_CONTROL, or
CC_D2_CONTROL

The values that can be associated with the flag_value parameter are specified in 7.4.1 and 7.4.2, respectively.

When Generated:

This primitive is generated to externally control the operation of the Configuration Control functions at a node.

Effect on Receipt:

The specified Configuration Control Flag is set to the specified value.

Additional Comments:

This management function might be used for testing purposes or by the network administrator to control the location of functions required to meet the fundamental subnetwork requirements defined in 5.4.2.1.

9.8 CRC32 Control Flag Management functions

The operations defined in this clause represent interactions to manage the CRC32 Control Flags defined in 7.4.3 and 7.4.4.

If an implementation is to be capable of supporting CRC32 generation or CRC32 checking, then it must support the ON value for the CRC32_GEN_CONTROL or CRC32_CHECK_CONTROL Flag, respectively, and be able to set the flag using the operation described in this clause.

9.8.1 LM-SET invoke (CRC32_FLAG)

Managed Object:

The identified CRC32 Control Flag.

Function:

This primitive directs the DQDB LME to set the value of the identified CRC32 Control Flag.

Semantics of the Service Primitive:

```
LM-SET invoke (
                CRC32_FLAG,
                flag_type,
                flag_value
                )
```

The flag_type parameter specifies which of the different CRC32 Control Flags is to be set to the specified value. The values that can be associated with the flag_type parameter are as follows:

CRC32_GEN_CONTROL, or
CRC32_CHECK_CONTROL

The values that can be associated with the flag_value parameter are specified in 7.4.3 and 7.4.4, respectively.

When Generated:

This primitive is generated to externally control the operation of the CRC32 Generation or Checking functions at a node.

Effect on Receipt:

The specified CRC32 Control Flag is set to the specified value.

Additional Comments:

If it is desired to simultaneously modify the CRC32_GEN_CONTROL Flag and the CRC32_CHECK_CONTROL Flag, then this compound set function is represented by two LM-SET invoke primitives, one for each flag.

9.9 Other management functions

9.9.1 LM-ACTION invoke (RESET)

Managed Object:

All functions in the DQDB Layer.

Function:

This primitive resets all functions in the DQDB Layer of a node to the appropriate "reset" state.

Semantics of the Service Primitive:

```
LM-ACTION invoke  (
                  RESET
                  )
```

When Generated:

This primitive is generated when it is necessary to return all DQDB Layer functions in a node to a known "reset" state.

Effect on Receipt:

All DQDB Layer functions are reset to the appropriate state.

An LM-ACTION reply is generated to indicate the success or failure of the invocation in the form of a status report. The general form of this reply is given in 9.1.1.1.

Additional Comments:

The "reset" state for all DQDB Layer functions is defined to be the power-up state for each function.

9.10 Summary of DQDB LMI primitives

Figure 9-1 contains a diagrammatic representation of the relationship between each DQDB Layer Management interaction and the functional elements of the DQDB Layer subsystem at a node described in clause 5.

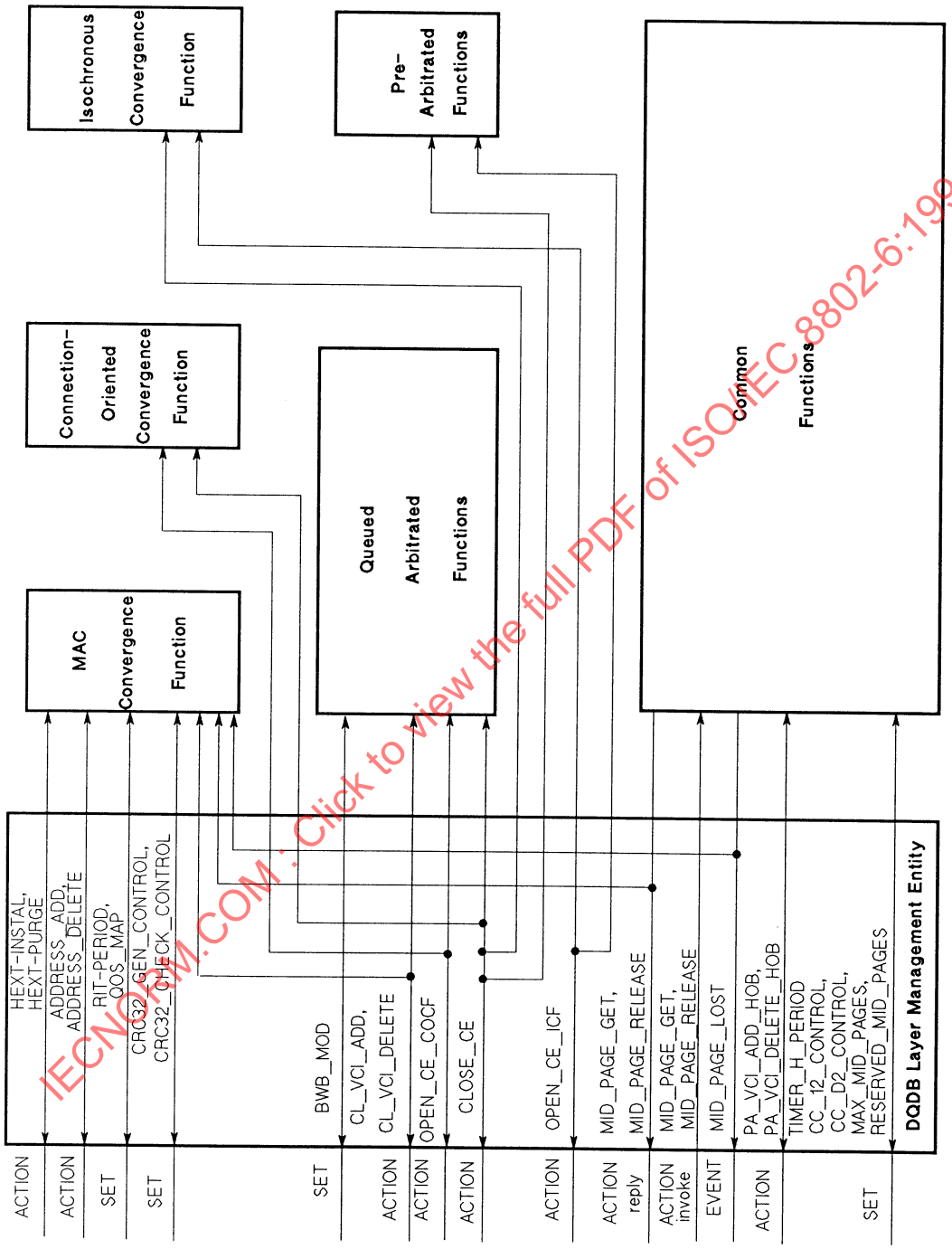


Figure 9-1—DQDB Layer Management interaction

10. DQDB Layer Management protocol

Management is related to activities that control or monitor the use of resources. Management of the DQDB subsystem at a node involves three complementary processes:

a) *Local management within the same node.* The flow of control and data from people and/or software acting as administrative agents local to a management process occurs entirely within the local node environment and, as such, *is outside the scope of OSI management standardization.*

b) *Remote management via network management services and protocols or system management services and protocols.* Network management provides mechanisms for the monitoring, control, and coordination of all managed objects within the Physical Layer and Data Link Layer of a node. Systems management provides mechanisms for the monitoring, control, and coordination of all managed objects within open systems. Systems management is effected through application layer protocols.

   The network and/or systems management processes at a node communicate with their peer management processes at other nodes using appropriate management protocols. *The management processes at a node access the DQDB Layer at that node via the DQDB Layer Management Interface, which is defined in clause 9.*

c) Remote management via DQDB Layer Management, which provides mechanisms for monitoring, control, and coordination of those managed objects used to accomplish communication activities within the DQDB Layer. DQDB Layer Management is required to enable management of the DQDB Layer when the DQDB Layer service to higher layers cannot be guaranteed to be fully operable. An example of this is when the network is being initialized.

   The DQDB Layer Management Entities (LMEs) communicate with one another via the DQDB Layer Management Protocol to support the management control of these managed objects. *The DQDB Layer Management Protocol is described in this clause.*

10.1 DQDB Layer Management Information octets

The DQDB Layer Management Information octets support communication between the DQDB Layer Management Entities in the subnetwork. The two management information octets have the format shown in figure 10-1. The length of each field in the octets is shown in bits. The Reserved bit is set to 0.

TYPE (set to 0)	Bus Identification Field (BIF)	Subnetwork Configuration Field (SNCF)
(1 bit)	(2 bits)	(5 bits)

TYPE (set to 1)	RESERVED	MID Page Allocation field
(1 bit)	(1 bit)	(6 bits)

Figure 10-1—DQDB Layer Management Information octets

The TYPE bit indicates whether the management information octet is carrying Bus Identification and Subnetwork Configuration information (TYPE = 0) or MID Page Allocation information (TYPE = 1). The format of the remaining fields are described in the subclauses of this clause.

Each management information octet is transferred between peer DQDB Layer Management Protocol Entities. (See 5.4.3.2.) The management information octets are generated and received as octets of type DQDB_MANAGEMENT in Ph-DATA request primitives and Ph-DATA indication primitives, respectively. (See 4.1 and 4.2.)

The frequency of generation of octets of type DQDB_MANAGEMENT in a subnetwork is a function of the Physical Layer Convergence Procedure (see 11.1) being operated as part of the Physical Layer for that subnetwork. In a manner consistent with the Physical Layer requirements, EMPTY octets of type DQDB_MANAGEMENT are generated at the node performing the Head of Bus function, as described in 4.2 and 4.3.2. The DQDB Layer Management Protocol Entity at a node with an active Head of Bus function shall set the TYPE bit alternately to (TYPE = 0) and (TYPE = 1) for successive DQDB_MANAGEMENT octets.

The management information octets are then operated on by the DQDB Layer Management Protocol Entity of nodes as described in 5.4.3.3 (Bus Identification Field) and 10.2 (Subnetwork Configuration field) for (TYPE = 0) octets, and as described in 10.3 (MID Page Allocation field) for (TYPE = 1) octets. As it passes downstream from the Head of Bus, the value of the TYPE bit shall not be modified by the DQDB Layer Management Protocol Entity at any node.

10.1.1 Bus Identification Field (BIF)

The BIF is used to uniquely identify each of the unidirectional buses. The BIF codes are generated for each bus by the DQDB Layer Management Protocol Entity at the node that contains the active Head of Bus function for that bus, as described in 5.4.3.3.

The BIF codes that may be generated are shown in table 10-1. The remaining code (11) is not available for use.

Table 10-1—BIF codes

BIF code	Bus
01	Bus A
10	Bus B
00	Unknown bus identification

10.1.2 Subnetwork Configuration Field (SNCF)

The SNCF is used to transfer Configuration Control information between the Configuration Control functions of all DQDB Layer Management Protocol Entities on the DQDB subnetwork. The Configuration Control function ensures that the following resources are activated into a proper Dual Bus topology:

a) Default Slot Generator function; and
b) Head of Bus A function; and
c) Head of Bus B function; and
d) Primary timing reference for the subnetwork.

The SNCF contains the three subfields shown in figure 10-2. Each subfield conveys information about the named resource and is described in the subclauses of this clause. The length of each subfield is shown in bits.

Default Slot Generator Subfield (DSGS)	Head of Bus Subfield (HOBS)	External Timing Source Subfield (ETSS)
(2 bits)	(2 bits)	(1 bit)

Figure 10-2—SNCF subfield formats

The SNCF is received at all nodes by the DQDB Layer Management Protocol Entity and made available to the Configuration Control functions at the node. An SNCF subfield value may be modified only by the DQDB Layer Management Protocol Entity at a node where the Configuration Control functions have activated the associated resource. At nodes where the Configuration Control functions have not activated the associated resource, the SNCF subfield value shall be relayed unchanged by the DQDB Layer Management Protocol Entity.

10.1.2.1 Default Slot Generator Subfield (DSGS)

The DSGS indicates whether or not there is an active Default Slot Generator function present upstream on the bus on which the DSGS was received.

The valid DSGS codes are shown in table 10-2. They are separated by a Hamming distance of two.

Table 10-2—DSGS codes

DSGS code	Active DSG upstream?
00	NOT_PRESENT
11	PRESENT

The invalid values are interpreted as NOT_PRESENT by the node that contains the Default Slot Generator function. The invalid values are not interpreted by other nodes and shall be relayed unchanged.

10.1.2.2 Head of Bus Subfield (HOBS)

The HOBS indicates whether the Head of Bus function is waiting to be activated or deactivated as a result of a change in the subnetwork configuration or is operating in a stable subnetwork configuration.

The valid HOBS codes are shown in table 10-3.

Table 10-3—HOBS codes

HOBS Code	Head of Bus Status
01	STABLE
10	WAITING
00	NO_ACTIVE_HOB

The invalid value (11) is not interpreted and shall be relayed unchanged.

10.1.2.3 External Timing Source Subfield (ETSS)

The ETSS indicates whether or not there is an active External Timing Source function present upstream on the bus on which the ETSS was received.

The valid ETSS codes are shown in table 10-4.

Table 10-4—ETSS codes

ETSS code	Active ETS upstream?
0	NOT_PRESENT
1	PRESENT

10.1.3 MID Page Allocation Field (MPAF)

The MPAF is used to transfer MID Page Allocation information between the MID Page Allocation function of all DQDB Layer Management Protocol Entities on the DQDB subnetwork. The MID Page Allocation function ensures that each node is uniquely allocated one or more MID pages.

The MPAF contains the three subfields shown in figure 10-3. Each subfield is 2 bits long.

Page reservation	Page counter modulus	Page counter control
(2 bits)	(2 bits)	(2 bits)

Figure 10-3—MPAF subfield formats

10.1.3.1 Page Reservation (PR) subfield

The PR subfield indicates whether the MID page associated with the current value of PAGE_CNTR_x (x = A or B, see 7.2.4) has been reserved or not reserved.

The valid PR codes are shown in table 10-5. They are separated by a Hamming distance of two.

Table 10-5—PR codes

PR code	MID page state
00	NOT_RESERVED
11	RESERVED

The other possible values of the PR subfield (binary 10 and 01) are ERROR values, and are interpreted according to the bus on which they are received, as follows:

— A PR value of ERROR is interpreted as NOT_RESERVED on Bus A.

— A PR value of ERROR is interpreted as RESERVED on Bus B.

10.1.3.2 Page Counter Modulus (PCM) subfield

The PCM subfield is used to check that the value of PAGE_CNTR_x (x = A or B, see 7.2.4) is synchronized to the MID page value that was associated with the PR subfield in this MPAF by the Head of Bus A function. The PCM subfield contains the modulo 4 value of the MID page value associated with the PR subfield of this MPAF.

The PCM codes are shown in table 10-6.

Table 10-6—PCM codes

PCM code	MID page value Modulo 4
00	0
01	1
10	2
11	3

10.1.3.3 Page Counter Control (PCC) subfield

The PCC subfield indicates whether PAGE_CNTR_x (x = A or B, see 7.2.4) should be incremented by 1 or reset to MIN_PAGE = 1 for use with the PR subfield in the next MPAF.

The valid PCC codes are shown in table 10-7. They are separated by a Hamming distance of two.

Table 10-7—PCC codes

PCC code	PAGE_CNTR operation
01	RESET (to MIN_PAGE = 1)
10	INCREMENT (by 1)

The other possible values of the PCC subfield (binary 00 and 11) are ERROR values, and are interpreted as INCREMENT.

10.2 Configuration Control protocol

The Configuration Control protocol is used to communicate information between the DQDB Layer Management Protocol Entities of nodes to support the operation of the Configuration Control (CC) function, described in 5.4.2. The CC function ensures that the following resources are activated into a proper Dual Bus topology:

a) Default Slot Generator function; and
b) Head of Bus A function; and
c) Head of Bus B function; and
d) Primary timing reference for the subnetwork.

The information required to support the CC protocol is carried in the subfields of the SNCF, described in 10.1.2. Additional information required to support certain elements of the CC protocol is carried in the BIF, described in 10.1.1.

In each of the figures and tables in 10.2.4, 10.2.5, and 10.2.6, which use received SNCF and BIF values as input values, the following rules apply to the SNCF subfield value(s) or BIF value used:

— The DSGS value used shall be the value of PRESENT or NOT_PRESENT, which has been received at least twice in the three most recent DSGS values that contained either PRESENT or NOT_PRESENT.
— The HOBS value used shall be the value of STABLE or WAITING, which has been received at least twice in the three most recent HOBS values that contained either STABLE or WAITING.
— The ETSS value used shall be the value of PRESENT or NOT_PRESENT, which has been received at least twice in the three most recent ETSS values that contained either PRESENT or NOT_PRESENT.
— The BIF value used shall be the value of Bus A or Bus B, which has been received at least twice in the three most recent BIF values that contained either Bus A or Bus B.

These rules apply to figures 10-7 and 10-8, and to tables 10-10a), 10-10b), 10-11, and 10-12.

In each of the tables in 10.2.4, 10.2.5, and 10.2.6, which relay an SNCF subfield value, the SNCF subfield value relayed shall be the value most recently received at the node. This rule applies to tables 10-10a), 10-10b), 10-11, and 10-12.

10.2.1 Allowed combinations of Configuration Control functions in a node

There are three types of Configuration Control (CC) function defined in this part of ISO/IEC 8802, each uniquely associated with a type of Slot Generator function. The three types of CC function are as follows:

— The Default Configuration Control function, CC_D, which is associated with the Default Slot Generator function, SG_D
— The Configuration Control function of Type 2, CC_2, which is associated with the Slot Generator Type 2 function, SG_2
— The Configuration Control function of Type 1, CC_1, which is associated with the Slot Generator Type 1 function, SG_1

This part of ISO/IEC 8802 defines two combinations of CC functions in a node, depending on whether or not the node contains the Default Slot Generator function.

Figure 10-4 shows the architecture of the Common functions block for a node containing the Default Slot Generator function. Hence, the node containing the Default Slot Generator function contains a Default Configuration Control (CC_D) function and a Configuration Control Type 2 (CC_2) function.

Figure 10-5 shows the architecture of the Common functions block for all nodes not containing the Default Slot Generator function. Hence, the nodes not containing the Default Slot Generator function contain a Configuration Control Type 1 (CC_1) function and a Configuration Control Type 2 (CC_2) function.

The Subnetwork Configuration Control function described in 5.4.2 relies on communication between all of the Configuration Control functions of all types in the subnetwork. The Configuration Control protocol described in this clause supports these communication requirements.

Subclauses 10.2.2.1 to 10.2.2.3 outline the individual resources controlled by each of the Configuration Control types. Subclause 10.2.3 describes the conditions under which the SNCF subfield values are generated and the information about subnetwork status that can be derived from the SNCF values received at the node.

Subclauses 10.2.4, 10.2.5, and 10.2.6 provide the rigorous definition of the operation of the different Configuration Control types.

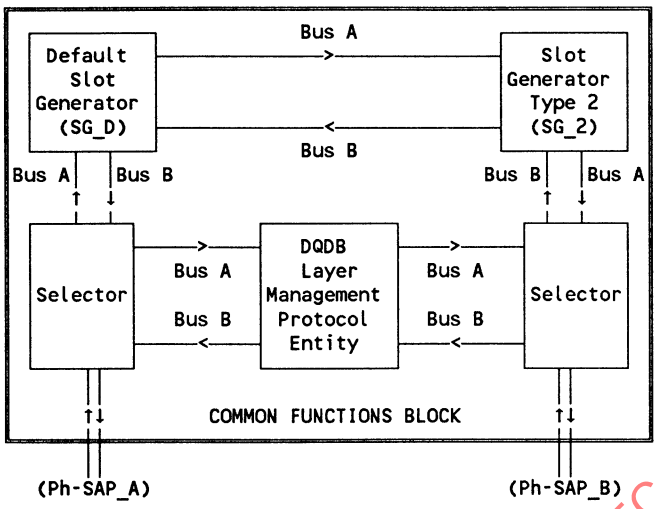


Figure 10-4—Common Functions block architecture for the node supporting the Default Slot Generator function

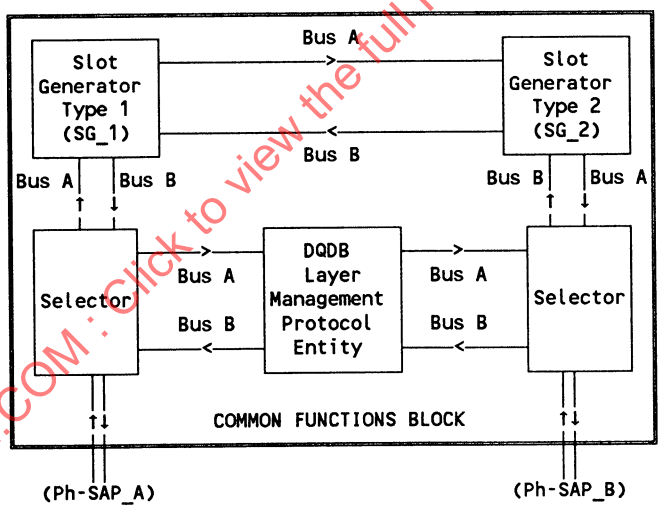


Figure 10-5—Common Functions block architecture for the nodes not supporting the Default Slot Generator function

10.2.2 Resources controlled by each Configuration Control function type

10.2.2.1 Default Configuration Control function

A CC_D function controls the activation and deactivation of the following resources, using the Default Configuration Control State Machine described in 10.2.4.

a) Default Slot Generator function
b) Head of Bus A function
c) Head of Bus B function
d) Provision of 125 µs timing for Bus A, either by
   1) Use of an external timing reference, if so directed by the network administrator
   2) Use of the node 125 µs clock
e) Provision of 125 µs timing for Bus B, either by
   1) Use of an external timing reference, if so directed by the network administrator
   2) Use of the node 125 µs clock

Item a) is required according to 5.4.2.1, Fundamental Subnetwork Requirement FR3.

Items b) and c) are performed according to 5.4.2.3.

Items d) and e) are performed according to 5.4.2.4.

10.2.2.2 Configuration Control Type 2 function

A CC_2 function controls the activation and deactivation of the following resources, using the Configuration Control Type 2 State Machine described in 10.2.5.

a) Head of Bus B function
b) Provision of 125 µs timing for Bus B by use of an external timing reference, if so directed by the network administrator

Item a) is performed according to 5.4.2.3.

Item b) is performed according to 5.4.2.4.

10.2.2.3 Configuration Control Type 1 function

A CC_1 function controls the activation and deactivation of the following resources, using the Configuration Control Type 1 State Machine described in 10.2.6.

a) Head of Bus A function
b) Provision of 125 µs timing for Bus A, either by
   1) Use of an external timing reference, if so directed by the network administrator.
   2) Use of the node 125 µs clock

Item a) is performed according to 5.4.2.3.

Item b) is required according to 5.4.2.4.

10.2.3 Generation of SNCF subfields

This clause describes the conditions under which the DQDB Layer Management Protocol Entity at a node may modify the value of each of the SNCF subfields. It also describes the information about subnetwork sta-

tus that can be derived from the SNCF values received at the node. A set of example stable subnetwork configurations is given in annex H. An example of a subnetwork during a change in configuration is given in annex I.

10.2.3.1 DSGS generation

The valid values for the DSGS are PRESENT and NOT_PRESENT. (See 10.1.2.1.) If any of the invalid values for the DSGS are received at a node that does not contain the Default Slot Generator function, the DSGS shall be relayed unchanged by the DQDB Layer Management Protocol Entity and the value of DSGS shall be ignored. The default value of the DSGS is NOT_PRESENT. The DSGS shall be set on both buses to PRESENT by the DQDB Layer Management Protocol Entity at the node that contains the active Default Slot Generator function.

The subnetwork configuration status can be derived at the node containing the Default Slot Generator function by examining the stable value of the DSGS received on both buses at the node (i.e., when the node is not receiving a HOBS value of WAITING on either bus). The values have the meaning given in table 10-8.

Table 10-8—Relationship of subnetwork topology to DSGS values received at a node containing the default slot generator

DSGS values received	Dual Bus configuration
PRESENT (both buses)	Looped
NOT_PRESENT (both buses)	Open (DSG in middle of bus)
NOT_PRESENT (end of Bus B)	Open (DSG at head of Bus A)

The subnetwork configuration status can be derived at a node not containing the Default Slot Generator function by examining the stable value of the DSGS received at the node on each bus. The values have the meaning given in table 10-9.

Table 10-9—Relationship of subnetwork topology to DSGS values received at a node not containing the default slot generator

DSGS Set to PRESENT on:	Dual Bus configuration
Both buses	Looped
One bus	Open
Neither bus	Island

10.2.3.2 HOBS generation

The valid values for the HOBS are WAITING, STABLE, and NO_ACTIVE_HOB. (See 10.1.2.2.) If the invalid value for the HOBS is received at a node, the HOBS shall be relayed unchanged by the DQDB Layer Management Protocol Entity and the value of HOBS shall be ignored.

The HOBS shall be set to STABLE on both Bus A and Bus B by the DQDB Layer Management Protocol Entity at the node containing the Default Slot Generator (SG_D) function, if the SG_D is acting as the Head of Bus A and as the Head of Bus B in a looped Dual Bus subnetwork. The HOBS shall be set to STABLE on Bus A by the DQDB Layer Management Protocol Entity at the node containing the SG_D function, if the SG_D is acting as the Head of Bus A only.

The HOBS shall be set to NO_ACTIVE_HOB on both Bus A and Bus B by the DQDB Layer Management Protocol Entity at a node containing either a Slot Generator Type 1 (SG_1) function or Slot Generator Type 2 (SG_2) function during early phases of initialization at power-up.

The HOBS shall be set to WAITING on Bus A by the DQDB Layer Management Protocol Entity at a node that contains a Slot Generator Type 1 (SG_1) function, which is waiting to determine whether it will activate or deactivate the Head of Bus A function (i.e., Timer_H_1 is running, 7.1.2). When the Head of Bus A function is active at an SG_1 function without Timer_H_1 running, the DQDB Layer Management Protocol Entity shall set HOBS to STABLE on Bus A.

The HOBS shall be set to WAITING on Bus B by the DQDB Layer Management Protocol Entity at a node that contains a Slot Generator Type 2 (SG_2) function, which is waiting to determine whether it will activate or deactivate the Head of Bus B function (i.e., Timer_H_2 is running, 7.1.2). When the Head of Bus B function is active at an SG_2 function without Timer_H_2 running, the DQDB Layer Management Protocol Entity shall set HOBS to STABLE on Bus B.

10.2.3.3 ETSS generation

The valid values for the ETSS are PRESENT and NOT_PRESENT. (See 10.1.2.3.) The default value of the ETSS is NOT_PRESENT. The ETSS shall be set on both buses to PRESENT by the DQDB Layer Management Protocol Entity at each node that is providing the External Timing Source function.

The subnetwork external timing status can be derived at a node by examining the value of the ETSS received at the node on both buses. If neither value of ETSS is PRESENT, then there is no External Timing Source function active in the subnetwork.

10.2.3.4 Subnetworks undergoing configuration changes

After initialization at power-up, the HOBS value of WAITING is used by all CC_1 and CC_2 functions that may be required to activate or deactivate the Head of Bus function for Bus A or Bus B, respectively, due to failure or restoration of the duplex transmission link where that bus enters the node. The HOBS value of WAITING is not generated in any stable subnetwork configuration, and is only generated by a CC_1 or CC_2 function for a period limited by a timer, Timer_H_w (w = 1 or 2), respectively. (See 7.1.2.) The rationale for using the Head of Bus Arbitration Timer is given in I.2 of annex I.

There are two conditions under which a node generates the HOBS value of WAITING:

a) The Timer_H_1 for a CC_1 function with an inactive Head of Bus A function is started when the DQDB Layer subsystem receives a Ph-STATUS indication (DOWN) at Physical Layer Service Access Point A (Ph-SAP_A). The Timer_H_2 for a CC_2 function with an inactive Head of Bus B function is started when the DQDB Layer subsystem receives a Ph-STATUS indication (DOWN) at Ph-SAP_B.

   In this case, the Timer_H_w runs while the CC_w function is determining whether or not to activate the Head of Bus function. While the Timer_H_w is running, the CC_w function temporarily activates the Head of Bus function, which sends a HOBS value of WAITING. If the Timer_H_w expires, then the Head of Bus function is kept in an active state and it starts generating a HOBS value of STABLE.

The Timer_H_w is stopped before it expires if the DQDB Layer subsystem receives a Ph-STATUS indication (UP) at the appropriate Ph-SAP. This causes the CC_w function to deactivate the Head of Bus function.

b) The Timer_H_1 for a CC_1 function with an active Head of Bus A function is started when the DQDB Layer subsystem receives a Ph-STATUS indication (UP) at Ph-SAP_A. The Timer_H_2 for a CC_2 function with an active Head of Bus B function is started when the DQDB Layer subsystem receives a Ph-STATUS indication (UP) at Ph-SAP_B.

   In this case, the Timer_H_w runs while the CC_w function is determining whether or not to deactivate the Head of Bus function. While the Timer_H_w is running, the active Head of Bus function sends a HOBS value of WAITING. If the Timer_H_w expires, then the Head of Bus function is deactivated.

   The Timer_H_w is stopped before it expires if the DQDB Layer subsystem receives a Ph-STATUS indication (DOWN) at the appropriate Ph-SAP. This causes the CC_w function to keep the Head of Bus function active and start generating a HOBS value of STABLE.

These processes are described formally in 10.2.5.1 for a CC_2 function and in 10.2.6.1 for a CC_1 function. An example of the operation of the Configuration Control protocol during a configuration change is given in I.3 of annex I.

10.2.4 Default Configuration Control State Machine

The Default Configuration Control State Machine controls the operation of the Default Configuration Control (CC_D) function described in 10.2.2.1. As such, it only operates at the node that contains the Default Slot Generator function.

The Default Configuration Control State Machine consists of two components:

a) The Default External Timing Source transition diagram described in 10.2.4.1, which characterizes the transitions associated with the external timing reference resource at the node with the CC_D function.
b) The two operations tables described in 10.2.4.2, which record the status of resources being controlled by the Default Configuration Control function that must result from each of the complete set of possible input conditions. One operations table is for Head of Bus functions under control of the CC_D. The other operations table is for Timing Source functions under control of the CC_D. The current state of the transition diagram is one of the input elements to the Timing Source operations table.

Note that the status of the transmission link at the Physical Layer service access point associated with the Default Configuration Control function, as given by LINK_STATUS_D, is a direct input element of the Head of Bus operations table.

10.2.4.1 Default External Timing Source transition diagram

The transition diagram for the Default External Timing Source State Machine is defined in figure 10-6. The control functions that may cause a transition to occur are the following:

— The value of the Default Configuration Control Flag for the node, CC_D2_CONTROL, as defined in 7.4.2.
— The status of the external timing source at the node, as given by the External Timing Source Status Indicator (ETS_STATUS), as defined in 7.5.4.

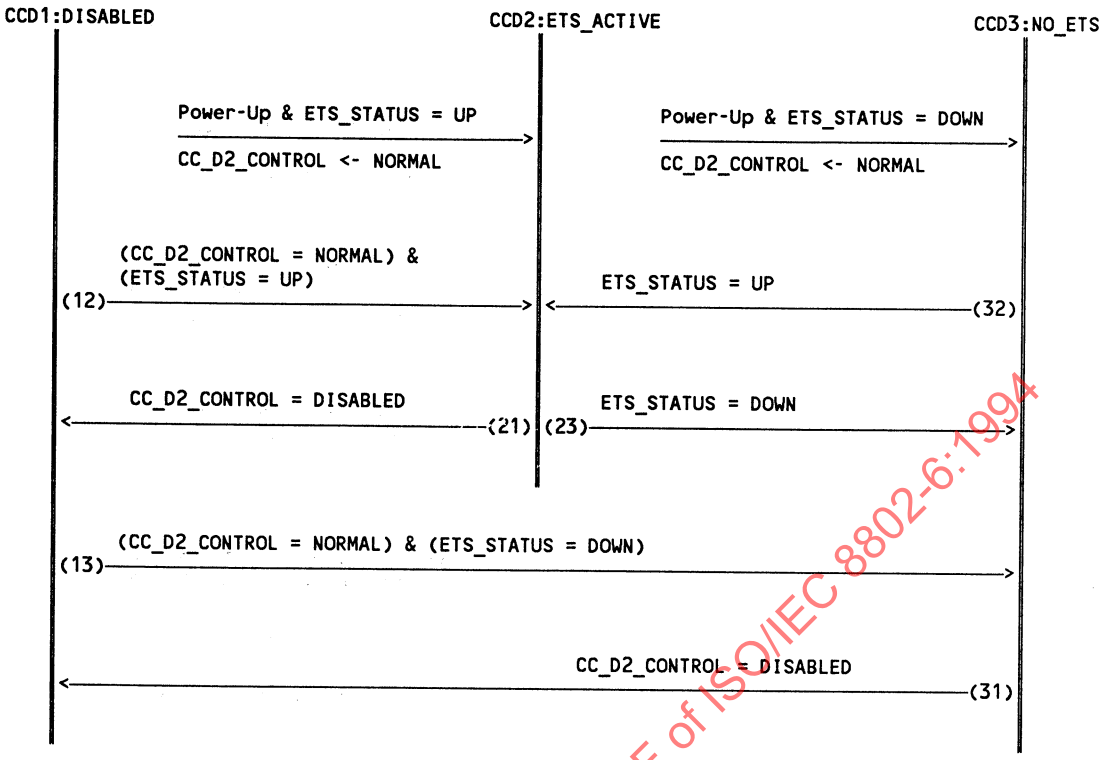


Figure 10-6—Default External Timing Source State Machine transition diagram

Default External Timing Source State Machine Transition diagram

The state machine can be in one of three states: Disabled, ETS_Active, or No_ETS. The state machine is in the ETS_Active state if Configuration Control is not disabled at the node, and the external timing source is available and to be used by the node. The state machine is in the No_ETS state if Configuration Control is not disabled at the node, and the external timing source is unavailable or not to be used by the node. The state machine is in the Disabled state if Configuration Control is disabled at the node.

The state machine shall be powered up in either the ETS_Active state or the No_ETS state, depending upon whether the External Timing Source function is available and to be used by the node (UP), or not (DOWN), respectively.

State CCD1: Disabled

The state machine remains in this state if the CC_D2_CONTROL is DISABLED.

Transition 12

If Configuration Control at the node is changed from being DISABLED to NORMAL, with the external timing source available and to be used by the node, a transition is made to the ETS_Active state.

Transition 13

If Configuration Control at the node is changed from being DISABLED to NORMAL, with the external timing source unavailable or not to be used by the node, a transition is made to the No_ETS state.

State CCD2: ETS_Active

The state machine remains in this state if the CC_D2_CONTROL is NORMAL and the external timing source is available and to be used by the node.

Transition 21

If Configuration Control at the node is changed from being NORMAL to DISABLED, a transition is made to the Disabled state.

Transition 23

If the external timing source at the node becomes unavailable or is not to be used, a transition is made to the No_ETS state.

State CCD3: No_ETS

The state machine remains in this state if the CC_D2_CONTROL is NORMAL, and the external timing source is unavailable or is not to be used by the node.

Transition 31

If Configuration Control at the node is changed from being NORMAL to DISABLED, a transition is made to the Disabled state.

Transition 32

If the external timing source at the node becomes available and is to be used, a transition is made to the ETS_Active state.

10.2.4.2 Default Configuration Control operations tables

The Head of Bus function and Timing Source function operations tables for the Default Configuration Control State Machine are defined in tables 10-10a) and 10-10b), respectively. Each row of an operations table specifies one set of possible input conditions for the Default Configuration Control (CC_D) function, and the status of the resource managed by the CC_D function that must result as the output from the input condition. The complete set of rows describes all possible input conditions to the CC_D function for that resource.

The elements of the input condition to the Default Configuration Control function (CC_D) Head of Bus function operations table are as follows:

a) The value of the Default Configuration Control Flag for the node, CC_D2_CONTROL, as defined in 7.4.2.
b) The Link Status Indicator, LINK_STATUS_D, which records the current status of the duplex transmission link associated with the Ph-SAP_A at the node, as defined in 7.5.3.
c) The values of the DSGS and HOBS being received at Ph-SAP_A for Bus A. (Subclause 10.2 defines which received value of DSGS and HOBS shall be used.)

Table 10-10a)—Default Configuration Control function (CC_D)
Head of Bus function operations table

INPUT								OUTPUT							
CC_D2_CONTROL	LINK_STATUS_D (Ph-SAP_A)	Bus A SNCF (From Ph-SAP_A)			Bus B SNCF (From CC_2)			CC_STATUS_D	HOB_OPERATION_D	Bus A SNCF (To CC_2)			Bus B SNCF (To Ph-SAP_A)		
		DSGS	HOBS		DSGS	HOBS				DSGS	HOBS		DSGS	HOBS	
DISABLED	X	X	X		X	X		INACTIVE	NULL	Relay	Relay		Relay	Relay	
NORMAL	UP	PRES	STAB		PRES	STAB		ACTIVE	DSG,HOB_A(1),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	UP	NOT	WAIT		NOT	WAIT		ACTIVE	DSG,HOB_A(2),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	UP	NOT	STAB		NOT	WAIT		ACTIVE	DSG,HOB_A(2),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	UP	NOT	WAIT		NOT	STAB		ACTIVE	DSG,HOB_A(2),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	UP	NOT	STAB		NOT	STAB		ACTIVE	DSG	PRES	Relay		PRES	Relay	
NORMAL	DOWN	N/A	N/A		PRES (transient)	STAB		ACTIVE	DSG,HOB_A(2),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	DOWN	N/A	N/A		NOT	WAIT		ACTIVE	DSG,HOB_A(2),HOB_B	PRES	STAB		PRES	STAB	
NORMAL	DOWN	N/A	N/A		NOT	STAB		ACTIVE	DSG,HOB_A(1)	PRES	STAB		PRES	Relay	

KEY:

X = Don't Care
N/A = Not Applicable
DSG = DEFAULT_SLOT_GENERATOR
HOB_A = HEAD_OF_BUS_A
HOB_B = HEAD_OF_BUS_B
PRES = PRESENT
NOT = NOT_PRESENT
STAB = STABLE
WAIT = WAITING

NOTES:

(1) Page Counter State Machine for Head of Bus A is enabled.
(2) Page Counter State Machine for Head of Bus A is disabled.
(3) The received combination of DSGS = PRES and HOBS = WAIT is an error.

Table 10-10b)—Default Configuration Control function (CC_D)
Timing Source function operations table

INPUT								OUTPUT						
CCD State	HOB_OPERATION_D	Bus A SNCF (From Ph-SAP_A)			Bus B SNCF (From CC_2)			Timing Source	Bus A SNCF (To CC_2)			Bus B SNCF (To Ph-SAP_A)		
				ETSS			ETSS				ETSS			ETSS
Disabled	NULL			X			X	EITHER_BUS			Relay			Relay
ETS_Active	X (1)			X			X	EXTERNAL_CLOCK			PRES			PRES
No_ETS	DSG,HOB_A,HOB_B			PRES			PRES	BUS_B (4)			NOT			NOT
No_ETS (2)	DSG,HOB_A,HOB_B			NOT			PRES	BUS_B (4)			NOT			NOT
No_ETS (2)	DSG,HOB_A,HOB_B			PRES			NOT	BUS_B (4)			NOT			NOT
No_ETS	DSG,HOB_A,HOB_B			NOT			NOT	NODE_CLOCK			NOT			NOT
No_ETS	DSG			PRES			PRES	EITHER_BUS			Relay			Relay
No_ETS	DSG			NOT			PRES	EITHER_BUS			Relay			Relay
No_ETS	DSG			PRES			NOT	EITHER_BUS			Relay			Relay
No_ETS	DSG			NOT			NOT	NODE_CLOCK (3)			Relay			Relay
No_ETS	DSG,HOB_A			N/A			PRES	BUS_B (4)			NOT			Relay
No_ETS	DSG,HOB_A			N/A			NOT	NODE_CLOCK			NOT			Relay

KEY:

X = Don't Care
N/A = Not Applicable
DSG = DEFAULT_SLOT_GENERATOR
HOB_A = HEAD_OF_BUS_A
HOB_B = HEAD_OF_BUS_B
PRES = PRESENT
NOT = NOT_PRESENT

NOTES:

(1) If HOB_OPERATION_D equals (DSG), then the node shall maintain data path continuity, as per 5.4.2.4.2.
(2) This input condition is transient only.
(3) The node shall maintain data path continuity, as per 5.4.2.4.2.
(4) Timing source will be NODE_CLOCK if LINK_STATUS_2 is DOWN.

d) The values of the DSGS and HOBS being received on Bus B, and after having been operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which received value of DSGS and HOBS shall be used.)

The output status resulting from the operation is described by the following:

a) The Configuration Control Status Indicator, CC_STATUS_D, which records whether any Head of Bus function resources under the control of the CC_D function are active or inactive, with consequences as defined in 7.5.1.
b) The Head of Bus Operation Indicator, HOB_OPERATION_D, which records which Head of Bus functions have been activated by the CC_D function, as defined in 7.5.2.
c) The DSGS and HOBS values being written to Bus A by the DQDB Layer Management Protocol Entity on behalf of the CC_D function. This shall be done before the DSGS and HOBS are operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which value of DSGS and HOBS shall be used if the output column contains the value Relay.)
d) The DSGS and HOBS values being written by the DQDB Layer Management Protocol Entity to Bus B at Ph-SAP_A. (Subclause 10.2 defines which value of DSGS and HOBS shall be used if the output column contains the value Relay.)

The elements of the input condition to the Default Configuration Control function (CC_D) Timing Source function operations table are as follows:

a) The state of the Default External Timing Source State Machine transition diagram, defined in figure 10-6 [given as CCD State in table 10-10b)].
b) The Head of Bus Operation Indicator, HOB_OPERATION_D, which records which Head of Bus functions have been activated by the CC_D function, as defined in 7.5.2. (This is an output from the CC_D Head of Bus functions operations table.)
c) The value of the ETSS being received at Ph-SAP_A for Bus A. (Subclause 10.2 defines which received value of ETSS shall be used.)
d) The value of the ETSS being received on Bus B, and after having been operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which received value of ETSS shall be used.)

The output status resulting from the operation is described by the following:

a) The 125 µs timing source to be used by the Physical Layer at the node. The specified source is sent as the source parameter in a Ph-TIMING-SOURCE request at Ph-SAP_A. (See 4.4.)
b) The ETSS value being written to Bus A by the DQDB Layer Management Protocol Entity on behalf of the CC_D function. This shall be done before being the ETSS is operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which value of ETSS shall be used if the output column contains the value Relay.)
c) The ETSS value being written by the DQDB Layer Management Protocol Entity to Bus B at Ph-SAP_A. (Subclause 10.2 defines which value of ETSS shall be used if the output column contains the value Relay.)

10.2.5 Configuration Control Type 2 State Machine

The Configuration Control Type 2 State Machine controls the operation of the Configuration Control Type 2 (CC_2) function described in 10.2.2.2. It therefore operates at all nodes in the subnetwork.

The CC_2 function at the node containing the Default Configuration Control function shall support all capabilities allowed by both settings of the Default Configuration Control Flag, CC_D2_CONTROL. (See 7.4.2.) All other nodes conforming to this part of ISO/IEC 8802 need only support the capabilities defined by the Configuration Control Flag, CC_12_CONTROL, being set to DISABLED. (See 7.4.1.)

The Configuration Control Type 2 State Machine consists of three components:

a) The Head of Bus transition diagram described in 10.2.5.1, which characterizes the transitions associated with the Head of Bus B resource at the node.

b) The External Timing Source transitions described in 10.2.5.2, which characterize the transitions associated with the external timing reference resource at the node.

c) The operations table described in 10.2.5.3, which records the status of resources being controlled by the Configuration Control Type 2 function that must result from each of the complete set of possible input conditions. The current state of the transitions described in 10.2.5.1 and 10.2.5.2 are two of the input elements to the operations table.

10.2.5.1 Head of Bus transition diagram

The transition diagram for the Head of Bus State Machine is defined in figure 10-7. The control functions that may cause a transition to occur are the following:

— Either the value of the Default Configuration Control Flag for the node, CC_D2_CONTROL, if the node contains the CC_D function, or the value of the Configuration Control Flag for the node, CC_12_CONTROL, if the node contains a CC_1 function. The CC_D2_CONTROL and CC_12_CONTROL Flags are defined in 7.4.2 and 7.4.1, respectively.

— The operations of the Head of Bus Arbitration Timer associated with the CC_2 function, Timer_H_2, defined in 7.1.2.

— The Link Status Indicator, LINK_STATUS_2, which records the current status of the duplex transmission link associated with the Ph-SAP_B at the node, as defined in 7.5.3.

— The value of BIF (see 10.1.1) received at the node. (Subclause 10.2 defines which received value of BIF shall be used.)

Head of Bus transition diagram (CC_2 function)

The state machine can be in one of five states: Initialize, HOB_Dflt, HOB_Wait_Dflt, HOB_Active, or HOB_Wait_Active. The state machine is in the Initialize state if a node that contains a CC_1 function has been powered up and has not yet received either (BIF = Bus A) or (BIF = Bus B) or has received BIF values that conflicted with the association of labels with Ph-SAPs at the node. The state machine is in the HOB_Dflt state if the Head of Bus B function is not active. The state machine is in the HOB_Wait_Dflt state if the Head of Bus B function is currently being activated. The state machine is in the HOB_Active state if the Head of Bus B function is active. The state machine is in the HOB_Wait_Active state if the Head of Bus B function is currently being deactivated.

The state machine is powered up in the HOB_Active state if the node contains the CC_D function. This causes the CC_2 function to activate the Head of Bus B resource, which generates a HOBS value of STABLE. The state machine is powered up in the Initialize state if the node contains a CC_1 function.

State HBB0: Initialize

The state machine at a node with a CC_1 function remains in this state if the node has been powered up and has not yet received either (BIF = Bus A) or (BIF = Bus B) from either bus, as required by 5.4.2.2, or has received BIF values that conflicted with the association of labels with Ph-SAPs at the node. In this state, the node shall generate a BIF value of 00, if the Physical Layer is generating EMPTY octets of type DQDB_MANAGEMENT in Ph-DATA indication primitives at the associated Ph-SAP.

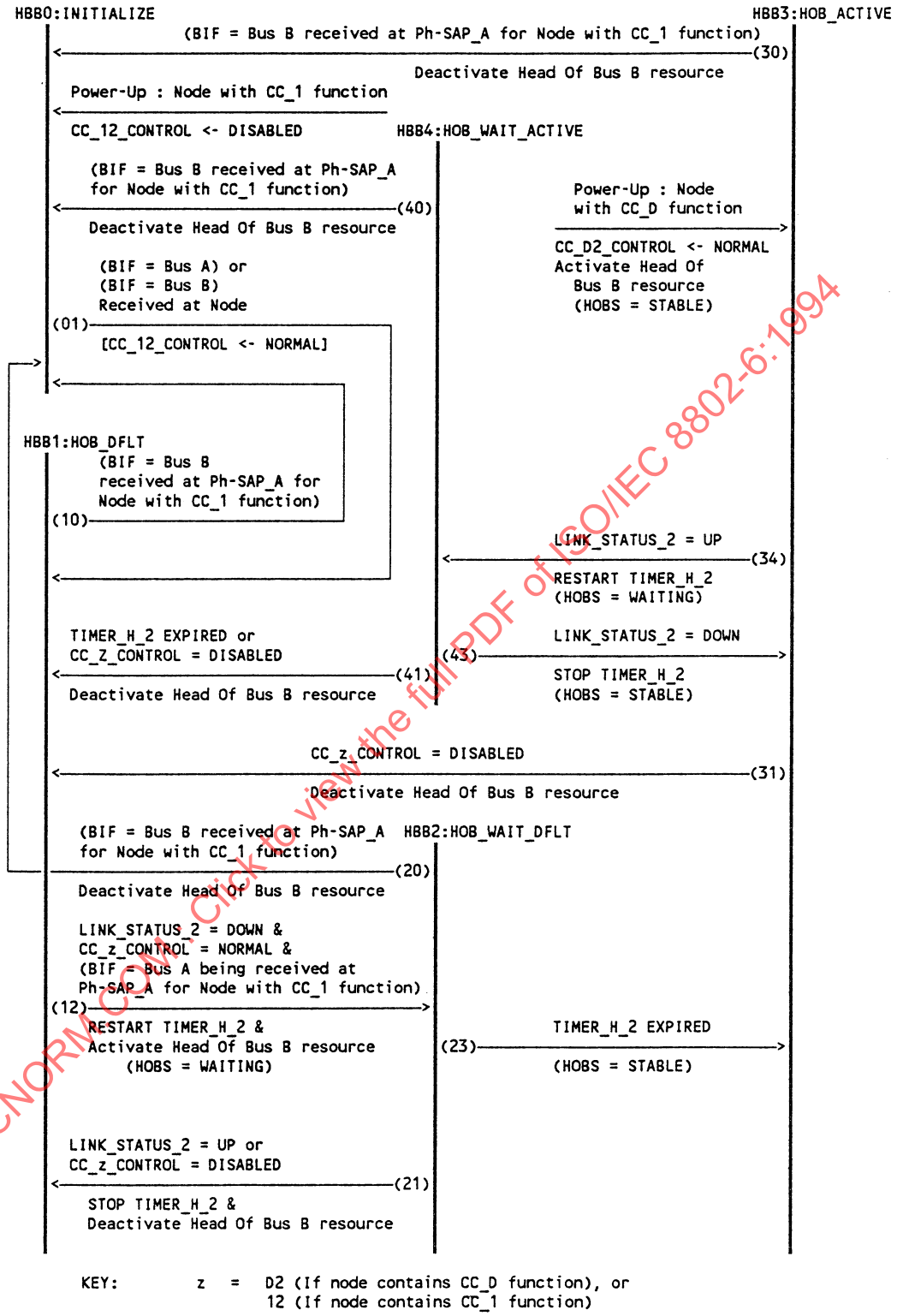


Figure 10-7—Head of Bus State Machine transition diagram (CC_function)

Transition 01

When a node with a CC_1 function receives either (BIF = Bus A) or (BIF = Bus B) from either bus, then the node is able to associate a label with the two Ph-SAPs, and the state machine enters the HOB_Dflt state. If the node supports Head of Bus functions, then CC_12_CONTROL shall be set to NORMAL.

Transition 10

If the node contains a CC_1 function and receives a BIF value of Bus B at the Ph-SAP labeled Ph-SAP_A, then the state machine returns to the Initialize state.

State HBB1: HOB_Dflt

The state machine remains in this state if the CC_z_CONTROL (z = 12 or D2) is DISABLED or if the CC_z_CONTROL is NORMAL and the LINK_STATUS_2 indicator is UP. In this state the Head of Bus B function is not active.

Transition 12

If the LINK_STATUS_2 indicator changes to DOWN, and if CC_z_CONTROL (z = 12 or D2) is NORMAL, and (only if the node contains a CC_1 function) the node is receiving BIF values of Bus A at the Ph-SAP labeled Ph-SAP_A, then the state machine restarts Timer_H_2 and enters the HOB_Wait_Dflt state. This causes the CC_2 function to activate the Head of Bus B resource, which generates HOBS = WAITING.

State HBB2: HOB_Wait_Dflt

The state machine remains in this state if the CC_z_CONTROL (z = 12 or D2) is NORMAL and the LINK_STATUS_2 indicator remains DOWN and the Timer_H_2 is running. In this state the Head of Bus B function is participating in the arbitration procedure to determine which CC_2 function will maintain an active Head of Bus B function.

Transition 20

If the node contains a CC_1 function and receives a BIF value of Bus B at the Ph-SAP labeled Ph-SAP_A, then the state machine enters the Initialize state. This causes the CC_2 function to deactivate the Head of Bus B resource.

Transition 21

If the LINK_STATUS_2 indicator changes back to UP, or if the CC_z_CONTROL (z = 12 or D2) is DISABLED, then the state machine returns to the HOB_Dflt state. This causes the CC_2 function to deactivate the Head of Bus B resource.

Transition 23

If the Timer_H_2 expires, then the state machine enters the HOB_Active state. The Head of Bus B function generates HOBS = STABLE.

State HBB3: HOB_Active

The state machine remains in this state if the CC_z_CONTROL (z = 12 or D2) is NORMAL and the LINK_STATUS_2 indicator remains DOWN. In this state the Head of Bus B function is active.

Transition 34

If the LINK_STATUS_2 indicator changes to UP, then the state machine restarts Timer_H_2 and enters the HOB_Wait_Active state. The Head of Bus B function generates HOBS = WAITING.

Transition 30

If the node contains a CC_1 function and receives a BIF value of Bus B at the Ph-SAP labeled Ph-SAP_A, then the state machine enters the Initialize state. This causes the CC_2 function to deactivate the Head of Bus B resource.

Transition 31

If the CC_z_CONTROL (z = 12 or D2) is DISABLED, then the state machine directly enters the HOB_Dflt state. This causes the CC_2 function to deactivate the Head of Bus B resource.

State HBB4: HOB_Wait_Active

The state machine remains in this state if the CC_z_CONTROL (z = 12 or D2) is NORMAL and the LINK_STATUS_2 indicator remains UP and the Timer_H_2 is running. In this state the Head of Bus B function is waiting to cease operation.

Transition 40

If the node contains a CC_1 function and receives a BIF value of Bus B at the Ph-SAP labeled Ph-SAP_A, then the state machine enters the Initialize state. This causes the CC_2 function to deactivate the Head of Bus B resource.

Transition 41

If the Timer_H_2 expires, or if the CC_z_CONTROL (z = 12 or D2) is DISABLED, then the state machine enters the HOB_Dflt state. This causes the CC_2 function to deactivate the Head of Bus B resource.

Transition 43

If the LINK_STATUS_2 indicator changes back to DOWN, then the state machine returns to the HOB_Active state. The Head of Bus B function generates HOBS = STABLE.

10.2.5.2 External Timing Source transitions

The transitions for the External Timing Source State Machine are defined below. The control functions that may cause a transition are as follows:

— Either the value of the Default Configuration Control Flag for the node, CC_D2_CONTROL, if the node contains the CC_D function, or the value of the Configuration Control Flag for the node, CC_12_CONTROL, if the node contains a CC_1 function. The CC_D2_CONTROL and CC_12_CONTROL Flags are defined in 7.4.2 and 7.4.1, respectively.

— The status of the external timing source at the node, as given by the External Timing Source Status Indicator (ETS_STATUS), as defined in 7.5.4.

External Timing Source transitions (CC_2 function)

The state machine can be in one of two states: No_ETS or ETS_Active. The state machine is in the No_ETS state if the external timing reference function is not active. The state machine is in the ETS_Active state if the external timing reference function is active.

The state machine is in the ETS_Active state if Condition A below is true. The state machine is in the No_ETS state if Condition A is not true.

Condition A

If the node contains a CC_1 function, then
Condition A = (ETS_STATUS = UP) & (CC_12_CONTROL = NORMAL)
If the node contains the CC_D function, then
Condition A = (ETS_STATUS = UP) & (CC_D2_CONTROL = NORMAL)

10.2.5.3 Configuration Control Type 2 operations table

The operations table for the Configuration Control Type 2 State Machine is defined in table 10-11. Each row of the operations table specifies one set of possible input conditions for the Configuration Control Type 2 (CC_2) function, and the status of the resources managed by the CC_2 function that must result as the output from the input condition. The complete set of rows describes all possible input conditions to the CC_2 function.

The elements of the input condition to the Configuration Control Type 2 operations table are as follows:

a) The state of the Head of Bus State Machine transition diagram, defined in figure 10-7. (Given as HBB State in table 10-11.)
b) The state of the External Timing Source State Machine, defined in 10.2.5.2. (Given as ETS State in table 10-11.)
c) The values of the DSGS, HOBS, and ETSS being received on Bus A, and after having been operated on for any requirements of the other Configuration Control function at the node (either CC_D or CC_1). (Subclause 10.2 defines which received value of DSGS, HOBS, and ETSS shall be used.)

The output status resulting from the operation is described by the following:

a) The Configuration Control Status Indicator, CC_STATUS_2, which records whether any resources under the control of the CC_2 function are active or inactive, with consequences as defined in 7.5.1.
b) The Head of Bus Operation Indicator, HOB_OPERATION_2, which records which Head of Bus functions have been activated by the CC_2 function, as defined in 7.5.2.
c) The 125 µs timing source to be used by the Physical Layer at the node. The specified source is sent as the source parameter in a Ph-TIMING-SOURCE request at Ph-SAP_B (see 4.4).
d) The DSGS, HOBS, and ETSS values being written to Bus B by the DQDB Layer Management Protocol Entity on behalf of the CC_2 function. This shall be done before the DSGS, HOBS, and ETSS are operated on for any requirements of the other Configuration Control function at the node (either CC_D or CC_1). (Subclause 10.2 defines which values of DSGS, HOBS, and ETSS shall be used if the output column contains the value Relay.)

10.2.6 Configuration Control Type 1 State Machine

The Configuration Control Type 1 State Machine controls the operation of the Configuration Control Type 1 (CC_1) function described in 10.2.2.3. It therefore operates at all nodes in the subnetwork, except the node containing the Default Configuration Control function.

Table 10-11—Configuration Control Type 2 function operations table

INPUT					OUTPUT					
HBB State	ETS State	Bus A SNCF (From CC_1/CC_D)			CC_STATUS_2	HOB_OPERATION_2	Timing Source	Bus B SNCF (To CC_1/CC_D)		
		DSGS	HOBS	ETSS				DSGS	HOBS	ETSS
Initialize	No_ETS	X	X	X	INACTIVE	NULL	NODE_CLOCK	Relay	Relay	Relay
Initialize	ETS_Active	X	X	X	INACTIVE	NULL	EXTERNAL_CLOCK	Relay	Relay	Relay
HOB_Dflt	No_ETS	X	X	X	INACTIVE	NULL	EITHER_BUS	Relay	Relay	Relay
HOB_Wait_Dflt	No_ETS	X	X	X	ACTIVE	HOB_B	NODE_CLOCK	NOT	WAIT	NOT
HOB_Wait_Active	No_ETS	X	X	X	ACTIVE	HOB_B	NODE_CLOCK	NOT	WAIT	NOT
HOB_Wait_Dflt	ETS_Active	X	X	X	ACTIVE	HOB_B	EXTERNAL_CLOCK	NOT	WAIT	PRES
HOB_Wait_Active	ETS_Active	X	X	X	ACTIVE	HOB_B	EXTERNAL_CLOCK	NOT	WAIT	PRES
HOB_Dflt	ETS_Active	X	X	X	INACTIVE	NULL (1)	EXTERNAL_CLOCK	Relay	Relay	PRES
HOB_Active	No_ETS	X	X	X	ACTIVE	HOB_B	BUS_A (2)	NOT	STAB	NOT
HOB_Active	ETS_Active	X	X	X	ACTIVE	HOB_B	EXTERNAL_CLOCK	NOT	STAB	PRES

KEY:

X = Don't Care
N/A = Not Applicable
HOB_B = HEAD_OF_BUS_B
PRES = PRESENT
NOT = NOT_PRESENT
STAB = STABLE
WAIT = WAITING

NOTES:

(1) The node shall maintain data path continuity, as per 5.4.2.4.2.
(2) Timing source will be NODE_CLOCK if LINK_STATUS_1/LINK_STATUS_D is DOWN.

Nodes conforming to this part of ISO/IEC 8802 need only support the capabilities defined by the Configuration Control Flag, CC_12_CONTROL, being set to DISABLED. (See 7.4.1.)

The Configuration Control Type 1 State Machine consists of three components:

a) The Head of Bus transition diagram described in 10.2.6.1, which characterizes the transitions associated with the Head of Bus A resource at the node.
b) The External Timing Source transitions described in 10.2.6.2, which characterize the transitions associated with the external timing reference resource at the node.
c) The operations table described in 10.2.6.3, which records the status of resources being controlled by the Configuration Control Type 1 function that must result from each of the complete set of possible input conditions. The current state of the transitions described in 10.2.6.1 and 10.2.6.2 are two of the input elements to the operations table.

10.2.6.1 Head of Bus transition diagram

The transition diagram for the Head of Bus State Machine is defined in figure 10-8. The control functions that may cause a transition to occur are as follows:

— The value of the Configuration Control Flag for the node, CC_12_CONTROL, defined in 7.4.1.
— The operations of the Head of Bus Arbitration Timer associated with the CC_1 function, Timer_H_1, defined in 7.1.2.
— The Link Status Indicator, LINK_STATUS_1, which records the current status of the duplex transmission link associated with the Ph-SAP_A at the node, as defined in 7.5.3.
— The value of BIF (see 10.1.1) received at the node. (Subclause 10.2 defines which received value of BIF shall be used.)

Head of Bus transition diagram (CC_1 function)

The state machine can be in one of five states: Initialize, HOB_Dflt, HOB_Wait_Dflt, HOB_Active, or HOB_Wait_Active. The state machine is in the Initialize state if a node has been powered up and has not yet received either (BIF = Bus A) or (BIF = Bus B) or has received BIF values that conflicted with the association of labels with Ph-SAPs at the node. The state machine is in the HOB_Dflt state if the Head of Bus A function is not active. The state machine is in the HOB_Wait_Dflt state if the Head of Bus A function is currently being activated. The state machine is in the HOB_Active state if the Head of Bus A function is active. The state machine is in the HOB_Wait_Active state if the Head of Bus A function is currently being deactivated.

The state machine is powered up in the Initialize state.

State HBA0: Initialize

The state machine remains in this state if the node has been powered up and has not yet received either (BIF = Bus A) or (BIF = Bus B) from either bus, as required by 5.4.2.2 or has received BIF values that conflicted with the association of labels with Ph-SAPs at the node. In this state, the node shall generate a BIF value of 00, if the Physical Layer is generating EMPTY octets of type DQDB_MANAGEMENT in Ph-DATA indication primitives at the associated Ph-SAP.

Transition 01

When a node receives either (BIF = Bus A) or (BIF = Bus B) from either bus, then the node is able to associate a label with the two Ph-SAPs, and the state machine enters the HOB_Dflt state. If the node supports Head of Bus functions then CC_12_CONTROL shall be set to NORMAL.

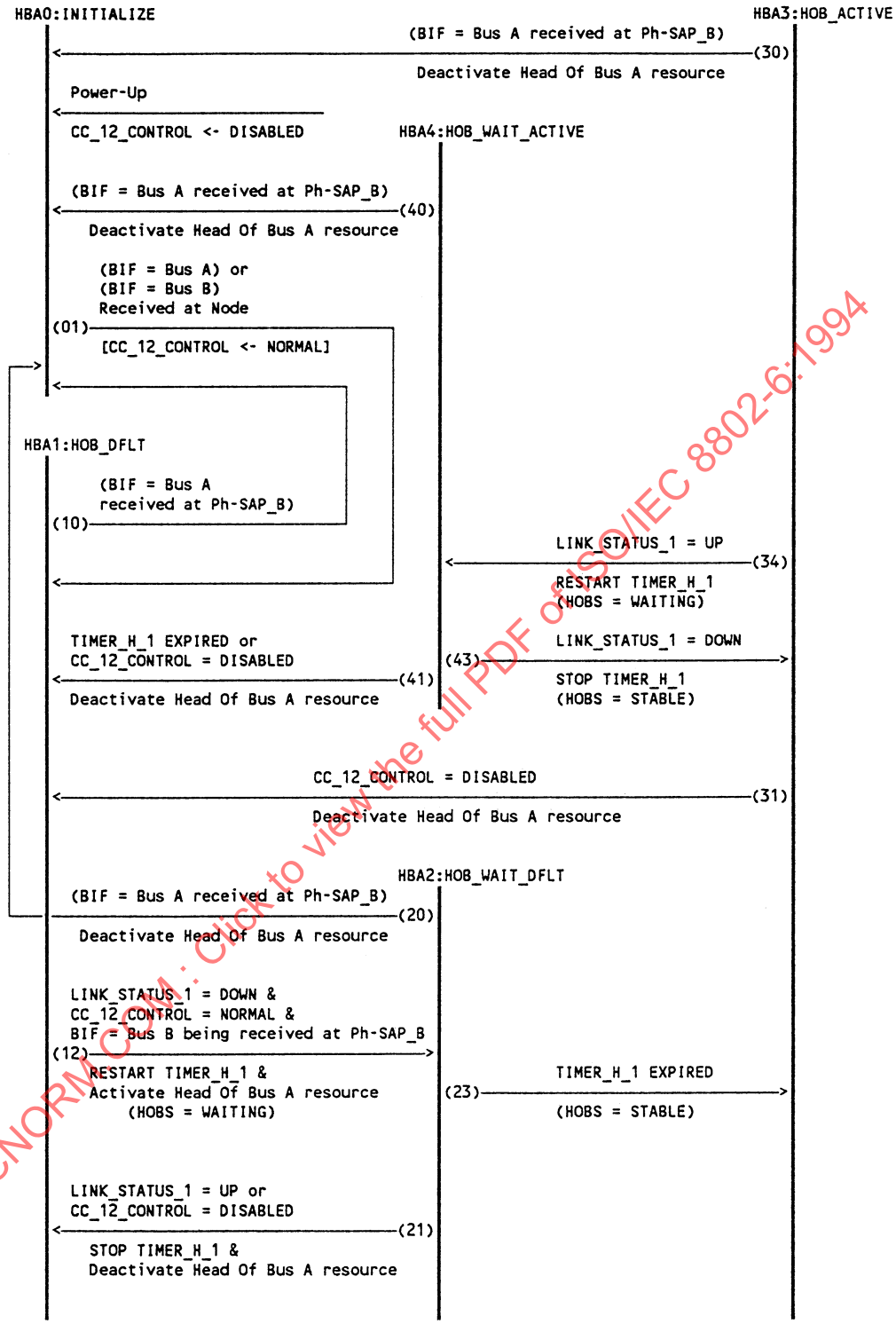


Figure 10-8—Head of Bus State Machine transition diagram (CC_1 function)

Transition 10

If the node receives a BIF value of Bus A at the Ph-SAP labeled Ph-SAP_B, then the state machine returns to the Initialize state.

State HBA1: HOB_Dflt

The state machine remains in this state if the CC_12_CONTROL is DISABLED or if the CC_12_CONTROL is NORMAL and the LINK_STATUS_1 indicator is UP. In this state the Head of Bus A function is not active.

Transition 12

If the LINK_STATUS_1 indicator changes to DOWN, if CC_12_CONTROL is NORMAL, and if the node is receiving BIF values of Bus B at the Ph-SAP labeled Ph-SAP_B, then the state machine restarts Timer_H_1 and enters the HOB_Wait_Dflt state. This causes the CC_1 function to activate the Head of Bus A resource, which generates HOBS = WAITING.

State HBA2: HOB_Wait_Dflt

The state machine remains in this state if the CC_12_CONTROL is NORMAL and the LINK_STATUS_1 indicator remains DOWN and the Timer_H_1 is running. In this state the Head of Bus A function is participating in the arbitration procedure to determine which CC_1 function will maintain an active Head of Bus A function.

Transition 20

If the node receives a BIF value of Bus A at the Ph-SAP labeled Ph-SAP_B, then the state machine enters the Initialize state. This causes the CC_1 function to deactivate the Head of Bus A resource.

Transition 21

If the LINK_STATUS_1 indicator changes back to UP, or if the CC_12_CONTROL is DISABLED, then the state machine returns to the HOB_Dflt state. This causes the CC_1 function to deactivate the Head of Bus A resource.

Transition 23

If the Timer_H_1 expires, then the state machine enters the HOB_Active state. The Head of Bus A function generates HOBS = STABLE.

State HBA3: HOB_Active

The state machine remains in this state if the CC_12_CONTROL is NORMAL and the LINK_STATUS_1 indicator remains DOWN. In this state the Head of Bus A function is active.

Transition 34

If the LINK_STATUS_1 indicator changes to UP, then the state machine restarts Timer_H_1 and enters the HOB_Wait_Active state. The Head of Bus A function generates HOBS = WAITING.

Transition 30

If the node receives a BIF value of Bus A at the Ph-SAP labeled Ph-SAP_B, then the state machine enters the Initialize state. This causes the CC_1 function to deactivate the Head of Bus A resource.

Transition 31

If the CC_12_CONTROL is DISABLED, then the state machine directly enters the HOB_Dflt state. This causes the CC_1 function to deactivate the Head of Bus A resource.

State HBA4: HOB_Wait_Active

The state machine remains in this state if the CC_12_CONTROL is NORMAL and the LINK_STATUS_1 indicator remains UP and the Timer_H_1 is running. In this state Head of Bus A function is waiting to cease operation.

Transition 40

If the node receives a BIF value of Bus A at the Ph-SAP labeled Ph-SAP_B, then the state machine enters the Initialize state. This causes the CC_1 function to deactivate the Head of Bus A resource.

Transition 41

If the Timer_H_1 expires, or if the CC_12_CONTROL is DISABLED, then the state machine enters the HOB_Dflt state. This causes the CC_1 function to deactivate the Head of Bus A resource.

Transition 43

If the LINK_STATUS_1 indicator changes back to DOWN, then the state machine returns to the HOB_Active state. The Head of Bus A function generates HOBS = STABLE.

10.2.6.2 External Timing Source transitions

The transitions for the External Timing Source State Machine are defined below. The control functions that may cause a transition are as follows:

— The value of the Configuration Control Flag for the node, CC_12_CONTROL, defined in 7.4.1.
— The status of the external timing source at the node, as given by the External Timing Source Status Indicator (ETS_STATUS), as defined in 7.5.4.

External Timing Source transitions (CC_1 function)

The state machine can be in one of two states: No_ETS or ETS_Active. The state machine is in the No_ETS state if the external timing reference function is not active. The state machine is in the ETS_Active state if the external timing reference function is active.

The state machine is in the ETS_Active state if Condition A below is true. The state machine is in the No_ETS state if Condition A is not true.

Condition A

Condition A = (ETS_STATUS = UP) & (CC_12_CONTROL = NORMAL)

10.2.6.3 Configuration Control Type 1 operations table

The operations table for the Configuration Control Type 1 State Machine is defined in table 10-12. Each row of the operations table specifies one set of possible input conditions for the Configuration Control Type 1 (CC_1) function, and the status of the resources managed by the CC_1 function that must result as the output from the input condition. The complete set of rows describes all possible input conditions to the CC_1 function.

The elements of the input condition to the Configuration Control Type 1 operations table are as follows:

a) The state of the Head of Bus State Machine transition diagram, defined in figure 10-8 (given as HBA State in table 10-12).
b) The state of the External Timing Source State Machine, defined in 10.2.6.2 (given as ETS State in table 10-12).
c) The value of the DSGS, HOBS, and ETSS being received on Bus B, and after having been operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which received value of DSGS, HOBS, and ETSS shall be used.)

The output status resulting from the operation is described by the following:

— The Configuration Control Status Indicator, CC_STATUS_1, which records whether any resources under the control of the CC_1 function are active or inactive, with consequences as defined in 7.5.1.
— The Head of Bus Operation Indicator, HOB_OPERATION_1, which records which Head of Bus functions have been activated by the CC_1 function, as defined in 7.5.2.
— The 125 μs timing source to be used by the Physical Layer at the node. The specified source is sent as the source parameter in a Ph-TIMING-SOURCE request at Ph-SAP_A. (See 4.4.)
— The DSGS, HOBS, and ETSS values being written to Bus A by the DQDB Layer Management Protocol Entity on behalf of the CC_1 function. This shall be done before the DSGS, HOBS, and ETSS are operated on for any requirements of the CC_2 function at the node. (Subclause 10.2 defines which values of DSGS, HOBS, and ETSS shall be used if the output column contains the value Relay.)

10.3 MID Page Allocation protocol

The MID Page Allocation protocol is used to communicate information between the DQDB Layer Management Protocol Entities of nodes to support the operation of the MID Page Allocation (MPA) function, described in 5.4.4. The MPA function controls the allocation of Message Identifiers (MIDs) to nodes along the Dual Bus. The unit of allocation of MIDs is an MID page of one MID value. The information required to support the MPA protocol is carried in the MID Page Allocation Field (MPAF) of the DQDB Layer Management Information field, described in 10.1.3.

The MPA protocol operates by logically associating an MID page value with each MPAF and allowing nodes to deterministically arbitrate for allocation of this MID page value. The MPAF consists of three subfields: the Page Reservation Subfield, the Page Counter Modulus Subfield, and the Page Counter Control Subfield. The Page Counter Control (PCC) Subfield is used to control the MID page value that will be logically associated with the Page Reservation (PR) Subfield in the next MPAF received at the node. The PR subfield is used to indicate whether the associated MID page value is reserved or not reserved for use by a node. The Page Counter Modulus (PCM) Subfield is used as an additional check that all nodes are associating the correct MID page value with each MPAF.

The MPA protocol is a two-pass protocol, with MID page values being obtained on Bus B and then maintained on Bus A. This process is illustrated in figure 10-9.

Table 10-12—Configuration Control Type 1 function operations table

INPUT					OUTPUT					
HBA State	ETS State	Bus B SNCF (From CC_2)			CC_STATUS_1	HOB_OPERATION_1	Timing Source	Bus A SNCF (To CC_2)		
		DSGS	HOBS	ETSS				DSGS	HOBS	ETSS
Initialize	No_ETS	X	X	X	INACTIVE	NULL	NODE_CLOCK	Relay	Relay	Relay
Initialize	ETS_Active	X	X	X	INACTIVE	NULL	EXTERNAL_CLOCK	Relay	Relay	Relay
HOB_Dflt	No_ETS	X	X	X	INACTIVE	NULL	EITHER_BUS	Relay	Relay	Relay
HOB_Wait_Dflt	No_ETS	X	X	X	ACTIVE	HOB_A (2)	NODE_CLOCK	NOT	WAIT	NOT
HOB_Wait_Active	No_ETS	X	X	X	ACTIVE	HOB_A (2)	NODE_CLOCK	NOT	WAIT	NOT
HOB_Wait_Dflt	ETS_Active	X	X	X	ACTIVE	HOB_A (2)	EXTERNAL_CLOCK	NOT	WAIT	PRES
HOB_Wait_Active	ETS_Active	X	X	X	ACTIVE	HOB_A (2)	EXTERNAL_CLOCK	NOT	WAIT	PRES
HOB_Dflt	ETS_Active	X	X	X	INACTIVE	NULL (1)	EXTERNAL_CLOCK	Relay	Relay	PRES
HOB_Active	No_ETS	PRES	X(3)	PRES	ACTIVE	HOB_A	BUS_B (4)	NOT	STAB	NOT
HOB_Active	No_ETS	PRES	X(3)	NOT	ACTIVE	HOB_A	BUS_B (4)	NOT	STAB	NOT
HOB_Active	No_ETS	NOT	X(3)	PRES	ACTIVE	HOB_A	BUS_B (4)	NOT	STAB	NOT
HOB_Active	No_ETS	NOT	X(3)	NOT	ACTIVE	HOB_A	NODE_CLOCK	NOT	STAB	NOT
HOB_Active	ETS_Active	X	X(3)	X	ACTIVE	HOB_A	EXTERNAL_CLOCK	NOT	STAB	PRES

KEY:

X = Don't Care
N/A = Not Applicable
HOB_A = HEAD_OF_BUS_B
PRES = PRESENT
NOT = NOT_PRESENT
STAB = STABLE
WAIT = WAITING

NOTES:

(1) The node shall maintain data path continuity, as per 5.4.2.4.2.
(2) Page Counter State Machine for Head of Bus A is disabled.
(3) Page Counter State Machine for Head of Bus A is enabled if HOBS = STAB and disabled if HOBS = WAIT.
(4) Timing source will be NODE_CLOCK if LINK_STATUS_2 is DOWN.

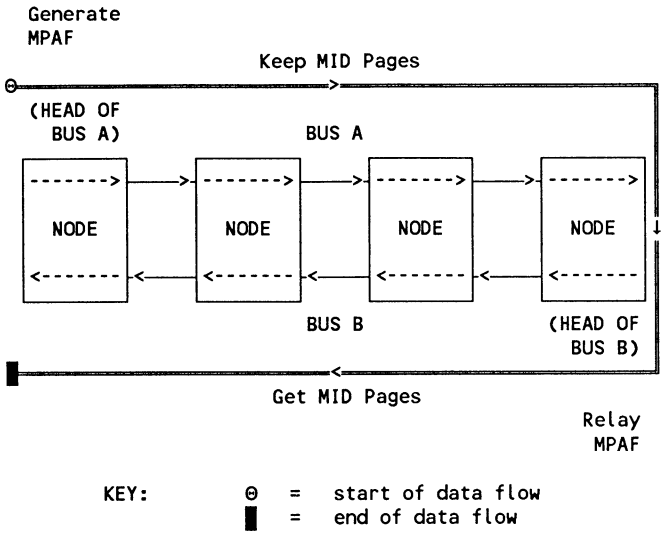


Figure 10-9—MID Page Allocation protocol information flow

The MPAF values are generated by the DQDB Layer Management Protocol Entity at the node that contains the active Head of Bus A function.

— The PCC subfield is generated so that all nodes associate the same MID page value with a given PR subfield. The PCC subfield is generated based on the value of a page counter at the head of Bus A (PAGE_CNTR_HOB, see 7.2.4) under the control of the Page Counter State Machine (PCSM) for the head of Bus A, as described in 10.3.1.
— The PR subfield is generated so that the RESERVED_MID_PAGES (see 7.3.4) number of MID page values that have been reserved for centralized allocation are not available for claiming by nodes on the subnetwork. The PR subfield is generated based on the value of the PAGE_CNTR_HOBA under control of the Page Reservation State Machine for the head of Bus A, as described in 10.3.2.
— The PCM subfield is generated so that a node can check that the MID page value associated by the node with a PR subfield matches the value associated with the PR subfield by the Head of Bus A function. The PCM subfield is generated at the head of Bus A, as described in 10.3.3.

The MPAF passes along Bus A and is operated on by the DQDB Layer Management Protocol Entity of all nodes on the subnetwork. The PCC and PCM subfields are used by the PCSM for Bus A to control the value of the page counter for Bus A (PAGE_CNTR_A, see 7.2.4). The operation of the PCSM is described in 10.3.4. The PR subfield may be used by the Keep Page State Machine (KPSM) at a node to keep or release the MID page value associated with the current value of PAGE_CNTR_A. The operation of the Keep Page State Machine (KPSM) is described in 10.3.5.

Each MPAF value received at the end of Bus A is echoed unchanged in the next MPAF generated for Bus B by the DQDB Layer Management Protocol Entity at the node that contains the active Head of Bus B function.

The MPAF then passes along Bus B and is operated on again by the DQDB Layer Management Protocol Entity of all nodes on the subnetwork. The PCC and PCM subfields are used by the PCSM for Bus B to control the value of the page counter for Bus B (PAGE_CNTR_B, see 7.2.4). The operation of the PCSM is described in 10.3.4. The PR subfield may be used by the Get Page State Machine (GPSM) at a node to obtain the allocation of the MID page value associated with the current value of PAGE_CNTR_B. The operation of the GPSM is described in 10.3.6.

The MPAF values received at the end of Bus B are discarded by the DQDB Layer Management Protocol Entity at the node that contains the active Head of Bus A function.

10.3.1 Page Counter State Machine (PCSM) for head of Bus A

The PCC subfield is normally set by the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function to INCREMENT. After every (MAX_PAGE – MIN_PAGE) = (2^{10} – 2) INCREMENTs, the PCC subfield should be set to RESET, unless MPA operation has been disabled due to the subnetwork undergoing a configuration change affecting the location of a Head of Bus function or by a DQDB Layer Management Interface operation. The PCSM for head of Bus A, defined in figure 10-10, maintains the value of the PAGE_CNTR_HOBA (see 7.2.4) to determine the value of PCC subfield to be generated upon receipt of an EMPTY PCC subfield. The PCSM for head of Bus A shall be automatically disabled upon power-up of the node containing the Default Slot Generator.

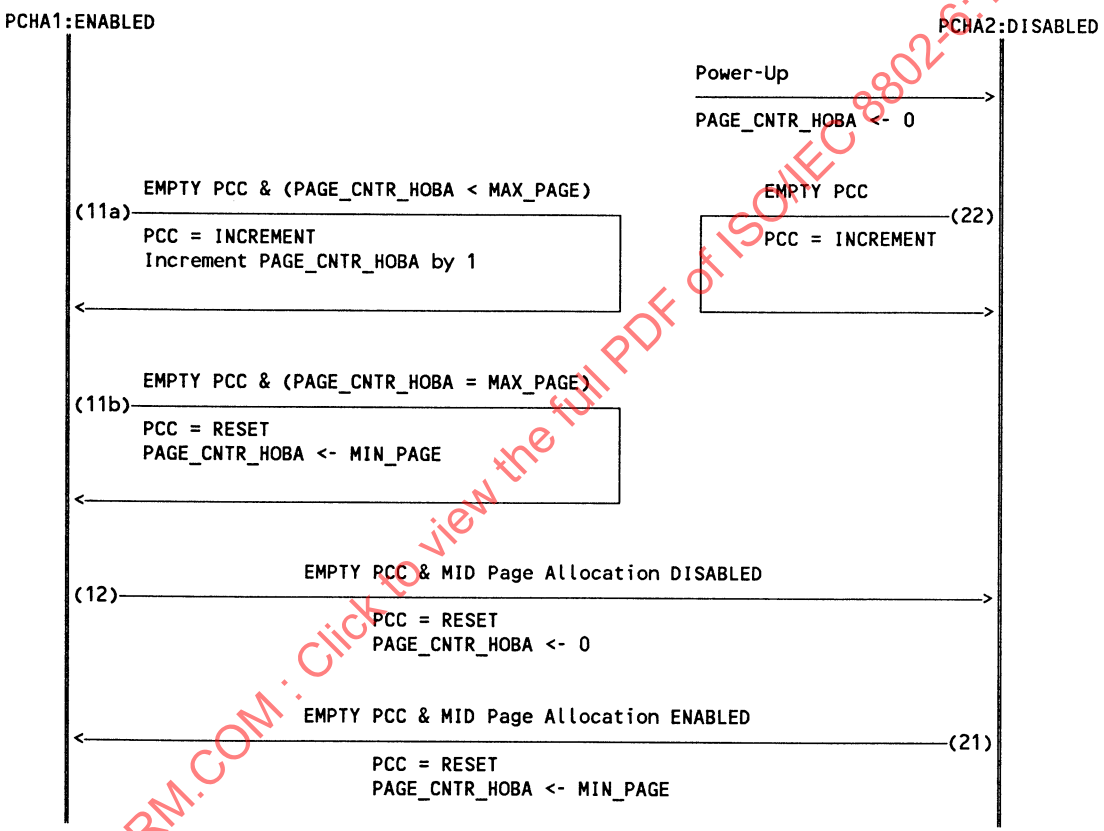


Figure 10-10—Page Counter State Machine (for head of Bus A)

Operation of the PCSM for head of Bus A

State PCHA1: Enabled

The MPA function is enabled when the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function is both sending a HOBS value of STABLE on Bus A and receiving a HOBS value of STABLE from Bus B. Once operation of the MPA function is enabled, the PCSM for head of Bus A

remains in the Enabled state unless MPA operation is disabled. The value of PAGE_CNTR_HOBA is maintained by Transitions 11a and 11b.

Transition 11a

If PAGE_CNTR_HOBA is less than MAX_PAGE = $(2^{10} - 1)$ then, upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity shall set the PCC subfield to INCREMENT and the PCSM for head of Bus A shall increment PAGE_CNTR_HOBA by one.

Transition 11b

If PAGE_CNTR_HOBA is equal to MAX_PAGE = $(2^{10} - 1)$ then, upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity shall set the PCC subfield to RESET and the PCSM for head of Bus A shall set PAGE_CNTR_HOBA to MIN_PAGE = 1.

Transition 12

If operation of the MID Page Allocation function is disabled, then, upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity shall set the PCC subfield to RESET and the PCSM for head of Bus A shall set PAGE_CNTR_HOBA to zero. The PCSM for head of Bus A shall then enter the Disabled state.

State PCHA2: Disabled

The PCSM for head of Bus A enters the Disabled state upon power-up or when operation of the MPA function is disabled. The MPA function is disabled if the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function starts to either send on Bus A or receive from Bus B a HOBS value of WAITING. MPA may be disabled via the DQDB Layer Management Interface.

Transition 21

If operation of the MID Page Allocation function is enabled, then, upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity shall set the PCC subfield to RESET and the PCSM for head of Bus A shall set PAGE_CNTR_HOBA to MIN_PAGE = 1. The PCSM for head of Bus A shall then enter the Enabled state.

Transition 22

In the Disabled state, upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity shall set the PCC subfield to INCREMENT.

10.3.2 Page reservation State Machine for head of Bus A

Upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity at the node with the active head of Bus A function shall generate the PR subfield according to the rules described below, which depend on the relative value of the page counter for head of Bus A, PAGE_CNTR_HOBA (see 7.2.4), and the RESERVED_MID_PAGES system parameter (see 7.3.4):

a) If the current value of the PAGE_CNTR_HOBA is less than or equal to the value of RESERVED_MID_PAGES, then the PR subfield shall be generated with a value of RESERVED.

b) If the current value of the PAGE_CNTR_HOBA is greater than the value of RESERVED_MID_PAGES, then the PR subfield shall be generated with a value of NOT_RESERVED.

The default value of RESERVED_MID_PAGES upon power-up of the node containing the active Head of Bus A function is zero.

Note that the value of PAGE_CNTR_HOBA used for the comparison is the value upon receipt of the PR subfield, i.e., before the counter is changed as a result of the actions described in 10.3.1.

Note also that if the MID Page Allocation function is disabled, according to 10.3.1, then the PR subfield shall be generated with the value of RESERVED, as the value of PAGE_CNTR_HOBA is zero, which is always less than or equal to RESERVED_MID_PAGES. (Note that the MID page value of zero is always reserved, as defined in 6.5.2.1.3.)

10.3.3 Page Counter Modulus Operation for head of Bus A

Upon receipt in a Ph-DATA indication of an EMPTY DQDB_MANAGEMENT octet for which the TYPE bit is to be set to 1, the DQDB Layer Management Protocol Entity at the node with the active Head of Bus A function shall generate the PCM subfield according to the formula below, which depends on the value of the page counter for head of Bus A, PAGE_CNTR_HOBA (see 7.2.4):

PCM_value = (value of PAGE_CNTR_HOBA)(mod 4)

As a typical implementation, the PCM subfield value is set to match the two least significant bits of PAGE_CNTR_HOBA.

Note that the value of PAGE_CNTR_HOBA used is the value upon receipt of the PCM subfield, i.e., before the counter is changed as a result of the actions described in 10.3.1.

Note also that if the MID Page Allocation function is disabled, according to 10.3.1, then the PCM subfield shall be generated with the value of 00, as the value of PAGE_CNTR_HOBA is zero.

10.3.4 Page Counter State Machine (PCSM) for Bus A or Bus B

There is a separate instance of the Page Counter State Machine for each Bus x (PCSM_x) (x = A or B). Each PCSM_x uses the PCC subfield received in an MPAF from Bus x to control the associated page counter for Bus x (PAGE_CNTR_x, see 7.2.4) and uses both the PCC subfield and the PCM subfield to ensure that PAGE_CNTR_x cycles correctly through the MID page range from MIN_PAGE = 1 to MAX_PAGE = $(2^{10} - 1)$. After the operation indicated by the PCC subfield has been performed, the value of PAGE_CNTR_x indicates the MID page value to be associated with the PR subfield of the next MPAF received on Bus x.

The value of the counter PAGE_CNTR_A is used by the KPSM (see 10.3.5) when it operates on the PR subfield in an MPAF received on Bus A. Upon receipt of an MPAF on Bus A, any KPSM operation on the PR subfield and PCSM operation of the PCM subfield shall be completed before the value of PAGE_CNTR_A is modified by the value of the PCC subfield. The KPSM may only use the value of PAGE_CNTR_A when it is greater than zero.

The value of the counter PAGE_CNTR_B is used by the GPSM (see 10.3.6) when it operates on the PR subfield in an MPAF received on Bus B. Upon receipt of an MPAF on Bus B, any GPSM operation on the PR subfield and PCSM operation of the PCM subfield shall be completed before the value of PAGE_CNTR_B is modified by the value of the PCC subfield. The GPSM may only use the value of PAGE_CNTR_B when it is greater than zero.

The operation of an instance of the PCSM for Bus x (x = A or B) is defined in figure 10-11. The PCC subfield may be received from a bus with any one of the values RESET (01), INCREMENT (10), or ERROR (00 or 11) described in 10.1.3.3.

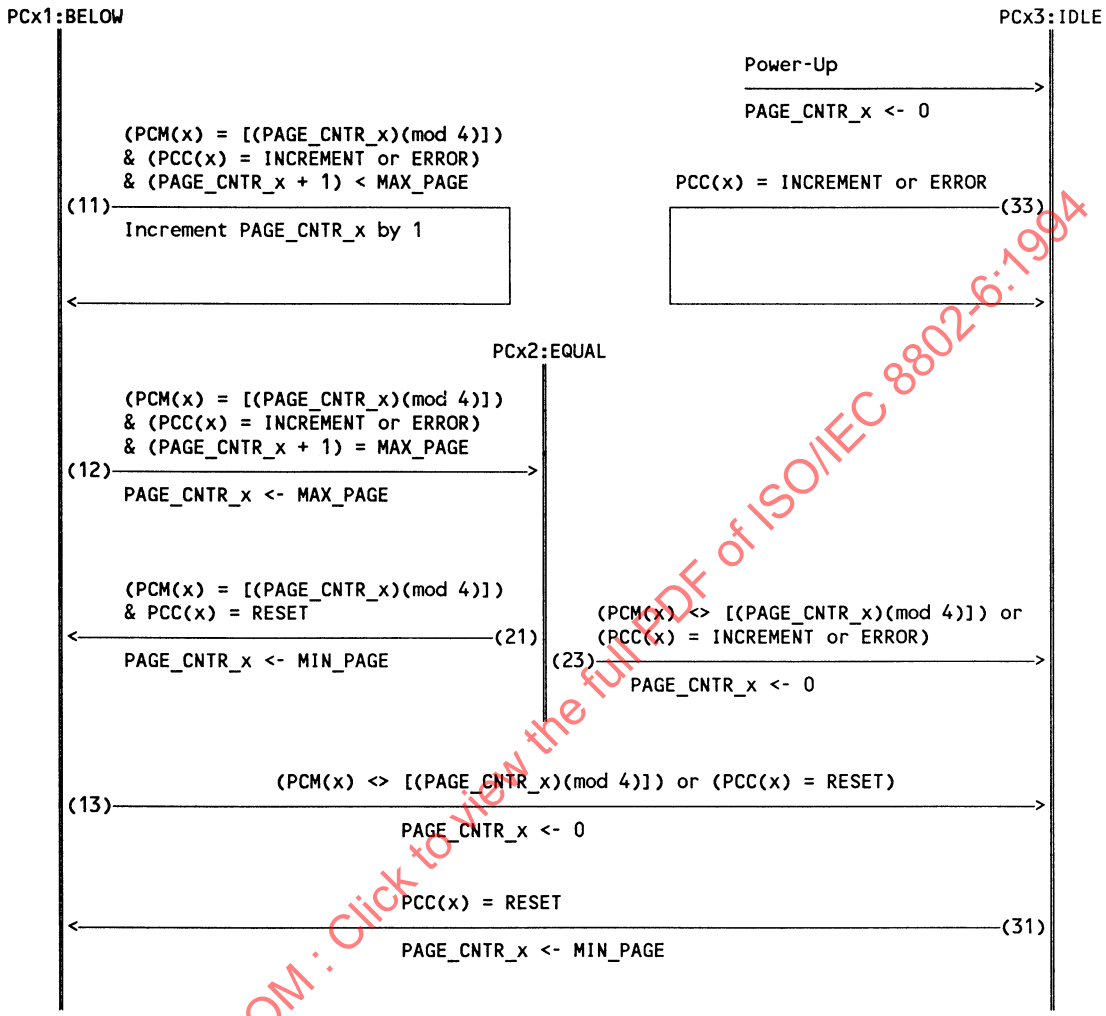


Figure 10-11—Page Counter State Machine for Bus x (PCSM_x)

Operation of the PCSM

The PCSM_x has three states: the Below state, when PAGE_CNTR_x is less than MAX_PAGE; the Equal state, when PAGE_CNTR_x is equal to MAX_PAGE; and the Idle state, when MID Page Allocation has been disabled at the head of Bus A, when an invalid sequence of PCC subfields has been received, or when the node has been powered up. In all states the DQDB Layer Management Protocol Entity shall forward the PCC subfield and the PCM subfield without modification.

At power-up, the PCSM_x shall be in the Idle state. During normal operation, the PCSM_x should remain in the Below and Equal states.

Transitions in the PCSM_x only occur upon receipt of a PCC or PCM subfield. Any changes in the PAGE_CNTR_x caused by a transition in the PCSM_x shall be used by the KPSM (x = A) or GPSM (x = B) for operating on the PR subfield in the next MPAF. Similarly, any changes in the PAGE_CNTR_x shall be used for checking the value of the PCM subfield in the next MPAF.

State PCx1: Below

The PCSM remains in the Below state while MID Page Allocation is enabled, and while PAGE_CNTR_x remains less than MAX_PAGE and synchronized to the correct MID page value. The value of PAGE_CNTR_x is maintained by actions in Transitions 11 and 12.

Transition 13

Upon receipt of an MPAF from Bus x for which the value of the PCM subfield does not match the modulo 4 value of the PAGE_CNTR_x counter, or which contains a PCC subfield set to RESET, PAGE_CNTR_x shall be set to zero and the PCSM_x shall enter the Idle state.

Transitions 11 and 12

Upon receipt of an MPAF from Bus x for which the value of the PCM subfield matches the modulo 4 value of the PAGE_CNTR_x counter, and which contains a PCC subfield set to either INCREMENT or ERROR, the value of PAGE_CNTR_x shall be incremented by one. Then,

Transition 11

If the incremented value of PAGE_CNTR_x remains less than MAX_PAGE = $(2^{10} - 1)$, the PCSM_x remains in the Below state.

Transition 12

If the incremented value of PAGE_CNTR_x is equal to MAX_PAGE = $(2^{10} - 1)$, the PCSM_x enters the Equal state.

State PCx2: Equal

The PCSM_x enters the Equal state when the value of PAGE_CNTR_x equals MAX_PAGE. The PCSM_x makes a transition from the Equal state upon receipt of the next MPAF from Bus x.

Transitions 21 and 23

Upon receipt of the next MPAF from Bus x, the PCSM_x shall make one of the following two transitions:

Transition 21

If the value of the PCM subfield matches the modulo 4 value of the PAGE_CNTR_x counter and the PCC subfield is set to RESET, the value of PAGE_CNTR_x shall be reset to MIN_PAGE = 1 and the PCSM_x shall enter the Below state.

Transition 23

If the value of the PCM subfield does not match the modulo 4 value of the PAGE_CNTR_x counter, or the PCC subfield is set to INCREMENT or ERROR, PAGE_CNTR_x shall be set to zero and the PCSM_x shall enter the Idle state.

State PCx3: Idle

The PCSM_x enters the Idle state at power-up and thereafter does not enter the Idle state during normal operation. The Idle state is only entered after power-up when the PCSM_x loses synchronism with the PCC subfields generated by the DQDB Layer Management Entity at the node with the active Head of Bus A function, or when the node with the active Head of Bus A function has been instructed to disable operation of the MPA function. The PCSM_x returns to the Below state upon Transition 31.

Transitions 31 and 33

Upon receipt of a PCC subfield from Bus x, the PCSM_x makes one of the following two transitions:

Transition 31

If the PCC subfield is set to RESET, PAGE_CNTR_x shall be reset to MIN_PAGE = 1 and the PCSM_x shall enter the Below state.

Transition 33

If the PCC subfield is set to INCREMENT or ERROR, the PCSM_x shall remain in the Idle state.

10.3.5 Keep Page State Machine (KPSM)

The node maintains a list of MID page values that are currently available for use by the MAC Convergence Function block by the set of operations described in 5.4.4.2.1. The KPSM, defined below, controls the MID Page Allocation protocol operation of keeping an MID page value currently in the node MID page list.

The KPSM operates on PR subfields in MPAFs received on Bus A. The counter PAGE_CNTR_A, controlled by the PCSM for Bus A (PCSM_A), is used by the KPSM when it operates on a PR subfield. Upon receipt of an MPAF on Bus A, any KPSM operation on the PR subfield shall be completed before the value of PAGE_CNTR_A is modified by the PCSM_A using the value of the PCC subfield. The KPSM may only use the value of PAGE_CNTR_A when it is greater than zero.

The PR subfield may be received from Bus A with any one of the values NOT_RESERVED (00), RESERVED (11), or ERROR (01 or 10) described in 10.1.3.1.

Operation of the KPSM

If the PAGE_CNTR_A has a value of zero, then, upon receipt of a PR subfield from Bus A, the DQDB Layer Management Protocol Entity shall forward the PR subfield without modification.

If the PAGE_CNTR_A has a value greater than zero, then, upon receipt of a PR subfield from Bus A, the KPSM shall compare the value of PAGE_CNTR_A with the list of node MID page values. Then,

a) If the value of PAGE_CNTR_A is not in the list of node MID page values, the DQDB Layer Management Protocol Entity shall forward the PR subfield without modification.
b) If the value of PAGE_CNTR_A is in the list of node MID page values, the KPSM shall examine the value in the PR subfield. Then,

1) If the PR subfield value is NOT_RESERVED or ERROR, the DQDB Layer Management Protocol Entity shall forward the PR subfield with a value of RESERVED.
2) If the PR subfield value is RESERVED, then,
   i) The KPSM shall generate an LM-EVENT notify (MID_PAGE_LOST) with the mid_page_id parameter set to the value of PAGE_CNTR_A.
   ii) The DQDB Layer Management Entity shall delete the value of PAGE_CNTR_A from the node MID page list.
   iii) The DQDB Layer Management Entity shall inform the MAC Convergence Function block that this MID page value may no longer be used.

10.3.6 Get Page State Machine (GPSM)

The node maintains a list of MID page values that are currently available for use by the MAC Convergence Function block by the set of operations described in 5.4.4.2.1. The node also maintains a transmit sequence number counter (TX_SEQUENCE_NUM, see 7.2.6) for each combination of MID value included in the set represented by the MID page list and VCI value that the node is programmed to receive on behalf of the MCF block. The GPSM, defined in figure 10-12, controls the MID Page Allocation protocol operation of obtaining the allocation of a new MID page value.

The GPSM operates on PR subfields in MPAFs received on Bus B. The counter PAGE_CNTR_B, controlled by the PCSM for Bus B (PCSM_B), is used by the GPSM when it operates on a PR subfield. Upon receipt of an MPAF on Bus B, any GPSM operation on the PR subfield shall be completed before the value of PAGE_CNTR_B is modified by the PCSM_B using the value of the PCC subfield. The GPSM may only use the value of PAGE_CNTR_B when it is greater than zero.

The PR subfield may be received from Bus B with any one of the values NOT_RESERVED (00), RESERVED (11), or ERROR (01 or 10) described in 10.1.3.1.

Operation of the GPSM

The GPSM can be in one of two states: Idle or Getting.

The GPSM shall start in the Idle state at power-up and remain there until the node needs to obtain the allocation of an MID page value. In this case, provided the number of MID page values in the node MID page list is less than the maximum number that the node can obtain, MAX_MID_PAGES (see 7.3.5), the GPSM enters the Getting state and remains there until it obtains the allocation of a page of MIDs.

State GP1: Idle

The GPSM remains in the Idle state until it receives an LM-ACTION invoke (MID_PAGE_GET), requesting the allocation of a page of MIDs.

Transition 11

Upon receipt of a PR subfield on Bus B, the DQDB Layer Management Protocol Entity shall forward the PR subfield without modification.

Transition 12

Upon receipt of an LM-ACTION invoke (MID_PAGE_GET), the GPSM shall enter the Getting state, provided that the number of MID pages in the node MID page list is less than MAX_MID_PAGES.

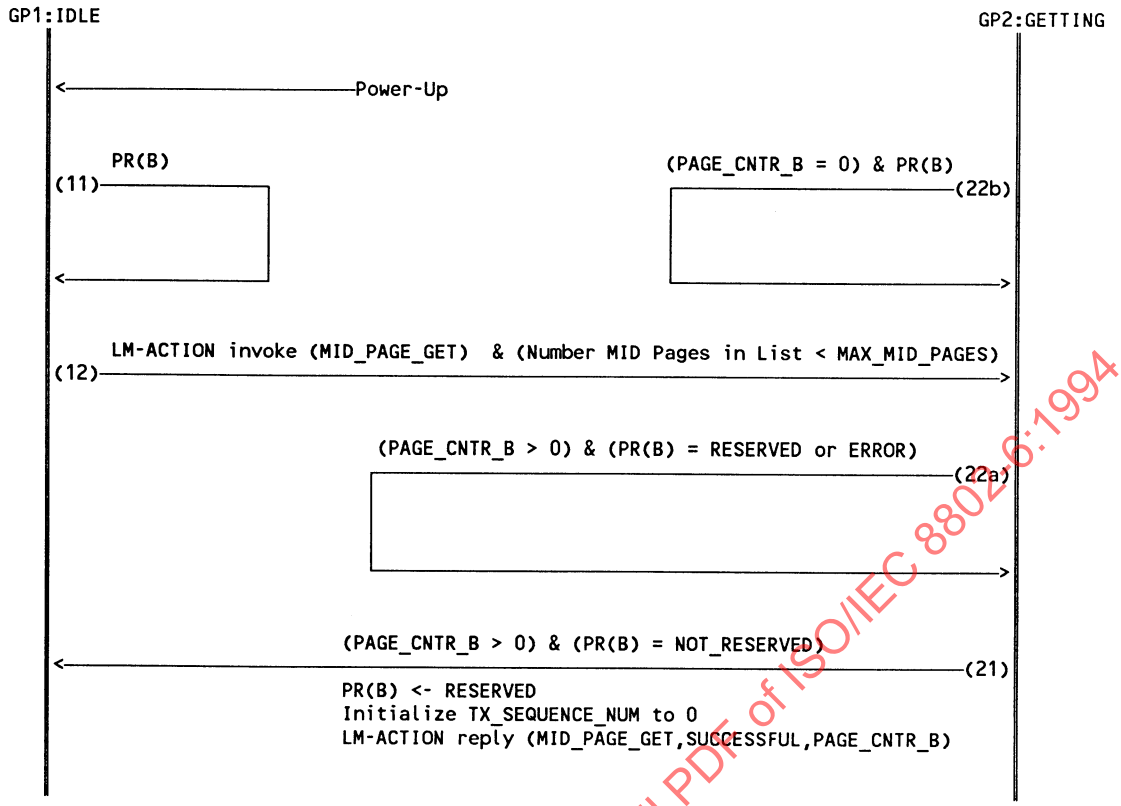


Figure 10-12—Get Page State Machine (GPSM)

State GP2: Getting

In the Getting state, the GPSM shall operate on the PR subfields received from Bus B. The GPSM shall remain in the Getting state until, with value of PAGE_CNTR_B greater than zero, the GPSM detects a PR subfield set to NOT_RESERVED on Bus B.

Transitions 21 and 22a

If the value of PAGE_CNTR_B is greater than zero, then, upon receipt of a PR subfield from Bus B, the GPSM shall examine the value in the PR subfield. Then,

Transition 22a

If the PR subfield value is RESERVED or ERROR, the DQDB Layer Management Protocol Entity shall forward the PR subfield without modification and the GPSM shall remain in the Getting state.

Transition 21

If the PR subfield value is NOT_RESERVED, then,

a) The DQDB Layer Management Protocol Entity shall forward the PR subfield with a value of RESERVED.
b) The GPSM shall generate an LM-ACTION reply (MID_PAGE_GET) with SUCCESSFUL status and with the mid_page_id parameter set to the value of PAGE_CNTR_B.
c) The DQDB Layer Management Entity shall add the value of PAGE_CNTR_B to the node MID page list.
d) The DQDB Layer Management Entity shall inform the MAC Convergence Function block that this MID page value may be used.
e) The MAC Convergence Function block shall initialize to zero each TX_SEQUENCE_NUM associated with each combination of MID value in the MID page, and VCI value that the node is programmed to receive on behalf of the MCF block.

Transition 22b

If the value of PAGE_CNTR_B equals zero, then upon receipt of a PR subfield from Bus B, the DQDB Layer Management Protocol Entity shall forward the PR subfield without modification and the GPSM shall remain in the Getting state.

11. Physical Layer principles of operation

11.1 Architectural considerations

The DQDB Layer uses the services of the Physical Layer defined in clause 4. Different Physical Layers are defined as part of this part of ISO/IEC 8802. Each consists of two protocol functions as follows:

a) A transmission system function defines the characteristics of, and method of attachment to, the transmission link between DQDB nodes. The interface to the transmission system meets with an existing transmission system standard.

b) A Physical Layer convergence function adapts the capabilities of the transmission system into the Physical Layer service. This function is supported by the Physical Layer Convergence Procedure (PLCP), which defines a method of mapping the DQDB Layer timing information, slot octets, and management information octets into a format that is suitable for transfer by the associated transmission system.

Each different transmission system requires the definition of a unique PLCP. If the transmission system already provides the defined Physical Layer service, the Physical Layer convergence function is null.

Due to the diversity of Physical Layer standards, this clause defines a set of principles of operation rather than a set of definitive rules to be followed by all PLCPs. Any function that is defined in this clause must be supported by an appropriate mechanism in each PLCP.

11.2 Node configuration

This part of ISO/IEC 8802 recognizes that implementations may package a number of DQDB nodes sharing common power, management, and control functions into a cluster. In this case the Physical Layer within the cluster may be proprietary. The Physical Layer functions specified in this clause pertaining to node maintenance should apply to a single isolated node and to a cluster.

11.3 Duplex operation of the transmission link

The transmission link between adjacent nodes must be able to support transfer of DQDB Layer timing information, slot octets, and management information octets in both directions simultaneously, so that a complete Dual Bus is operating between the nodes.

The number of parallel transmission paths in the link is dependent upon implementation choices and possibly upon administration requirements. However, the slot rate must be the same for both directions, because stable operation of the Distributed Queue requires that the frequency of Request bits on each bus be equal to or greater than the frequency of empty slots on the other bus. Since the number of Request bits at each priority level is equal to the number of slots, then the slot rates must be the same on the two buses.[58]

The ability of a transmission link to support or not support communication in both directions is indicated to the DQDB Layer by the status parameter in Ph-STATUS indication primitives generated at the Physical Layer service access point associated with the transmission link. (See 4.6.) Each PLCP will include a definition of the conditions under which a transmission link is declared up or down. (See also 11.5.3 and 11.5.4.)

[58] The mechanism that would cause instability in the Distributed Queue for the case of unequal slot rates is described in annex D.

11.3.1 Physical Layer Connection State Machine (PLCSM)

In order to determine the status of a transmission link, the PLCP *may* support the PLCSM. The PLCSM is a *recommended procedure*, and it is not mandatory for a PLCP to support the PLCSM, provided the required function can be supported in an alternative manner.

For PLCPs which support the PLCSM, there is one PLCSM for each transmission link at the node. The PLCSMs are functionally located in the Physical Layer Management Entity. The Physical Layer Management Entities at the two ends of a transmission link communicate via a Physical Layer Management Protocol. The information to support this protocol is carried in the Link Status Signal (LSS), which is defined in abstract form in 11.3.2. The concrete encoding of the LSS by a PLCP is specified in the clause where that PLCP is defined (see, for example, clause 13).

The PLCSM is specified in tabular form in table 11-1. This table specifies the complete set of input conditions which can occur, and gives the output condition for each input condition.

Table 11-1—Physical Layer Connection State Machine (PLCSM)

Input		Output	
Incoming LSS	**PLCSM control**	**Outgoing LSS**	**Status**
connected	NORMAL	connected	UP
--	NORMAL	rx_link_dn	DOWN
rx_link_dn	NORMAL	rx_link_up	DOWN
rx_link_up	NORMAL	connected	UP
X	FORCE_DOWN	rx_link_dn	DOWN

KEY: -- = No input (link failed)
X = Don't care
Status = Passed as parameter in Ph-STATUS indication

The inputs to table 11-1 are as follows:

a) The LSS received from the incoming direction of the transmission link; and
b) the PLCSM Control from the Physical Layer Management Interface. This has the value of either NORMAL or FORCE_DOWN. (See 11.6.1.)

The outputs from table 11-1 are as follows:

a) The LSS to be transmitted on the outgoing direction of the transmission link; and
b) The status parameter which is passed to the DQDB Layer via a Ph-STATUS indication. This has the value of either UP or DOWN.

The PLCSM is also specified in figure 11-1 in the form of a state machine using the representation described in 1.7.2. Input conditions are listed above the transition arrow and output conditions are listed below the transition arrow. Any output condition that is enclosed in braces describes a flag that does not change as a result of the transition.

The specification of the PLCSM in table 11-1 and figure 11-1 are equivalent. In the case of conflict between table 11-1 and figure 11-1, the table shall take precedence.

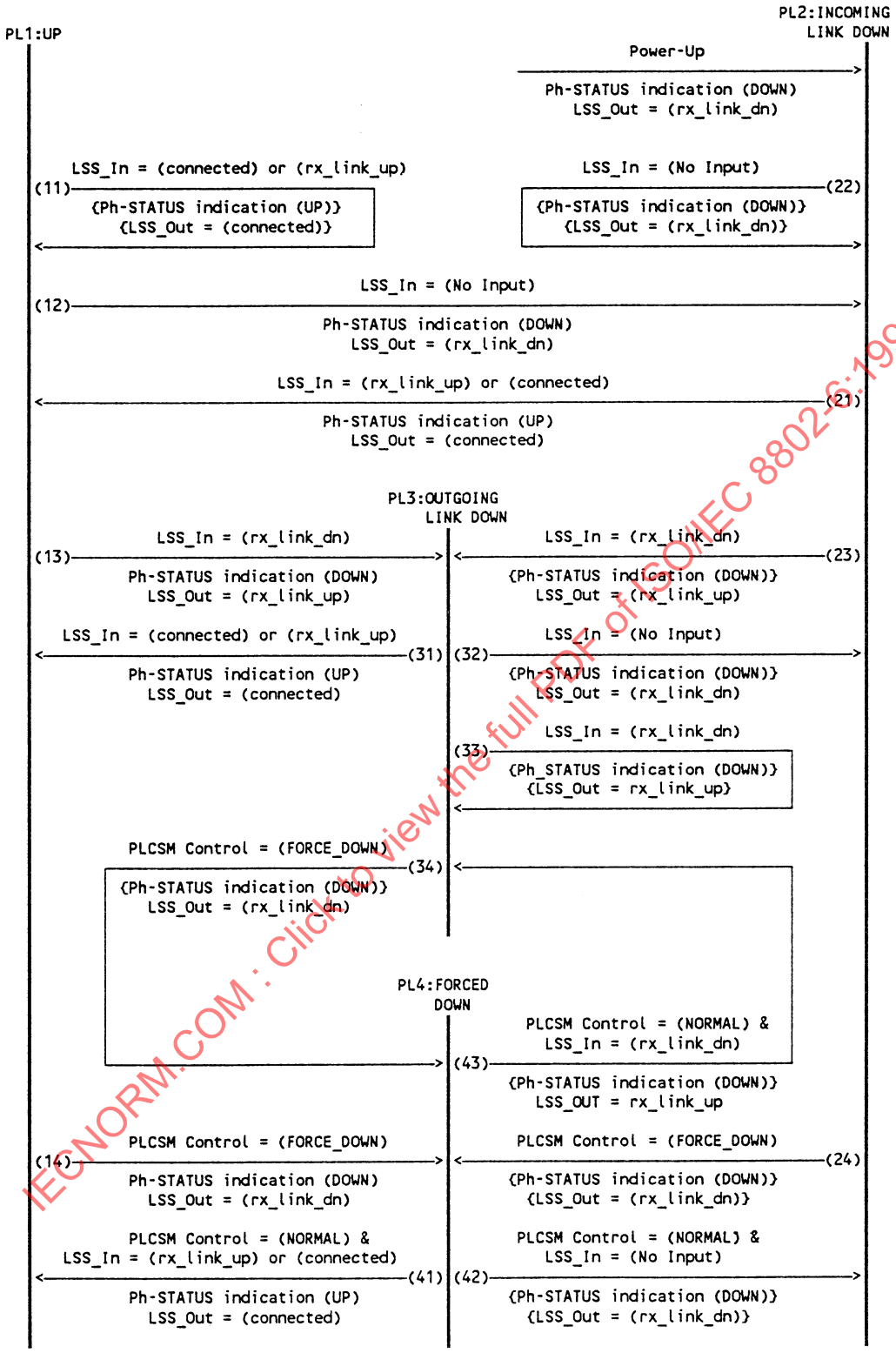


Figure 11-1—Physical Layer Connection State Machine

The PLCSM may be forced by the PLCSM Control input at the Physical Layer Management Interface into a state where the PLCSM protocol entity signals that the incoming link is down, even if the link is actually up as indicated by receipt of LSS = (connected) or (rx_link_up). Forcing a link down might be done for testing or maintenance purposes, or to allow controlled insertion of a node.

When the local node is forcing the link down, then this node is sending LSS = rx_link_dn. Receipt of LSS = rx_link_up at the local node confirms that the remote node now considers the link down. However, confirmation cannot be received unless the link is in fully operable condition. If it is not, then receipt of LSS = rx_link_dn by the local node indicates that the link outgoing from the local node is down, whereas receipt of no LSS by the local node indicates that the incoming link is down.

11.3.2 Link Status Signal (LSS)

The LSS is used to communicate information about the status of a transmission link between the two peer Physical Layer Management Entities. The LSS is defined in abstract form in table 11-2. The concrete encoding of the LSS for a PLCP is specified in the clause where the PLCP is defined.

Table 11-2—Abstract encoding of LSS

LSS name	Link status
connected	Received link connected; Operating in full duplex mode
rx_link_dn	Received link down, no input or PLCSM control = FORCE_DOWN
rx_link_up	Received link up

11.4 Node synchronization

The Physical Layer is responsible for the following:

a) The PLCP shall delineate slots and recognize the management information octets.
b) The Physical Layer shall propagate the DQDB Layer timing along the bus.
c) The PLCP shall limit jitter to levels acceptable to the transmission system.

Each PLCP clause should define the tolerance allowed on the node clock that is used to provide 125 μs timing in the absence of an external timing source.

11.5 Physical Layer Maintenance functions

11.5.1 Fault detection within nodes

The Physical Layer subsystem at a node shall monitor octets passing into and out of the local DQDB Layer subsystem of the node whenever it is inserted into the subnetwork. This monitoring shall establish that the node is operating in accordance with the requirements of 4.3, which specifies the relationship between octets passed in Ph-DATA indication primitives, and octets received in Ph-DATA request primitives.

11.5.2 Node isolation

A node is isolated from the subnetwork when its ability to write onto the buses is disabled or bypassed. Node bypass could be provided via a number of techniques, including the following:

a) Physical bypass using passive techniques, such as a passive optical bypass switch.
b) Physical bypass using active techniques, which require termination and regeneration of the transmission system. In order to minimize reconfiguration in this case, the PLCP shall maintain the same regeneration function during node isolation as when the node is in service.
c) Isolation of the access control functions of the node.

Each PLCP shall specify a bypass function. The Physical Layer could use the bypass function to isolate a node from the subnetwork in the following conditions:

a) When the node is powered down (which could correspond to a) or c) above), unless power is provided to the Physical Layer to directly connect the two transmission links at the node (which could correspond to b) above);
b) When powered-up, until the node is synchronized to the subnetwork (which could correspond to b) above);
c) When the Physical Layer determines that the access control functions for that node are corrupting the subnetwork (which could correspond to c) above); or
d) When the Physical Layer Management Entity forces isolation.

11.5.3 Fault detection on transmission links

The Physical Layer shall monitor the error performance of the incoming transmission link to establish that the transmission system is providing an acceptable grade of service. The error thresholds at which the status of a link is declared to have changed depend upon the transmission system and medium used, the network requirements, and the nature of the PLCP. The thresholds for each situation shall be defined as parameters passed to the Physical Layer Management Entity.

11.5.4 Nodes not supporting Head of Bus functions

If a node detects an incoming transmission link failure on Bus x (x = A or B), and the node does not support Head of Bus functions, then the node shall signal this fact to the next node downstream on Bus x. This signal is described in more detail in each PLCP clause. (This function is intended to allow network management functions to isolate the transmission link that has actually failed.)

As well as generating a Ph-STATUS indication (DOWN) at the Ph-SAP corresponding to the failed link, the Physical Layer also generates a Ph-STATUS indication (DOWN) at the other Ph-SAP of the node.

A node that receives the indication that there is no upstream Head of Bus capability shall behave as if the incoming transmission link had failed, and take appropriate action, depending on whether or not the node has Head of Bus capability.

In summary, all PLCP definitions shall support a mechanism for differentiating between failure of a link due to conditions defined in 11.5.3 and failure due to lack of upstream Head of Bus capability. The signaling of lack of Head of Bus capability could take the form of an unframed PLCP "Jam" signal (for example, see 13.4).

11.6 Physical Layer facilities

11.6.1 Physical Layer Connection State Machine (PLCSM) Control Flag

If a PLCP definition supports the PLCSM defined in 11.3.1, the PLCP requires two PLCSM Control Flags at each node: one for each transmission link at the node. Each PLCSM Control Flag is used to control the operation of the PLCSM related to the transmission link. The PLCSM Control Flag shall contain one of these values:

NORMAL
FORCE_DOWN

The value NORMAL directs the PLCSM to operate normally. The value FORCE_DOWN is used to direct the PLCSM to signal to the peer Physical Layer Management Entity that the transmission link has a status of DOWN.[59]

11.6.2 Head of Bus Capable Flag (HOB_CAPABLE)

The Head of Bus Capable Flag (HOB_CAPABLE) is used by the Physical Layer to determine the action to take when one of the transmission links at the node is determined to have a status of DOWN. The HOB_CAPABLE Flag shall contain one of these values:

YES
NO

The value YES indicates that the node is capable of performing Head of Bus functions. The value NO indicates that the node is not capable of performing Head of Bus functions.

[59] This might be done for testing or maintenance purposes, or to allow controlled insertion of a node.

12. Physical Layer Convergence Procedure (PLCP) for DS1-based systems

This clause has not yet been approved as an International Standard, but will be incorporated at a later date.[60]

[60]The current text of this PLCP can be found in IEEE Std 802.6c-1993.

13. PLCP for DS3-based systems

13.1 Overview

This clause provides a convergence procedure in which the DQDB Layer is mapped into a standard DS3 transmission system. An additional application, intended primarily for applications within a single building or campus environment, which uses the same 12 slot per 125 µs PLCP frame format, denoted SDS3, will be described in a future addition to this part of ISO/IEC 8802. This latter application provides the capability to map directly into a fiber optic transmission system with the same effective data rate. A single DQDB subnetwork may thus contain nodes with a mixture of DS3 and SDS3 transmission systems interfaces.

The DS3 PLCP makes use of the optional status parameter in Ph-DATA indication primitives. (See 4.2.) Hence, the status parameter is mandatory for the service provided by the DS3 PLCP.

13.1.1 DS3 relationship to the PLCP

The rate, format, electrical characteristics, and other attributes of the unchannelized DS3 signal are defined in ANSI T1.102 and ANSI T1.107. The use of the DS3 overhead bits (i.e., F, M, P, X, and C bits) is defined in ANSI T1.107. This part of ISO/IEC 8802 does not preclude or endorse any C-bit application (i.e., SYNTRAN or C-bit Parity). The implementer, however, is warned of possible interworking problems for nodes using differing C-bit applications.

The DS3 signal has a nominal line rate of 44.736 Mbit/s and a frame duration that is nominally 106.4 µs. The DS3 signal consists of 699 octets (5592 bits or 1398 nibbles) per 125 µs time period. One bit in 85 is used for DS3 overhead functions providing a nominal information payload rate of 44.210 Mbit/s (84/85 × 44.736 Mbit/s) and leaving approximately 690.78 octets (5526.2 bits or 1381.6 nibbles) available for use by the DS3 PLCP. The DS3 information payload shall be *nibble aligned* (i.e., a nibble boundary shall follow each DS3 overhead bit).

13.1.2 SDS3 fiber extension of simplified DS3 signal transmission system relationship to the PLCP

The attributes of the underlying transmission system for the SDS3 application will be described in a future addition to this part of ISO/IEC 8802.[61]

13.2 The PLCP frame format

The PLCP frame format consists of 12 rows by 57 octets with the last row containing a trailer of either 13 or 14 nibbles. The PLCP frame has a nominal duration of 125 µs.

The PLCP frame format is asynchronously mapped into the DS3 information payload or SDS3 signal using a nibble stuffing technique. A nibble-stuffing opportunity occurs once every 375 µs to maintain a nominal frame repetition rate of 125 µs. Since the DS3 PLCP frame duration is 125 µs and the nibble-stuffing opportunity occurs once every 375 µs, the DS3 PLCP frame contains a cycle/stuff counter to indicate the phase of the 375 µs cycle. Within the stuffing opportunity cycle, there are three PLCP frames. The first frame in the cycle contains a trailer of 13 nibbles, the second frame in the cycle contains a trailer of 14 nibbles, and the third frame in the cycle contains either 13 or 14 nibbles, depending on whether a nibble stuff has occurred. The stuffing opportunity cycle is controlled by the cycle/stuff counter, which is explained further in 13.3.3.5.

[61] SDS3 Transmission System Coding Specifications and SDS3 Transmission System Fiber Optic Specifications will be described in subsequent additions to this International Standard.

The PLCP delivers 12 DQDB slots every 125 µs interval to the DQDB Layer. The first three columns of the PLCP frame format are used for framing. The fourth column is used to carry PLCP Path Overhead octets.

The complete frame structure is shown in figure 13-1. Each row of bits in the PLCP frame format illustrated in figure 13-1 shall be transmitted in order, from left to right, top to bottom.

←1→	←1→	←1→	←1→	←53 octets→	
A1	A2	P11	Z6	First DQDB slot	
A1	A2	P10	Z5	DQDB slot	
A1	A2	P9	Z4	DQDB slot	
A1	A2	P8	Z3	DQDB slot	
A1	A2	P7	Z2	DQDB slot	
A1	A2	P6	Z1	DQDB slot	
A1	A2	P5	F1	DQDB slot	
A1	A2	P4	B1	DQDB slot	
A1	A2	P3	G1	DQDB slot	
A1	A2	P2	M2	DQDB slot	
A1	A2	P1	M1	DQDB slot	13–14 nibbles
A1	A2	P0	C1	Last DQDB slot	

125 µs

KEY: A1 = Framing octet (11110110)
A2 = Framing octet (00101000)
P11 – P0 = Path Overhead Identifier octets

PLCP Path Overhead octets:

Z6 – Z1 = Growth octets
F1 = PLCP Path user channel
B1 = BIP-8
G1 = PLCP Path status
M2 – M1 = DQDB Layer Management Information octets
C1 = Cycle/stuff counter

Figure 13-1—The DS3 PLCP frame format

13.3 PLCP field definitions

(Refer to figure 13-1.) See 6.1 for ordering principles of fields. The values of fields are described as bit patterns. The left-most bit of each octet is the most significant.

13.3.1 Framing octets (A1, A2)

The first two columns (A1, A2) are used to provide slot delineation. The encoding of the A1 and A2 octets is shown in table 13-1.

NOTE—Alternatively, slot delineation can be provided by using the Header Check Sequence (HCS) (see 14.3, 15.3, and 16.3). Since the PLCP for DS3-based systems was originally created by a different standards body than the other PLCPs, this possibility was not included, although it is also technically feasible in this case.

Table 13-1—A1 and A2 codes

A1	A2
11110110	00101000

These codes are the same pattern as used in the Synchronous Digital Hierarchy (SDH) CCITT Recommendations G.707, G.708, and G.709. See 13.6 for PLCP framing requirements.

13.3.2 Path Overhead Identifier (P11–P0)

The third column (P11–P0) identifies the PLCP Overhead octets contained in the fourth column of figure 13-1. Figure 13-2 shows the format of the Path Overhead Identifier (POI) octet. The left-most 6 bits of these octets provide numbering of the 12 rows. The reserved bit shall be set to (0). The parity bit provides odd parity over this field.

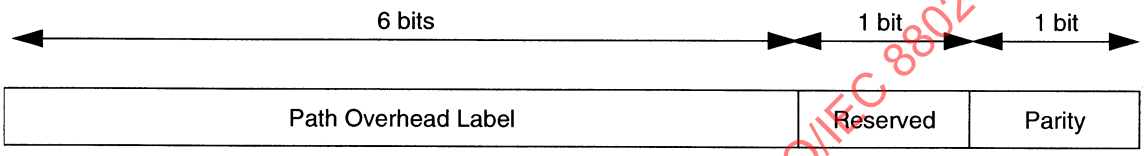


Figure 13-2—POI octet

Table 13-2 defines the codes for P11–P0, which shall be generated by the PLCP at each node. All other codes are invalid. See 13.6 for PLCP framing requirements.

Table 13-2—POI codes

P11	001011	0	0
P10	001010	0	1
P9	001001	0	1
P8	001000	0	0
P7	000111	0	0
P6	000110	0	1
P5	000101	0	1
P4	000100	0	0
P3	000011	0	1
P2	000010	0	0
P1	000001	0	0
P0	000000	0	1

13.3.3 PLCP Path Overhead octets

The PLCP Path is defined between two adjacent peer PLCP entities. The F1, B1, G1, and C1 PLCP Path Overhead octets are related to PLCP operation and shall be terminated/generated at each PLCP on the subnetwork. The M1 and M2 octets are provided for the transport of DQDB Layer Management Information octets and shall not be processed by the PLCP Sublayer.

13.3.3.1 PLCP Path user channel (F1)

The F1 octet is the user channel, which is allocated for user communication purposes between adjacent PLCPs. The use of this octet in DQDB subnetworks is for further study.[62] The default code for this octet shall be (00000000).

13.3.3.2 Bit Interleaved Parity-8 (B1)

One octet is allocated for PLCP Path error monitoring. This function shall be a Bit Interleaved Parity-8 (BIP-8) code using even parity. The PLCP Path BIP-8 is calculated over the 12 × 54 octet structure (columns 4 to 57, PLCP Path Overhead octets and DQDB slot octets) of the previous PLCP frame and inserted into the B1 octet of the current PLCP frame.

A BIP-8 is an 8-bit code in which the first bit of the BIP-8 code represents even parity calculated over the first bit of each octet in the 12 × 54 octet structure, the second bit of the BIP-8 code calculates even parity over the second bit of each octet in the 12 × 54 octet structure, etc. Therefore, the BIP-8 code provides for 8 separate even parity codes covering the corresponding bit of each octet in the 12 × 54 octet structure.

13.3.3.3 PLCP Path status (G1)

The G1 octet is allocated to convey the received PLCP status and performance back to the transmitting PLCP. This octet permits the status and performance of the full duplex PLCP Path to be monitored at either PLCP entity. See figure 13-3 for an illustration of the G1 octet.

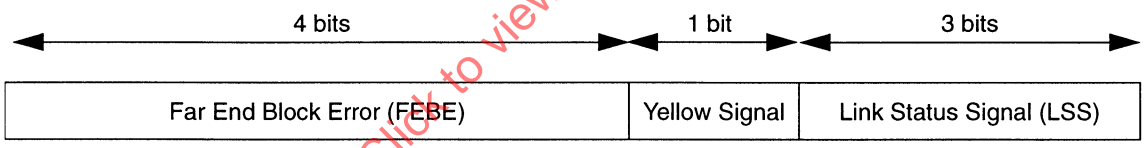


Figure 13-3—PLCP Path status (G1)

The first four bits of the G1 octet are the Far End Block Error (FEBE) code, which should be used to convey the count of interleaved-bit blocks that have been detected to be in error by the PLCP BIP-8 code in the preceding frame. If implemented, this count shall have nine legal codes, namely zero (0000) to eight (1000) errors. If not implemented, the code shall be (1111). The remaining six possible codes (1001 through 1110) would have been a result of an error condition and shall be interpreted as zero errors.

The fifth bit is used for the Yellow Signal. The Yellow Signal alerts the transmitting PLCP that a received failure indication has been declared along the PLCP Path. When an incoming failure (i.e., PLCP Loss-Of-Frame) is detected on Bus x (x = A or B) that persists for 2.5 ± 0.5 s, a Yellow Signal may be generated on Bus y (y = B or A) by setting the fifth bit of the G1 octet to one (1). The Yellow Signal shall be detected by a one (1) in the fifth bit of the G1 octet for ten consecutive frames. When the incoming failure has ceased for 15 ± 5 s, a Yellow Signal is removed from Bus y by setting the fifth bit of the G1 octet to zero (0). Removal

[62] One possible application is for assisting maintenance personnel.

of the Yellow Signal shall be detected by a zero (0) in the fifth bit of the G1 octet for ten consecutive frames. If not implemented, the default code for the Yellow Signal shall be (0).

The remaining three bits shall be used for the Link Status Signal (LSS) code as described in 11.3.2. The LSS is used to communicate information about the status of the transmission link between the two adjacent PLCP entities. This information is conveyed only between these two entities.

The LSS codes for the G1 octet are shown in table 13-3. All other codes are invalid and shall be ignored by the receiver.

Table 13-3—LSS codes

LSS code	LSS name	Link status
000	connected	Received link connected
011	rx_link_dn	Received link down, no input or FORCE_DOWN
110	rx_link_up	Received link up

13.3.3.4 DQDB Layer Management Information octets (M2, M1)

The octets M1 and M2 carry the DQDB Layer Management Information octets, which are described in 10.1. The DQDB Layer Management Information octets shall be generated at the head of a bus as described in 4.2, and shall be operated on by the DQDB Layer Management Protocol Entity as described in 5.4.3.3, 10.2, and 10.3. There need be no correlation between TYPE = 0 or 1 octets and the M1 or M2 octets.

13.3.3.5 Cycle/stuff counter (C1)

The C1 octet provides a stuffing opportunity cycle indicator for the PLCP frames. C1 shall be used as an indication of the phase of the 375 μs stuffing opportunity cycle and shall be used as a stuff indicator. Thus, the C1 octet indicates the PLCP frame in which a nibble-stuffing opportunity shall occur.

The first PLCP frame in this cycle shall contain 13 nibbles, the second frame shall contain 14 nibbles, and the third frame in the cycle shall provide for an opportunity to nibble stuff. The C1 octet shall indicate whether the third frame in the cycle has or has not been stuffed. Each PLCP entity on the subnetwork shall be *frame cycle aligned* (i.e., the phase of the incoming frame in the cycle shall be the same as the phase of the outgoing frame in the cycle).

The HOB shall generate the frame phase of the cycle. The External Timing Source shall nibble stuff given the opportunity to stuff in order to provide for a nominal 125 μs time period. When the HOB is using its own local clock to generate the timing information, the HOB shall nibble stuff every other opportunity. The encoding of this octet is shown in table 13-4.

13.3.3.6 Growth octets (Z6–Z1)

The octets Z1, Z2, Z3, Z4, Z5, and Z6 are reserved for future standardization. The PLCP shall encode these octets to the default code of (00000000).

13.3.4 Trailer nibbles

Each of the 13 or 14 nibbles in the trailer shall be encoded as (1100).

Table 13-4—Cycle/stuff counter codes

C1 code	Frame phase of cycle	Trailer length
11111111	1	13
00000000	2	14
01100110	3 (no stuff)	13
10011001	3 (stuff)	14

13.4 PLCP behavior during faults

There are three types of conditions that directly influence the operation of the PLCP: transmission system faults, PLCP faults, and DQDB Layer out-of-service. The PLCP shall not differentiate between transmission system faults and PLCP faults. Therefore, all transmission system faults shall force the PLCP Out-Of-Frame.

When the PLCP declares a PLCP Out-Of-Frame condition (see 13.6), it shall start the Timer_P_x (x = A or B). This timer (Timer_P_x) shall be set to 1 ms ± 10 μs. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall generate a Jam signal on Bus x. The Jam signal is defined as an unframed continuous bit pattern contained within the framed DS3 information payload or SDS3 signal. The bit pattern will be a repeating (1100) sequence (starting with a one–one (11) after each DS3 overhead bit for the DS3 application).

If the PLCP detects the Jam signal for at least 12 μs, the PLCP shall reset and start the Timer_P_x. The PLCP shall continue to send the Jam signal on Bus x and shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

— If the PLCP enters the In-Frame state before the timer, Timer_P_x, expires, the timer shall be stopped. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as VALID at Ph-SAP_x. The PLCP shall resume its normal operation of processing/generating PLCP framing and PLCP Path Overhead octets and of performing its local stuffing.
— If the timer expires before the detection of valid PLCP framing octets, the PLCP Sublayer shall enter the Loss-Of-Frame state and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. The PLCP shall transmit on Bus y (y = B or A) an LSS equal to rx_link_dn.
  - If the DQDB Layer is capable of becoming HOB (i.e., HOB_CAPABLE Flag is set to YES; see 11.6.2), then when the DQDB Layer receives the Ph-STATUS indication equal to DOWN at the Ph-SAP_x, the DQDB Layer shall start the Head of Bus Arbitration Timer, Timer_H_w (w = 1 or 2), as defined in 7.1.2, and shall send the HOBS value of WAITING, as defined in 10.2.3.4, on Bus x. Thus, the PLCP shall generate a valid PLCP frame with the appropriate PLCP framing octets and PLCP Path Overhead octets and shall perform the cycle/stuffing mechanism.
  - If the DQDB Layer is not capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to NO; see 11.6.2), the PLCP Layer shall continue to generate the Jam signal on Bus x.

13.5 PLCP behavior during DQDB Layer out-of-service

The Physical Layer subsystem of a DQDB node connected to a DS3-based transmission system shall always be powered up in normal operation. However, if for some reason the Physical Layer subsystem of a DQDB node is powered down, the stations downstream and upstream of this node would immediately detect this condition as a transmission system fault and the DQDB subnetwork would begin the fault detection process, as defined in 13.4, to reconfigure around the powered-down node.

The DQDB Layer shall have the ability of going in and out of service without interrupting the operation of the Physical Layer. When the DQDB Layer goes out of service, the PLCP, as well as the transmission system, shall continue normal operation (i.e, perform the stuffing mechanism and process/generate the PLCP Path Overhead octets). The DQDB Layer Management Information octets (M1 and M2) shall be relayed unmodified through the PLCP.

13.6 PLCP framing

The transition diagram for the PLCP Framing State Machine is defined in figure 13-4. Each DQDB node has two PLCP Framing State Machines, one at the receiver for each bus.

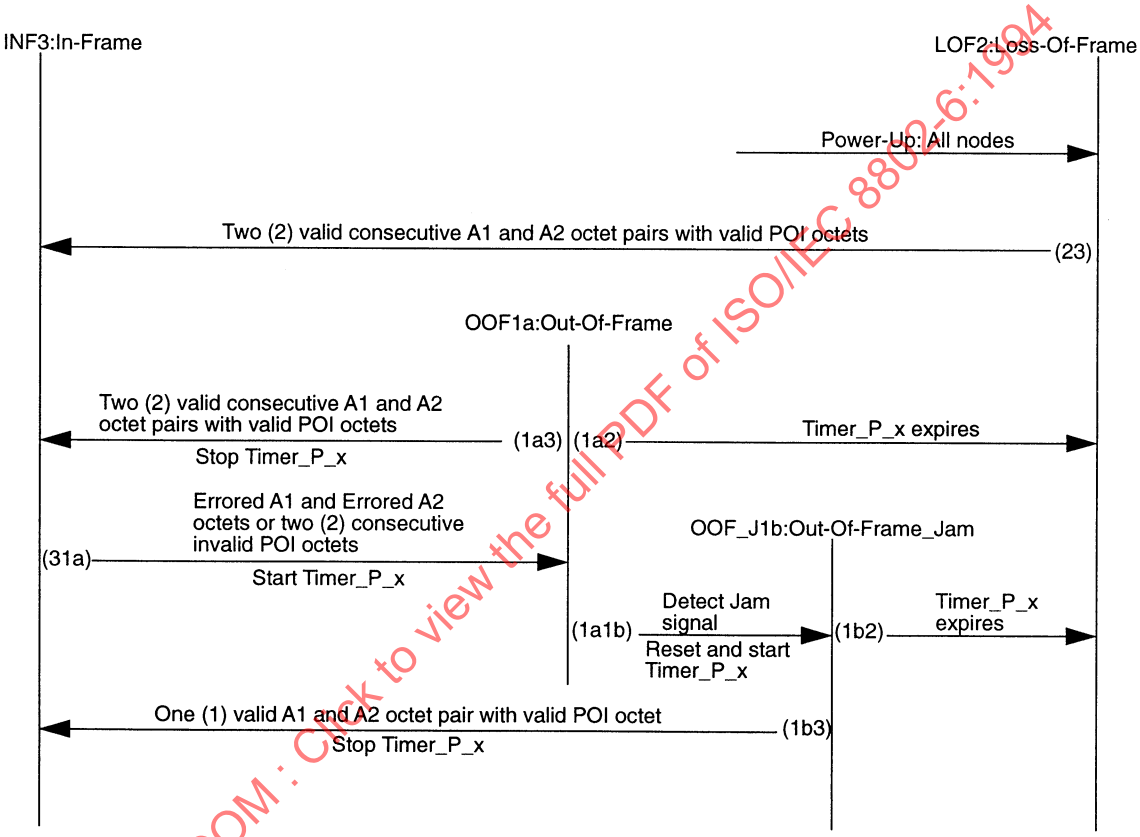


Figure 13-4—PLCP Framing State Machine transition diagram

PLCP Framing transition diagram

The state machine can be in one of four states: In-Frame (INF3), Out-Of-Frame (OOF1a), Out-Of-Frame_Jam (OOF_J1b), and Loss-Of-Frame (LOF2). The state machine is powered up in the Loss-Of-Frame (LOF2) state.

State OOF1a: Out-Of-Frame

When entering this state, the PLCP Out-Of-Frame timer, Timer_P_x, shall be started, and the PLCP shall start generating the Jam signal on Bus x. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x (see 7.5.2), the PLCP shall continue to generate a framed PLCP signal on Bus x. The PLCP shall remain in this state until the Timer_P_x expires, the Jam signal is detected on Bus x, or until

valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

Transition 1a3

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential Path Overhead Identifier (POI) octets, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

Transition 1a2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1a1b

If the PLCP detects a Jam signal for at least 12 μs, then the state machine shall enter the OOF_J1b state. The PLCP shall reset and start the timer, Timer_P_x.

State OOF_J1b: Out-Of-Frame_Jam

The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall continue to generate the Jam signal. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x, then the PLCP shall continue to generate a framed PLCP signal.

Transition 1b2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1b3

If the PLCP detects one valid A1 and A2 octet pair with a valid POI octet, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

State LOF2: Loss-Of-Frame

When entering this state, the PLCP shall transmit on Bus y an LSS equal to rx_link_dn and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to YES, the PLCP shall generate a framed PLCP signal on Bus x starting with an A1 and A2 framing pattern sequence. If the HOB_CAPABLE Flag is set to NO, the PLCP shall generate the Jam signal on Bus x. The PLCP shall remain in this state until valid PLCP framing is found.

Transition 23

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential POI octets, then the state machine shall enter the INF3 state. The PLCP shall send to the DQDB Layer a Ph-STATUS indication equal to UP at Ph-SAP_x.

State INF3: In-Frame

The PLCP shall process/generate PLCP framing octets, POI octets, and PLCP Path Overhead octets as in normal operations and shall perform the stuffing mechanism. When the PLCP enters this state, it shall always begin transmission of the PLCP frame on Bus x with an A1 and A2 framing pattern sequence, and the PLCP shall send to the DQDB Layer Ph-DATA indication octets of type DQDB_MANAGEMENT (M2 and

M1 octets) marked as INVALID at Ph-SAP_x until the PLCP detects the P11 octet. After the PLCP has detected the P11 octet, the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. The PLCP shall remain in this state until errored A1 and errored A2 octets or two consecutive invalid or nonsequential POI octets are detected.

Transition 31a

If the PLCP detects one or more errors in the A1 octet, and one or more errors in the A2 octet of an A1 and A2 octet framing pair or two consecutive invalid or nonsequential POI octets, then the state machine shall enter the OOF1a state. The PLCP shall start the Timer_P_x.

13.6.1 LSS operations table

The operations table for the LSS is defined in table 13-5. The operations table determines the status of the transmission link according to the state of the PLCP Framing State Machine, the incoming LSS, and the Physical Layer Connection State Machine (PLCSM) control. This table supplements table 11-1. Additional states from table 11-1, corresponding to rows 4–6, are highlighted in boldface font.

Table 13-5—LSS operations table

INPUT				OUTPUT	
PLCP Frame State	**PLCSM Control**	**Incoming LSS**	**Detect Jam**	**Ph-STATUS**	**Outgoing LSS**
INF3	NORMAL	Connected	X	UP	connected
INF3	NORMAL	rx_link_up	X	UP	connected
INF3	NORMAL	rx_link_dn	X	DOWN	rx_link_up
OOF1a/OOF_J1b	**NORMAL**	**X**	**X**	**no change**	**rx_link_up**
LOF2	**NORMAL**	**X**	**NO**	**DOWN**	**rx_link_dn**
LOF2	**NORMAL**	**X**	**YES**	**DOWN**	**rx_link_up**
X	FORCE_DN	X	X	DOWN	rx_link_dn

KEY: X = Don't care

If a DQDB node with HOB_CAPABLE Flag set to NO receives an LSS code equal to rx_link_dn on Bus x, then the PLCP shall transmit on Bus x an LSS code equal to rx_link_dn, irrespective of the incoming LSS code on Bus y. This node adjacent to a failure would, therefore, be isolated from the DQDB subnetwork.

13.6.2 Physical Layer Frame Timing operations table

The Physical Layer frame timing operations table, table 13-6, determines whether an unframed Jam signal or a framed PLCP signal shall be transmitted on Bus x (x = A or B) and which 125 µs timing shall be used in the latter case to generate the framed PLCP signal. The transmission on Bus x is dependent on three inputs:

a) The state of the PLCP Framing State Machine for both incoming buses (see figure 13-4).
b) The Timing Source and HOB_OPERATION_z (z = 1, 2, or Default) outputs from the CC_z operations tables in the DQDB Layer, defined in tables 10-10b), 10-11, and 10-12, respectively.
c) The value of the HOB_CAPABLE Flag, defined in 11.6.2.

This table supplements the Timing Source information of tables 10-10b), 10-11, and 10-12.

Table 13-6—Physical Layer Frame Timing operations supplementary table

INPUT			OUTPUT	
Bus x PLCP Frame State	**HOB OPERATION_z**	**HOB_CAPABLE Flag**	**Bus x Tx Frame**	**125 μs Timing**
OOF1a/ OOF_J1b	not HEAD_ OF_BUS_x	X	Jam	N/A
OOF1a/ OOF_J1b	HEAD_OF_ BUS_x	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)
LOF2	not HEAD_ OF_BUS_x	NO	Jam	N/A
LOF2	X	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)

KEY: X = Don't care
N/A = Not applicable

NOTES

1—Selection between NODE_CLOCK and EXTERNAL_CLOCK is determined by tables 10-10b), 10-11, and 10-12 when the Bus x PLCP Frame State is OOF1a, OOF_J1b, or LOF2.

2—The HOB_OPERATION_z column refers to the value of HOB_OPERATION for the CC_z associated with Ph-SAP_x at the node.

14. PLCP for CCITT Recommendation G.703 (2.048 Mbit/s)

14.1 Overview

This clause provides a convergence procedure in which the DQDB Layer is mapped into a standard transmission system according to CCITT Recommendations G.704 and G.703 operating at 2.048 Mbit/s as used in public networks. Beyond the provisions of CCITT Recommendations G.704 and G.703, a 2.375 ms framing period shall be provided to support transportation of full DQDB slots.

NOTE—The CCITT Recommendation G.704 structure itself provides a 125 µs frame and Physical Layer management information.

The E1 PLCP[63] makes use of the optional status parameter in Ph-DATA indication primitives. (See 4.2.) Hence, the status parameter is mandatory for the service provided by the E1 PLCP.

14.1.1 E1 relationship to the PLCP

The rate, format, electrical characteristics, and other attributes of the E1 signal shall be as defined in CCITT Recommendations G.704 and G.703. The first and 17th octet (time slots 0 and 16) of each CCITT Recommendation G.704 frame shall not be used for the PLCP. They are left for E1 synchronization (i.e., frame alignment) and overhead bits compatible with existing equipment.

Therefore, the net bit rate available to the PLCP is 1.920 Mbit/s. The CCITT Recommendation G.704 nominal frame rate is 8 kHz.

NOTE—The PLCP shall provide sufficient buffering or other provisions to accommodate the "jump" resulting from the passing of the 2 unavailable octets.

The 2.375 ms PLCP frame shall be aligned with the 125 µs (8 kHz) frame of the transmission system. Therefore, every second 2.375 ms frame starts with a synchronization octet of an E1 frame. Nineteen CCITT Recommendation G.704 frames are contained in one PLCP frame of 2.375 ms. In such a PLCP frame of 608 octets, 570 octets are available to the PLCP.

14.2 The PLCP frame format

A frame duration of 2.375 ms is chosen to allow efficient mapping of DQDB slots into the E1 frames based on octets. Four octets shall be added to each DQDB slot of 53 octets to provide framing and overhead functions so that each row contains 57 octets. (The first three octets are used for framing. The fourth octet is used to carry PLCP Path Overhead octets.) Ten of these 57 octet rows are placed into one 2.375 ms frame. No trailing octets are provided. (They are not needed as the CCITT Recommendation G.704 systems are synchronous.)

The complete frame structure is shown in figure 14-1. Each row of bits in the PLCP frame format illustrated in figure 14-11 shall be transmitted in order, from left to right, top to bottom.

14.3 PLCP field definitions

(Refer to figure 14-1.) See 6.1 for ordering principles of fields. The values of fields are described as bit patterns. The left-most bit of each octet is the most significant.

---

[63]The designation E1 is used for a 2.048 Mbit/s transmission system according to CCITT Recommendations G.703 and G.704. It is the first level of the plesiochronous digital hierarchy.

←1→	←1→	←1→	←1→	←53 octets→
A1	A2	P9	Z4	First DQDB slot
A1	A2	P8	Z3	DQDB slot
A1	A2	P7	Z2	DQDB slot
A1	A2	P6	Z1	DQDB slot
A1	A2	P5	F1	DQDB slot
A1	A2	P4	B1	DQDB slot
A1	A2	P3	G1	DQDB slot
A1	A2	P2	M2	DQDB slot
A1	A2	P1	M1	DQDB slot
A1	A2	P0	C1	Last DQDB slot

2.375 ms

KEY: A1 = Framing octet (11110110)
A2 = Framing octet (00101000)
P9 – P0 = Path Overhead Identifier octets

PLCP Path Overhead octets:

Z4 – Z1 = Growth octets
F1 = PLCP Path user channel
B1 = BIP-8
G1 = PLCP Path status
M2 – M1 = DQDB Layer Management Information octets
C1 = Stuff counter

Figure 14-1—The PLCP frame format

14.3.1 Framing octets (A1, A2)

The first two columns (A1, A2) may be used to provide slot delineation. The encoding of the A1 and A2 octets is shown in table 14-1.

These codes are the same pattern as used in the Synchronous Digital Hierarchy (SDH) CCITT Recommendations G.707, G.708, and G.709. See 14.6 for PLCP framing requirements.

Table 14-1—A1 and A2 codes

A1	A2
11110110	00101000

Alternatively, slot delineation based on the Header Check Sequence (HCS) of the DQDB header may be used.

14.3.2 Path Overhead Identifier (P9–P0)

The third column (P9–P0) identifies the PLCP Overhead octets contained in the fourth column of figure 14-1. Figure 14-2 shows the format of the Path Overhead Identifier (POI) octet. The left-most 6 bits of these octets provide numbering of the 10 rows. The reserved bit shall be set to (0). The parity bit provides odd parity over this field.

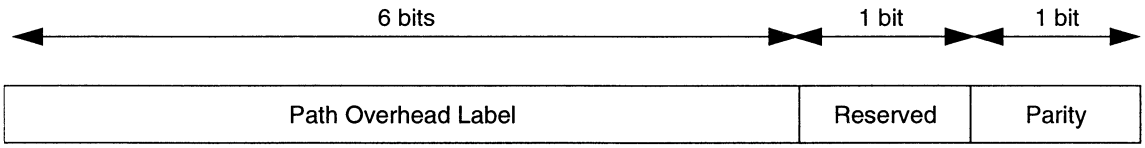


Figure 14-2—POI octet

Table 14-2 defines the codes for P9–P0, which shall be generated by the PLCP at each node. All other codes are invalid. A code shall also be considered invalid if the parity bit contained in the LSB is not correct. The response to invalid codes is described in 14.6.

Table 14-2—E1 POI codes

P9	001001	0	1
P8	001000	0	0
P7	000111	0	0
P6	000110	0	1
P5	000101	0	1
P4	000100	0	0
P3	000011	0	1
P2	000010	0	0
P1	000001	0	0
P0	000000	0	1

14.3.3 PLCP Path Overhead octets

The PLCP Path is defined between two adjacent peer PLCP entities. The F1, B1, G1 and C1 PLCP Path Overhead octets are related to PLCP operation and shall be terminated/generated at each PLCP on the sub-network. The M1 and M2 octets are provided for the transport of DQDB Layer Management Information octets and shall not be processed by the PLCP Sublayer.

14.3.3.1 PLCP Path user channel (F1)

The F1 octet is the user channel, which is allocated for user communication purposes between adjacent PLCPs. The use of this octet in DQDB subnetworks is for further study.[64] The default code for this octet shall be (00000000).

[64]One possible application is for assisting maintenance personnel.

14.3.3.2 Bit Interleaved Parity-8 (B1)

One octet is allocated for PLCP Path error monitoring. This function shall be a Bit Interleaved Parity-8 (BIP-8) code using even parity. The PLCP Path BIP-8 is calculated over the 10 × 54 octet structure (columns 4 to 57, 1 PLCP Path Overhead octet and 53 DQDB slot octets per row) of the previous PLCP frame and inserted into the B1 octet of the current PLCP frame.

A BIP-8 is an 8-bit code in which the first bit of the BIP-8 code represents even parity calculated over the first bit of each octet in the 10 × 54 octet structure, the second bit represents even parity over the second bit of each octet in the 10 × 54 octet structure, etc. Therefore, the BIP-8 code provides for 8 separate even parity codes covering the corresponding bit of each octet in the 10 × 54 octet structure.

14.3.3.3 PLCP Path status (G1)

The G1 octet is allocated to convey the received PLCP status and performance back to the transmitting PLCP. This octet permits the status and performance of the full duplex PLCP path to be monitored at either PLCP entity. Figure 14-3 depicts the G1 octet.

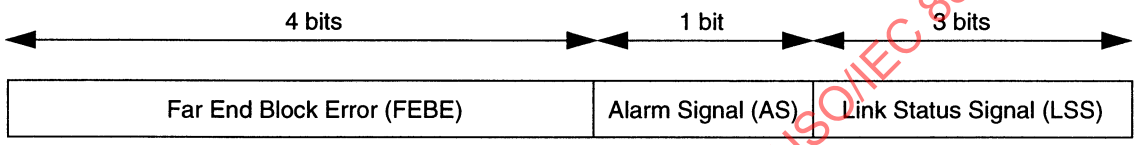


Figure 14-3—PLCP Path status (G1)

The first four bits of the G1 octet are the Far End Block Error (FEBE) code, which should be used to convey the count of interleaved-bit blocks that have been detected to be in error by the BIP-8 code in the preceding frame. If implemented, this count shall have nine legal codes namely zero (0000) to eight (1000) errors. If not implemented, the code shall be (1111). The remaining six possible codes (1001 through 1110) would have been the result of an error condition and shall be interpreted as zero errors.

The fifth bit may be used for the Alarm Signal (AS). The AS alerts the transmitting PLCP that a received failure indication has been declared along the PLCP Path. When an incoming failure (i.e., PLCP Loss-Of-Frame) is detected on Bus x (x = A or B) that persists for 2.5 ± 0.5 s, an AS shall be generated on Bus y (y = B or A) by setting the fifth bit of the G1 octet to one (1). The AS shall be detected by a one (1) in the fifth bit of the G1 octet for ten consecutive frames. When the incoming failure has ceased for 15 ± 5 s, the AS shall be removed from Bus y by setting the fifth bit of the G1 octet to zero (0). Removal of the AS shall be detected by a zero (0) in the fifth bit of the G1 octet for ten consecutive frames. If the AS is not implemented, the default code for the AS shall be (0).

The remaining three bits shall be used for the Link Status Signal (LSS) as described in 11.3.2. The LSS is used to communicate information about the status of the transmission link between two adjacent PLCP entities. This information is conveyed only between these two entities.

The LSS codes for the G1 octet are shown in table 14-3. All other codes are invalid and shall be ignored by the receiver.

14.3.3.4 DQDB Layer Management Information octets (M2, M1)

The octets M1 and M2 carry the DQDB Layer Management Information octets, which are described in 10.1. The DQDB Layer Management Information octets shall be generated at the head of bus as described in 4.2,

Table 14-3—LSS codes

LSS code	LSS name	Link status
000	connected	Received link connected
011	rx_link_dn	Received link down, no input or FORCE_DOWN
110	rx_link_up	Received link up

and shall be operated on by the DQDB Layer Management Protocol Entity as described in 5.4.3.3, 10.2, and 10.3. There need be no correlation between TYPE = 0 or 1 octets and the M1 and M2 octets.

14.3.3.5 Stuff counter (C1)

The C1 octet provides a stuffing indicator for the PLCP frames. It is not used for the E1 PLCP and shall be encoded to the default code of (00000000).

14.3.3.6 Growth octets (Z4–Z1)

The octets Z1, Z2, Z3, and Z4 are reserved for future standardization. The PLCP shall encode these octets to the default code of (00000000).

14.3.4 Trailer octets

There are no trailer octets for the E1 PLCP.

14.4 PLCP behavior during faults

There are three types of conditions that directly influence the operation of the PLCP: transmission system faults, PLCP faults, and DQDB Layer out-of-service. The PLCP shall not differentiate between transmission system faults and PLCP faults. Therefore, all transmission system faults shall force the PLCP Out-Of-Frame.

When the PLCP declares a PLCP Out-Of-Frame condition (see 14.6), it shall start the Timer_P_x (x = A or B). This timer (Timer_P_x) shall be set to 19 ms ± 0.2 ms. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall generate a Jam signal on Bus x. The Jam signal is defined as an unframed continuous bit pattern contained within the framed E1 information payload. The bit pattern shall be a repeating (1100) sequence (starting with a one–one (11) after each E1 synchronization octet for the E1 application).

NOTE—The Jam signal is not needed in point-to-point configurations. It shall, however, be generated in any configuration.

If the PLCP detects the Jam signal for at least 270 μs, the PLCP shall reset and start the Timer_P_x. The PLCP shall continue to send the Jam signal on Bus x and shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

— If the PLCP enters the In-Frame state before the timer, Timer_P_x, expires, the timer shall be stopped. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as VALID at Ph-SAP_x. The PLCP shall resume its normal operation of processing/generating PLCP framing and PLCP Path Overhead octets.

— If the timer expires before the detection of valid PLCP framing octets, the PLCP Sublayer shall enter the Loss-Of-Frame state and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. The PLCP shall transmit on Bus y (y = B or A) an LSS equal to rx_link_dn.
  - If the DQDB Layer is capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to YES), then when the DQDB Layer receives the Ph-STATUS indication equal to DOWN at the Ph-SAP_x, the DQDB Layer shall start the Head of Bus Arbitration Timer, Timer_H_w (w = 1 or 2), as defined in 7.1.2, and shall send the HOBS value of WAITING, as defined in 10.2.3.4, on Bus x. Thus the PLCP shall generate a valid PLCP frame with the appropriate PLCP framing octets and PLCP Path Overhead octets.
  - If the DQDB Layer is not capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to NO), the PLCP Layer shall continue to generate the Jam signal on Bus x.

14.5 PLCP behavior during DQDB Layer out-of-service

The Physical Layer subsystem of a DQDB node connected to an E1-based transmission system shall always be powered up in normal operation. However, if for some reason the Physical Layer subsystem of a DQDB node is powered down, the stations downstream and upstream of this node can quickly detect this condition as a transmission system fault and the DQDB subnetwork would begin the fault detection process, as defined in 14.4, to reconfigure around the powered-down node.

The DQDB Layer shall have the ability of going in and out of service without interrupting the operation of the Physical Layer. When the DQDB Layer goes out-of-service the PLCP as well as transmission system, shall continue normal operation (i.e., process/generate the PLCP Path Overhead octets). The DQDB Layer Management Information octets (M1 and M2) shall be relayed unmodified through the PLCP.

14.6 PLCP framing

The transition diagram for the PLCP Framing State Machine is defined in figure 14-4. Each DQDB node has two PLCP Framing State Machines, one at the receiver for each bus.

PLCP Framing transition diagram

The state machine can be in one of four states: In_Frame (INF3), Out-Of-Frame (OOF1a), Out-Of-Frame_Jam (OOF_J1b), and Loss-Of-Frame (LOF2). The state machine is powered up in the Loss-of-Frame (LOF2) state.

State OOF1a: Out-Of-Frame

When entering this state, the PLCP Out-Of-Frame timer, Timer_P_x, shall be started, and the PLCP shall start generating the Jam signal on Bus x. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x (see 7.5.2), the PLCP shall continue to generate a framed PLCP signal on Bus x. The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

Transition 1a3

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential Path Overhead Identifier (POI) octets, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

Transition 1a2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

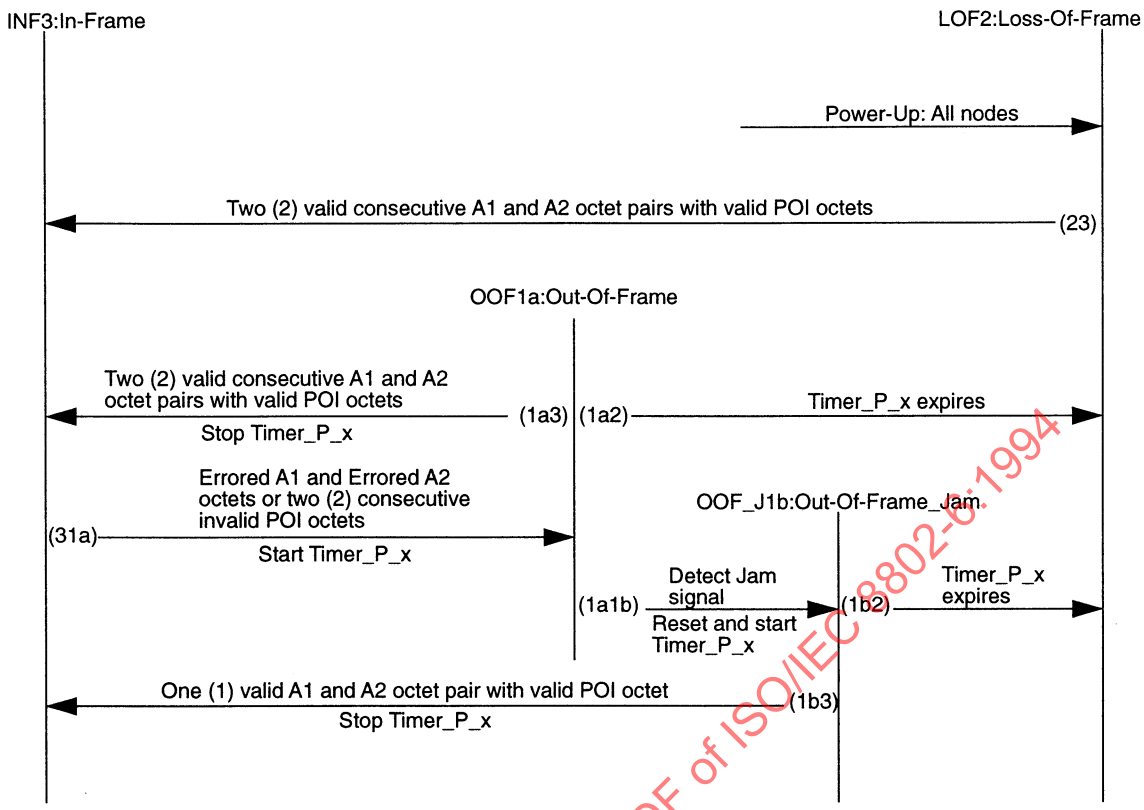


Figure 14-4—PLCP Framing State Machine transition diagram

Transition 1a1b

If the PLCP detects a Jam signal for at least 270 µs, then the state machine shall enter the OOF_J1b state. The PLCP shall reset and start the timer, Timer_P_x.

State OOF_J1b: Out-Of-Frame_Jam

The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall continue to generate the Jam signal. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x, then the PLCP shall continue to generate a framed PLCP signal.

Transition 1b2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1b3

If the PLCP detects one valid A1 and A2 octet pair with a valid POI octet, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

State LOF2: Loss-Of-Frame

When entering this state, the PLCP shall transmit on Bus y an LSS equal to rx_link_dn and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to

YES, the PLCP shall generate a framed PLCP signal on Bus x starting with an A1 and A2 framing pattern sequence. If the HOB_CAPABLE Flag is set to NO, the PLCP shall generate the Jam signal on Bus x. The PLCP shall remain in this state until valid PLCP framing is found.

Transition 23

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential POI octets, then the state machine shall enter the INF3 state. The PLCP shall send to the DQDB Layer a Ph-STATUS indication equal to UP at Ph-SAP_x.

State INF3: In-Frame

The PLCP shall process/generate PLCP framing octets, POI octets, and PLCP Path Overhead octets as in normal operations. When the PLCP enters this state, it shall always begin transmission of the PLCP frame on Bus x with an A1 and A2 framing pattern sequence, and the PLCP shall send to the DQDB Layer Ph-DATA indication octets of type DQDB_MANAGEMENT (M2 and M1 octets) marked as INVALID at Ph-SAP_x until the PLCP detects the P9 octet. After the PLCP has detected the P9 octet, the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. The PLCP shall remain in this state until errored A1 and errored A2 octets or two consecutive invalid or non-sequential POI octets are detected.

Transition 31a

If the PLCP detects one or more errors in the A1 octet, and one or more errors in the A2 octet of an A1 and A2 octet framing pair or two consecutive invalid or nonsequential POI octets, then the state machine shall enter the OOF1a state. The PLCP shall start the Timer_P_x.

14.6.1 LSS operations table

The operations table for the LSS is defined in table 14-4. The operations table determines the status of the transmission link according to the state of the PLCP Framing State Machine, the incoming LSS, and the Physical Layer Connection State Machine (PLCSM) control. This table supplements table 11-1. Additional states from table 11-1, corresponding to rows 4 to 6, are highlighted in boldface font.

Table 14-4—LSS operations table

INPUT				OUTPUT	
PLCP Frame State	**PLCSM Control**	**Incoming LSS**	**Detect Jam**	**Ph-STATUS**	**Outgoing LSS**
INF3	NORMAL	Connected	X	UP	connected
INF3	NORMAL	rx_link_up	X	UP	connected
INF3	NORMAL	rx_link_dn	X	DOWN	rx_link_up
OOF1a/OOF_J1b	**NORMAL**	**X**	**X**	**no change**	**rx_link_up**
LOF2	**NORMAL**	**X**	**NO**	**DOWN**	**rx_link_dn**
LOF2	**NORMAL**	**X**	**YES**	**DOWN**	**rx_link_up**
X	FORCE_DN	X	X	DOWN	rx_link_dn

KEY: X = Don't care

If a DQDB node with HOB_CAPABLE Flag set to NO receives an LSS code equal to rx_link_dn on Bus x, then the PLCP shall transmit on Bus x an LSS code equal to rx_link_dn, irrespective of the incoming LSS code on Bus y. This node adjacent to a failure would, therefore, be isolated from the DQDB subnetwork.

14.6.2 Physical Layer Frame Timing operations table

The Physical Layer Frame Timing operations table, table 14-5, determines whether an unframed Jam signal or a framed PLCP signal will be transmitted on Bus x (x = A or B) and which 125 µs timing shall be used in the latter case to generate the framed PLCP signal. The transmission on Bus x is dependent on three inputs:

a) The state of the PLCP Framing State Machine for both incoming buses (see figure 14-4).
b) The Timing Source and HOB_OPERATION_ z (z = 1, 2, or Default) outputs from the CC_z operations tables in the DQDB Layer, defined in tables 10-10b), 10-11, and 10-12, respectively.
c) The value of the HOB_CAPABLE Flag, defined in 11.6.2.

This table supplements the Timing Source information of tables 10-10b), 10-11, and 10-12.

Table 14-5—Physical Layer Frame Timing operations supplementary table

INPUT			OUTPUT	
Bus x PLCP Frame State	**HOB OPERATION_z**	**HOB_CAPABLE Flag**	**Bus x Tx Frame**	**125 µs Timing**
OOF1a/ OOF_J1b	not HEAD_ OF_BUS_x	X	Jam	N/A
OOF1a/ OOF_J1b	HEAD_OF_ BUS_x	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)
LOF2	not HEAD_ OF_BUS_x	NO	Jam	N/A
LOF2	X	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)

KEY: X = Don't care
N/A = Not applicable

NOTES

1—Selection between NODE_CLOCK and EXTERNAL_CLOCK is determined by tables 10-10b), 10-11, and 10-12 when the Bus x PLCP Frame State is OOF1a, OOF_J1b, or LOF2.

2—The HOB_OPERATION_z column refers to the value of HOB_OPERATION for the CC_z associated with Ph-SAP_x at the node.

15. PLCP for CCITT Recommendations G.751 and G.703 (34.368 Mbit/s)

15.1 Overview

This clause provides a convergence procedure in which the DQDB Layer is mapped into a standard transmission system according to CCITT Recommendations G.751 and G.703 operating at 34.368 Mbit/s as used in public networks. Beyond the provisions of CCITT Recommendations G.751 and G.703, a 125 μs framing period shall be provided to support $n \times 64$ kbit/s channels based on octets.

The E3 PLCP[65] makes use of the optional status parameter in Ph-DATA indication primitives. (See 4.2.) Hence, the status parameter is mandatory for the service provided by the E3 PLCP.

15.1.1 E3 relationship to the PLCP

The rate, format, electrical characteristics and other attributes of the E3 signal shall be as defined in CCITT Recommendations G.751 and G.703. The first 2 octets of each CCITT Recommendation G.751 frame shall not be used for the PLCP. They are left for E3 synchronization (i.e., frame alignment) and overhead bits compatible with existing equipment.

NOTES

1—Only 12 bits of the first two octets are used for the Synchronization Pattern, Alarm Indication to the Remote End (A) and National Use (N). The remaining 4 bits shall be set to (1100). Thus the two octets will be (1111010000 A N 1100). The "sliding" of the two octets through the 125 μs frame shall be done via full octets.

Thus the net bit rate available to the PLCP is 34.010 Mbit/s. The nominal frame rate is 22.375 kHz.

2—The PLCP shall provide sufficient buffering or other provisions to accommodate the "jump" resulting from the skipping of the 2 unavailable octets.

The 125 μs frame is not in any way aligned with the 44.6927 μs (22.375 kHz) frame of the transmission system. The E3 payload slots are only octet aligned to the E3 overhead octets.

15.2 The PLCP frame format

A PLCP frame duration of 125 μs is chosen to easily accommodate $n \times 64$ kbit/s channels based on octets. Four octets shall be added to each DQDB slot of 53 octets to provide framing and overhead functions so that each row contains 57 octets. (The first three octets are used for framing. The fourth octet is used to carry PLCP Path Overhead octets.) Nine of these 57 rows are placed into one 125 μs PLCP frame. Eighteen to 20 octets (nominal) shall be left as a trailer depending on whether 2 or 3 E3 overhead double octets occurred during the 125 μs frame.

The complete frame structure is shown in figure 15-1. Each row of bits in the PLCP frame format illustrated in figure 15-1 shall be transmitted in order, from left to right, top to bottom.

15.3 PLCP field definitions

(Refer to figure 15-1.) See 6-1 for ordering principles of fields. The values of fields are described as bit patterns. The left-most bit of each octet is the most significant.

[65]The designation E3 is used for a 34.368 Mbit/s transmission system according to CCITT Recommendation G.751. It is the third level of the plesiochronous digital hierarchy.

←1→	←1→	←1→	←1→	←53 octets→	
A1	A2	P8	Z3	First DQDB slot	
A1	A2	P7	Z2	DQDB slot	
A1	A2	P6	Z1	DQDB slot	
A1	A2	P5	F1	DQDB slot	
A1	A2	P4	B1	DQDB slot	
A1	A2	P3	G1	DQDB slot	
A1	A2	P2	M2	DQDB slot	
A1	A2	P1	M1	DQDB slot	18–20 octets
A1	A2	P0	C1	Last DQDB slot	Trailer

125 µs

KEY: A1 = Framing octet (11110110)
A2 = Framing octet (00101000)
P8 – P0 = Path Overhead Identifier octets

PLCP Path Overhead octets:

Z3 – Z1 = Growth octets
F1 = PLCP Path user channel
B1 = BIP-8
G1 = PLCP Path status
M2 – M1 = DQDB Layer Management Information octets
C1 = Stuff counter

Figure 15-1—The E3 PLCP frame format

15.3.1 Framing octets (A1, A2)

The first two columns (A1, A2) may be used to provide slot delineation. The encoding of the A1 and A2 octets is shown in table 15-1.

These codes are the same patterns as used in the Synchronous Digital Hierarchy (SDH) CCITT Recommendations G.707, G.708, and G.709. See 15.6 for PLCP framing requirements.

Table 15-1—A1 and A2 codes

A1	A2
11110110	00101000

Alternatively, slot delineation based on the Header Check Sequence (HCS) of the DQDB segment header may be used.

15.3.2 Path Overhead Identifier (P8–P0)

The third column (P8–P0) identifies the PLCP Overhead octets contained in the fourth column of figure 15-1. Figure 15-2 shows the format of the Path Overhead Identifier (POI) octet. The left-most 6 bits of these octets provide numbering of the 9 rows. The reserved bit shall be set to (0). The parity bit provides odd parity over this field.

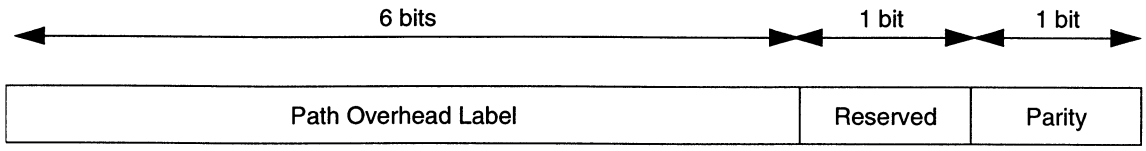


Figure 15-2—POI octet

Table 15-2 defines the codes for P8–P0, which shall be generated by the PLCP at each node. All other codes are invalid. A code shall also be considered invalid if the parity bit is not correct. The response to invalid codes is described in 15.6.

Table 15-2—POI codes

P8	001000	0	0
P7	000111	0	0
P6	000110	0	1
P5	000101	0	1
P4	000100	0	0
P3	000011	0	1
P2	000010	0	0
P1	000001	0	0
P0	000000	0	1

15.3.3 PLCP Path Overhead octets

The PLCP Path is defined between two adjacent peer PLCP entities. The F1, B1, G1, and C1 PLCP Path Overhead octets are related to PLCP operation and shall be terminated/generated at each PLCP on the subnetwork. The M1 and M2 octets are provided for the transport of DQDB Layer Management Information octets and shall not be processed by the PLCP Sublayer.

15.3.3.1 PLCP Path user channel (F1)

The F1 octet is the user channel, which is allocated for user communication purposes between adjacent PLCPs. The use of this octet in DQDB subnetworks is under study.[66] The default code for this octet shall be (00000000).

15.3.3.2 Bit Interleaved Parity-8 (B1)

One octet is allocated for PLCP Path error monitoring. This function shall be a Bit Interleaved Parity-8 (BIP-8) code using even parity. The PLCP Path BIP-8 is calculated over the 9×54 octet structure (columns 4 to 57, 1 PLCP Path Overhead octet and 53 DQDB slot octets per row) of the previous PLCP frame and inserted into the B1 octet of the current PLCP frame.

[66]One possible application is for assisting maintenance personnel.

A BIP-8 is an 8-bit code in which the first bit of the BIP-8 code represents even parity calculated over the first bit of each octet in the 9 × 54 octet structure, the second bit represents even parity over the second bit of each octet in the 9 × 54 octet structure, etc. Therefore, the BIP-8 code provides for 8 separate even parity codes covering the corresponding bit of each octet in the 9 × 54 octet structure.

15.3.3.3 PLCP Path status (G1)

The G1 octet is allocated to convey the received PLCP status and performance back to the transmitting PLCP. This octet permits the status and performance of the full duplex PLCP Path to be monitored at either PLCP entity. Figure 15-3 depicts the G1 octet.

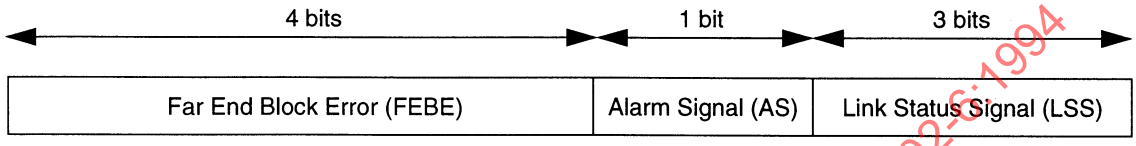


Figure 15-3—PLCP Path status (G1)

The first four bits of the G1 octet are the Far End Block Error (FEBE) code, which should be used to convey the count of interleaved-bit blocks that have been detected to be in error by the PLCP BIP-8 code in the preceding frame. If implemented, this count shall have nine legal codes, namely zero (0000) to eight (1000) errors. If not implemented, the code shall be (1111). The remaining six possible codes (1001 through 1110) would have been the result of an error condition and shall be interpreted as zero errors.

The fifth bit may be used for the Alarm Signal (AS). The AS alerts the transmitting PLCP that a received failure indication has been declared along the PLCP Path. When an incoming failure (i.e., PLCP Loss-Of-Frame) is detected on Bus x (x = A or B) that persists for 2.5 ± 0.5 s, an AS shall be generated on Bus y (y = B or A) by setting the fifth bit of the G1 octet to one (1). The AS shall be detected by a one (1) in the fifth bit of the G1 octet for ten consecutive frames. When the incoming failure has ceased for 15 ± 5 s, the AS shall be removed from Bus y by setting the fifth bit of the G1 octet to zero (0). Removal of the AS shall be detected by a zero (0) in the fifth bit of the G1 octet for ten consecutive frames. If the AS is not implemented, the default code for the AS shall be (0).

The remaining three bits shall be used for the Link Status Signal (LSS) as described in 11.3.2. The LSS is used to communicate information about the status of the transmission link between two adjacent PLCP entities. This information is conveyed only between these two entities.

The LSS codes for the G1 octet are shown in table 15-3. All other codes are invalid and shall be ignored by the receiver.

Table 15-3—LSS codes

LSS code	LSS name	Link status
000	connected	Received link connected
011	rx_link_dn	Received link down, no input or FORCE_DOWN
110	rx_link_up	Received link up

15.3.3.4 DQDB Layer Management Information octets (M2, M1)

The octets M1 and M2 carry the DQDB Layer Management Information octets, which are described in 10.1. The DQDB Layer Management Information octets shall be generated at the head of bus as described in 4.2, and shall be operated on by the DQDB Layer Management Protocol Entity, as described in 5.4.3.3, 10.2, and 10.3. There need be no correlation between TYPE = 0 or 1 octets and the M1 and M2 octets.

15.3.3.5 Stuff counter (octet C1)

The C1 octet shall provide a stuffing indicator for the PLCP frames. The C1 octet indicates the PLCP frame in which an octet-stuffing shall occur and contains the number of trailer octets transmitted (17–21).

The External Timing Source shall octet stuff given the opportunity to stuff in order to provide for a nominal 125 μs time period by adding/deleting one (1) trailer octet. When the HOB is using its own local clock generator to generate the timing information, no stuffing shall be used. The encoding of this octet is shown in table 15-4.

Table 15-4—E3 stuff counter codes

17	001	1101	1
18	010	0111	1
19	011	1010	1
20	100	1110	1
21	101	0011	1

The C1 codes have been chosen to provide error correction capability for 1 bit error and 2 adjacent bit errors and error detection capability for 3 random bit errors using the Abramson code: $(x^3 + x + 1)(x + 1)$.

The left-most 3 bits contain the numbering, the middle 4 bits provide the protection code, and the right-most bit is not used in the coding process. C1 evaluation shall always use the error correction mode.

15.3.3.6 Growth octets (Z3–Z1)

The octets Z1, Z2, and Z3 are reserved for future standardization. The PLCP shall encode these octets to the default code of (00000000).

15.3.4 Trailer octets

Each of the 17 to 21 trailer octets shall be encoded as (11001100).

15.4 PLCP behavior during faults

There are three types of conditions that directly influence the operation of the PLCP: transmission system faults, PLCP faults, and DQDB Layer out-of-service. The PLCP shall not differentiate between transmission system faults and PLCP faults. Thus all transmission system faults shall force the PLCP Out-Of-Frame.

When the PLCP declares a PLCP Out-Of-Frame condition (see 15.6), it shall start the Timer_P_x (x = A or B). This timer (Timer_P_x) shall be set to 1 ms ± 10 μs. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall generate a Jam signal on Bus x. The

Jam signal is defined as an unframed continuous bit pattern contained within the framed E3 information payload. The bit pattern shall be a repeating (1100) sequence (starting with a one–one (11) after each E3 synchronization double octet for the E3 application).

NOTE—The Jam signal is not needed in point-to-point configurations. It shall, however, be generated in any configuration.

If the PLCP detects the Jam signal for at least 20 μs, the PLCP shall reset and start the Timer_P_x. The PLCP shall continue to send the Jam signal on Bus x and shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

— If the PLCP enters the In-Frame state before the timer, Timer_P_x, expires, the timer shall be stopped. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as VALID at Ph-SAP_x. The PLCP shall resume its normal operation of processing/generating PLCP framing and PLCP Path Overhead octets and of performing its local stuffing.
— If the timer expires before the detection of valid PLCP framing octets, the PLCP Sublayer shall enter the Loss-Of-Frame state and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. The PLCP shall transmit on Bus y (y = B or A) an LSS equal to rx_link_dn.
  - If the DQDB Layer is capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to YES), then when the DQDB Layer receives the Ph-STATUS indication equal to DOWN at the Ph-SAP_x, the DQDB Layer shall start the Head of Bus Arbitration Timer, Timer_H_w (w = 1 or 2), as defined in 7.1.2, and shall send the HOBS value of WAITING, as defined in 10.2.3.4, on Bus x. Thus the PLCP shall generate a valid PLCP frame with the appropriate PLCP framing octets and PLCP Path Overhead octets and shall perform the stuffing mechanism.
  - If the DQDB Layer is not capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to NO; see 11.6.2), the PLCP Layer shall continue to generate the Jam signal on Bus x.

15.5 PLCP behavior during DQDB Layer out-of-service

The Physical Layer subsystem of a DQDB node connected to an E3-based transmission system shall always be powered up in normal operation. However, if for some reason the Physical Layer subsystem of a DQDB node is powered down, the stations downstream and upstream of this node can quickly detect this condition as a transmission system fault and the DQDB subnetwork would begin the fault detection process, as defined in 15.4, to reconfigure around the powered-down node.

The DQDB Layer shall have the ability of going in and out of service without interrupting the operation of the Physical Layer. When the DQDB Layer goes out of service, the PLCP, as well as the transmission system, shall continue normal operation (i.e., perform the stuffing mechanism and process/generate the PLCP Path Overhead octets). The DQDB Layer Management Information octets (M1 and M2) shall be relayed unmodified through the PLCP.

15.6 PLCP framing

The transition diagram for PLCP Framing State Machine is defined in figure 15-4. Each DQDB node has two PLCP Framing State Machines, one at the receiver for each bus.

PLCP Framing transition diagram

The state machine can be in one of four states: In-Frame (INF3), Out-Of-Frame (OOF1a), Out-Of-Frame_Jam (OOF_J1b), and Loss-Of-Frame (LOF2). The state machine shall be powered up in the Loss-Of-Frame (LOF2) state.

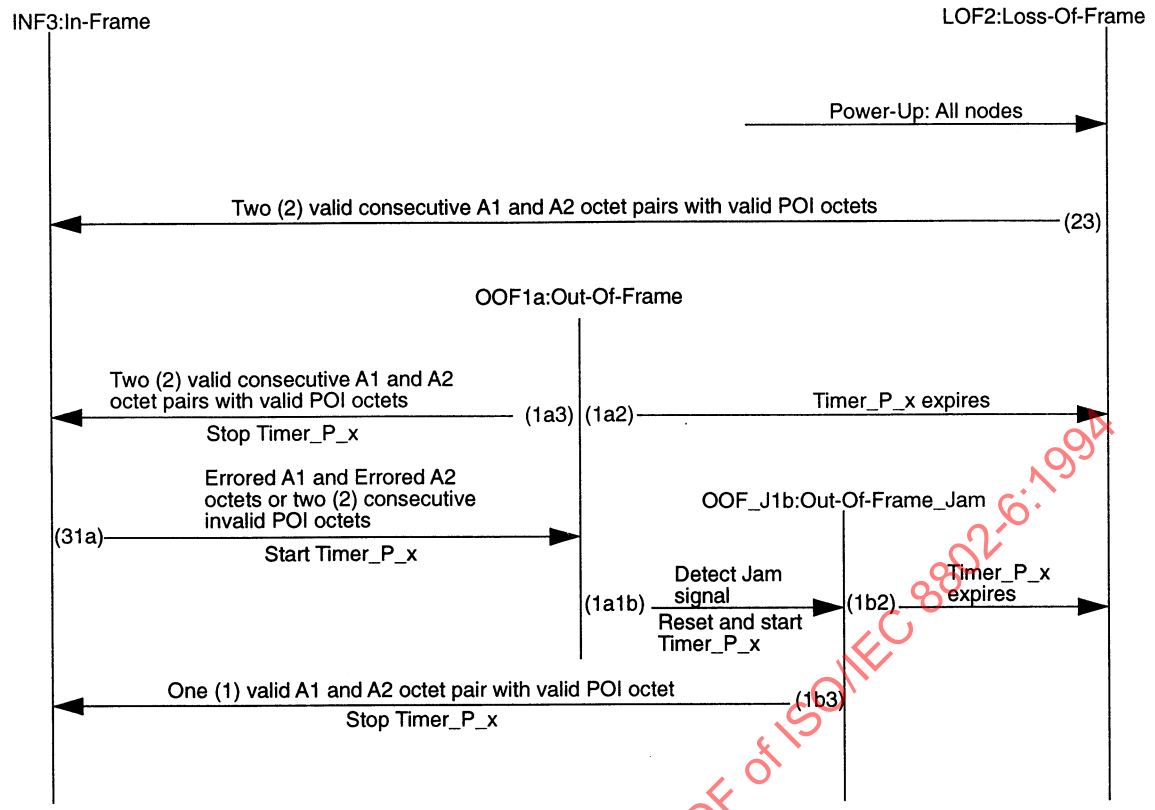


Figure 15-4—PLCP Framing State Machine transition diagram

State OOF1a: Out-Of-Frame

When entering this state, the PLCP Out-Of-Frame timer, Timer_P_x, shall be started, and the PLCP shall start generating the Jam signal on Bus x. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x (see 7.5.2), the PLCP shall continue to generate a framed PLCP signal on Bus x. The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

Transition 1a3

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential Path Overhead Identifier (POI) octets, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

Transition 1a2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1a1b

If the PLCP detects a Jam signal for at least 16 μs, then the state machine shall enter the OOF_J1b state. The PLCP shall reset and start the timer, Timer_P_x.

State OOF_J1b: Out-Of-Frame_Jam

The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall continue to generate the Jam signal. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x, then the PLCP shall continue to generate a framed PLCP signal.

Transition 1b2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1b3

If the PLCP detects one valid A1 and A2 octet pair with a valid POI octet, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

State LOF2: Loss-Of-Frame

When entering this state, the PLCP shall transmit on Bus y an LSS equal to rx_link_dn and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to YES, the PLCP shall generate a framed PLCP signal on Bus x starting with an A1 and A2 framing pattern sequence. If the HOB_CAPABLE Flag is set to NO, the PLCP shall generate the Jam signal on Bus x. The PLCP shall remain in this state until valid PLCP framing is found.

Transition 23

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential POI octets, then the state machine shall enter the INF3 state. The PLCP shall send to the DQDB Layer a Ph-STATUS indication equal to UP at Ph-SAP_x.

State INF3: In-Frame

The PLCP shall process/generate PLCP framing octets, POI octets, and PLCP Path Overhead octets as in normal operations and shall perform the stuffing mechanism. When the PLCP enters this state, it shall always begin transmission of the PLCP frame on Bus x with an A1 and A2 framing pattern sequence, and the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT (M2 and M1 octets) marked as INVALID at Ph-SAP_x until the PLCP detects the P8 octet. After the PLCP has detected the P8 octet, the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. The PLCP shall remain in this state until errored A1 and errored A2 octets or two consecutive invalid or nonsequential POI octets are detected.

Transition 31a

If the PLCP detects one or more errors in the A1 octet and one or more errors in the A2 octet of an A1 and A2 octet framing pair or two consecutive invalid or nonsequential POI octets, then the state machine shall enter the OOF1a state. The PLCP shall start the Timer_P_x.

15.6.1 LSS operations table

The operations table for the LSS is defined in table 15-5. The operations table determines the status of the transmission link according to the state of the PLCP Framing State Machine, the incoming LSS, and the Physical Layer Connection State Machine (PLCSM) control. This table supplements table 11-1. Additional states from table 11-1, corresponding to rows 4 to 6, are highlighted in boldface font.

Table 15-5—LSS operations table

INPUT				OUTPUT	
PLCP Frame State	**PLCSM Control**	**Incoming LSS**	**Detect Jam**	**Ph-STATUS**	**Outgoing LSS**
INF3	NORMAL	Connected	X	UP	connected
INF3	NORMAL	rx_link_up	X	UP	connected
INF3	NORMAL	rx_link_dn	X	DOWN	rx_link_up
OOF1a/OOF_J1b	**NORMAL**	**X**	**X**	**no change**	**rx_link_up**
LOF2	**NORMAL**	**X**	**NO**	**DOWN**	**rx_link_dn**
LOF2	**NORMAL**	**X**	**YES**	**DOWN**	**rx_link_up**
X	FORCE_DN	X	X	DOWN	rx_link_dn

KEY: X = Don't care

If a DQDB node with HOB_CAPABLE Flag set to NO receives an LSS code equal to rx_link_dn on Bus x, then the PLCP shall transmit on Bus x an LSS code equal to rx_link_dn, irrespective of the incoming LSS code on Bus y. This node adjacent to a failure would, therefore, be isolated from the DQDB subnetwork.

15.6.2 Physical Layer Frame TIming operations table

The Physical Layer Frame Timing operations table, table 15-6, determines whether an unframed Jam signal or a framed PLCP signal shall be transmitted on Bus x (x = A or B) and which 125 µs timing shall be used in the latter case to generate the framed PLCP signal. The transmission on Bus x is dependent on three inputs:

a) The state of the PLCP Framing State Machine for both incoming buses (see figure 15-4).
b) The Timing Source and HOB_OPERATION_z (z = 1, 2, or Default) outputs from the CC_z operations tables in the DQDB Layer, defined in tables 10-10b), 10-11, and 10-12, respectively.
c) The value of the HOB_CAPABLE Flag, defined in 11.6.2.

This table supplements the Timing Source information of tables 10-10(b), 10-11, and 10-12.

Table 15-6—Physical Layer Frame Timing operations supplementary table

INPUT			OUTPUT	
Bus x PLCP Frame State	**HOB OPERATION_z**	**HOB_CAPABLE Flag**	**Bus x Tx Frame**	**125 μs Timing**
OOF1a/ OOF_J1b	not HEAD_ OF_BUS_x	X	Jam	N/A
OOF1a/ OOF_J1b	HEAD_OF_ BUS_x	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)
LOF2	not HEAD_ OF_BUS_x	NO	Jam	N/A
LOF2	X	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)

KEY: X = Don't care
N/A = Not applicable

NOTES

1—Selection between NODE_CLOCK and EXTERNAL_CLOCK is determined by tables 10-10b), 10-11, and 10-12 when the Bus x PLCP Frame State is OOF1a, OOF_J1b, or LOF2.

2—The HOB_OPERATION_z column refers to the value of HOB_OPERATION for the CC_z associated with Ph-SAP_x at the node.

16. PLCP for CCITT Recommendations G.751 and G.703 (139.264 Mbit/s)

16.1 Overview

This clause provides a convergence procedure in which the DQDB Layer is mapped into a standard transmission system according to CCITT Recommendations G.751 and G.703 operating at 139.264 Mbit/s as used in public networks. Beyond the provisions of CCITT Recommendations G.751 and G.703, a 125 μs framing period is provided to support $n \times 64$ kbit/s channels based on octets.

The E4 PLCP[67] makes use of the optional status parameter in Ph-DATA indication primitives. (See 4.2.) Hence, the status parameter is mandatory for the service provided by the E4 PLCP.

16.1.1 E4 relationship to the PLCP

The rate, format, electrical characteristics, and other attributes of the E4 signal shall be as defined in CCITT Recommendations G.751 and G.703. The first 2 octets of each CCITT Recommendation G.751 frame shall not be used for the PLCP. They are left for E4 synchronization (i.e., frame alignment) and overhead bits compatible with existing equipment.

NOTES

1—The coding of the two octets shall be (111110100000 A NNN), where A means the Alarm Indication to the remote end bit and NNN are three bits reserved for National Use. The "sliding" of the 2 octets through the 125 μs frame shall be done via full octets.

Thus the net bit rate available to the PLCP is 138.503 Mbit/s. The nominal frame rate is 47.563 kHz.

2—The PLCP shall provide sufficient buffering or other provisions to accommodate the "jump" resulting from the skipping of the 2 unavailable octets.

The 125 μs frame is not in any way aligned with the 21.0247 μs (47.563 kHz) frame of the transmission system. The E4 payload slots are only octet aligned to the E4 overhead octets.

16.2 The PLCP frame format

A frame duration of 125 μs is chosen to easily accommodate $n \times 64$ kbit/s channels based on octets. Four octets shall be added to each DQDB slot of 53 octets to provide framing and overhead functions so that each row contains 57 octets. (The first three octets are used for framing. The fourth octet is used to carry PLCP Path Overhead octets.) Thirty-seven of these 57 rows are placed into one 125 μs PLCP frame. Fifty-five to 57 octets (nominal) shall be left as a trailer depending on whether 5 or 6 E4 overhead double octets occurred during the 125 μs frame.

The complete frame structure is shown in figure 16-1. Each row of bits in the PLCP frame format illustrated in figure 16-1 shall be transmitted in order, from left to right, top to bottom.

16.3 PLCP field definitions

(Refer to figure 16-1.) See 6.1 for ordering principles of fields. The values of fields are described as bit patterns. The left-most bit of each octet is the most significant.

[67]The designation E4 is used for a 139.264 Mbit/s transmission system according to CCITT Recommendation G.751. It is the fourth level of the plesiochronous digital hierarchy.

←1→	←1→	←1→	←1→	←53 octets→	
A1	A2	P36	Z31	First DQDB slot	
A1	A2	P35	Z30	DQDB slot	
•	•	•	•		
•	•	•	•		
A1	A2	P6	Z1	DQDB slot	
A1	A2	P5	F1	DQDB slot	
A1	A2	P4	B1	DQDB slot	
A1	A2	P3	G1	DQDB slot	
A1	A2	P2	M2	DQDB slot	
A1	A2	P1	M1	DQDB slot	55–57 octets
A1	A2	P0	C1	Last DQDB slot	Trailer

125 µs

KEY: A1 = Framing octet (11110110)
A2 = Framing octet (00101000)
P8 – P0 = Path Overhead Identifier octets

PLCP Path Overhead octets:

Z3 – Z1 = Growth octets
F1 = PLCP Path user channel
B1 = BIP-8
G1 = PLCP Path status
M2 – M1 = DQDB Layer Management Information octets
C1 = Stuff counter

Figure 16-1—The E4 PLCP frame format

16.3.1 Framing octets (A1, A2)

The first two columns (A1, A2) may be used to provide slot delineation. The encoding of the A1 and A2 octets is shown in table 16-1.

These codes are the same patterns as used in the Synchronous Digital Hierarchy (SDH) CCITT Recommendations G.707, G.708, and G.709. See 16.6 for PLCP framing requirements.

Table 16-1—A1 and A2 codes

A1	A2
11110110	00101000

Alternatively, slot delineation based on the Header Check Sequence (HCS) of the DQDB header may be used.

16.3.2 Path Overhead Identifier (P36–P0)

The third column (P36–P0) identifies the PLCP Overhead octets contained in the fourth column of figure 16-1. Figure 16-2 shows the format of the Path Overhead Identifier (POI) octet. The left-most 6 bits of these octets provide numbering of the 37 rows. The reserved bit shall be set to (0). The parity bit provides odd parity over this field.

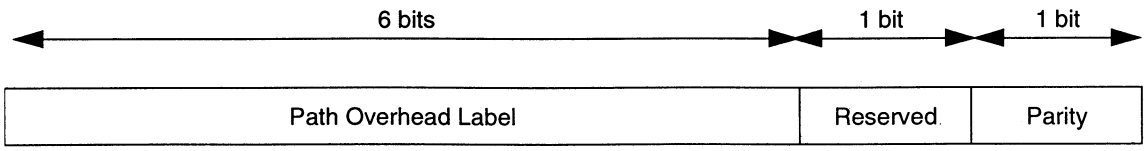


Figure 16-2—POI octet

Table 16-2 defines the codes for P36–P0, which shall be generated by the PLCP at each node. All other codes are invalid. A code shall also be considered invalid if the parity bit contained in the LSB is not correct. The response to invalid codes is described in 16.6.

Table 16-2—POI codes

P36	100100	0	1
P35	100011	0	0
	•		
	•		
P7	000111	0	0
P6	000110	0	1
P5	000101	0	1
P4	000100	0	0
P3	000011	0	1
P2	000010	0	0
P1	000001	0	0
P0	000000	0	1

16.3.3 PLCP Path Overhead octets

The PLCP Path is defined between two adjacent peer PLCP entities. The F1, B1, G1 and C1 PLCP Path Overhead octets are related to PLCP operation and shall be terminated/generated at each PLCP on the subnetwork. The M1 and M2 octets are provided for the transport of DQDB Layer Management Information octets and shall not be processed by the PLCP Sublayer.

16.3.3.1 PLCP Path user channel (F1)

The F1 octet is the user channel, which is allocated for user communication purposes between adjacent PLCPs. The use of this octet in DQDB subnetworks is for further study.[68] The default code for this octet shall be (00000000).

16.3.3.2 Bit Interleaved Parity-8 (B1)

One octet is allocated for PLCP Path error monitoring. This function shall be a Bit Interleaved Parity-8 (BIP-8) code using even parity. The PLCP Path BIP-8 is calculated over the 37 × 54 octet structure (columns

[68]One possible application is for assisting maintenance personnel.

4 to 57, 1 PLCP Path Overhead octet and 53 DQDB slot octets per row) of the previous PLCP frame and inserted into the B1 octet of the current PLCP frame.

A BIP-8 is an 8-bit code in which the first bit of the BIP-8 code represents even parity calculated over the first bit of each octet in the 37 × 54 octet structure, the second bit represents even parity over the second bit of each octet in the 37 × 54 octet structure, etc. Therefore, the BIP-8 code provides for 8 separate even parity codes covering the corresponding bit of each octet in the 37 × 54 octet structure.

16.3.3.3 PLCP Path status (G1)

The G1 octet is allocated to convey the received PLCP status and performance back to the transmitting PLCP. This octet permits the status and performance of the full duplex PLCP Path to be monitored at either PLCP entity. Figure 16-3 depicts the G1 octet.

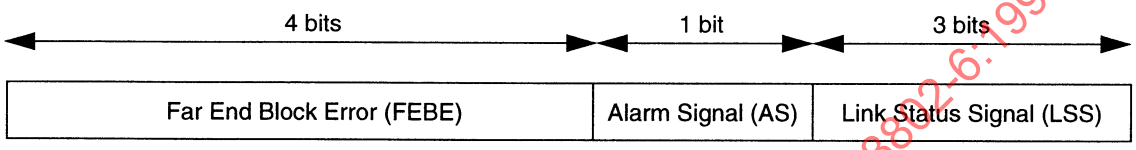


Figure 16-3—PLCP Path status (G1)

The first four bits of the G1 octet are the Far End Block Error (FEBE) code, which should be used to convey the count of interleaved-bit blocks that have been detected to be in error by the PLCP BIP-8 code in the preceding frame. If implemented, this count shall have nine legal codes, namely zero (0000) to eight (1000) errors. If not implemented, the code shall be (1111). The remaining six possible codes (1001 through 1110) would have been the result of an error condition and shall be interpreted as zero errors.

The fifth bit may be used for the Alarm Signal (AS). The AS alerts the transmitting PLCP that a received failure indication has been declared along the PLCP Path. When an incoming failure (i.e., PLCP Loss-Of-Frame) is detected on Bus x (x = A or B) that persists for 2.5 ± 0.5 s, an AS shall be generated on Bus y (y = B or A) by setting the fifth bit of the G1 octet to one (1). The AS shall be detected by a one (1) in the fifth bit of the G1 octet for ten consecutive frames. When the incoming failure has ceased for 15 ± 5 s, the AS shall be removed from Bus y by setting the fifth bit of the G1 octet to zero (0). Removal of the AS shall be detected by a zero (0) in the fifth bit of the G1 octet for ten consecutive frames. If the AS is not implemented, the default code for the AS shall be (0).

The remaining three bits are used for the Link Status Signal (LSS) code as described in 11.3.2. The LSS shall be used to communicate information about the status of the transmission link between two adjacent PLCP entities. This information shall be conveyed only between these two entities.

The LSS codes for the G1 octet are shown in table 16-3. All other codes shall be invalid and shall be ignored by the receiver.

Table 16-3—LSS codes

LSS code	LSS name	Link status
000	connected	Received link connected
011	rx_link_dn	Received link down, no input or FORCE_DOWN
110	rx_link_up	Received link up

16.3.3.4 DQDB Layer Management Information octets (M1, M2)

The octets M1 and M2 carry the DQDB Layer Management Information octets, which are described in 10.1. The DQDB Layer Management Information octets shall be generated at the head of bus as described in 4.2, and shall be operated on by the DQDB Layer Management protocol entity as described in 5.4.3.3, 10.2, and 10.3. There need be no correlation between TYPE = 0 or 1 octets and the M1 and M2 octets.

16.3.3.5 Stuff counter (C1)

The C1 octet shall provide a stuffing indicator for the PLCP frames. The C1 octet shall indicate the PLCP frame in which an octet-stuffing shall occur and contains the number of trailer octets transmitted (54 to 58).

The External Timing Source shall octet stuff given the opportunity to stuff in order to provide for a nominal 125 µs time period by adding/deleting one (1) trailer octet. When the HOB is using its own local clock generator to generate the timing information, no stuffing shall be used. The encoding of this octet is shown in table 16-4.

Table 16-4—E3 stuff counter codes

54	001	1101	1
55	010	0111	1
56	011	1010	1
57	100	1110	1
58	101	0011	1

The C1 codes have been chosen to provide error correction capability for 1 bit error and 2 adjacent bit errors and error detection capability for 3 random bit errors using the Abramson code: $(x^3 + x + 1)(x + 1)$.

The left-most 3 bits contain the numbering, the middle 4 bits provide the protection code, and the right-most bit is not used in the coding process. C1 evaluation shall always use the error correction mode.

16.3.3.6 Growth octets (Z31–Z1)

The octets Z31–Z1 are reserved for future standardization. The PLCP shall encode these octets to the default code of (00000000).

16.3.4 Trailer octets

Each of the 54 to 58 trailer octets shall be encoded as (11001100).

16.4 PLCP behavior during faults

There are three types of conditions that directly influence the operation of the PLCP: transmission system faults, PLCP faults, and DQDB Layer out-of-service. The PLCP shall not differentiate between transmission system faults and PLCP faults. Therefore, all transmission system faults shall force the PLCP Out-Of-Frame.

When the PLCP declares a PLCP Out-Of-Frame condition (see 16.6), it shall start the Timer_P_x (x = A or B). This timer (Timer_P_x) shall be set to 1 ms ± 10 µs. The PLCP shall send to the DQDB Layer Ph-DATA

indication octets marked as INVALID at Ph-SAP_x. The PLCP shall generate a Jam signal on Bus x. The Jam signal is defined as an unframed continuous bit pattern contained within the framed E4 information payload. The bit pattern shall be a repeating (1100) sequence (starting with a one–one (11) after each E4 synchronization double octet for the E4 application).

NOTE—The Jam signal is not needed in point-to-point configurations. It shall, however, be generated in any configuration.

If the PLCP detects the Jam signal for at least 8 μs, the PLCP shall reset and start the Timer_P_x. The PLCP shall continue to send the Jam signal on Bus x and shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

— If the PLCP enters the In-Frame state before the timer, Timer_P_x, expires, the timer shall be stopped. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as VALID at Ph-SAP_x. The PLCP shall resume its normal operation of processing/generating PLCP framing and PLCP Path Overhead octets and of performing its local stuffing.
— If the timer expires before the detection of valid PLCP framing octets, the PLCP Sublayer shall enter the Loss-Of-Frame state and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. The PLCP shall transmit on Bus y (y = B or A) an LSS equal to rx_link_dn.
  • If the DQDB Layer is capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to YES), then when the DQDB Layer receives the Ph-STATUS indication equal to DOWN at the Ph-SAP_x, the DQDB Layer shall start the Head of Bus Arbitration Timer, Timer_H_w (w = 1 or 2), as defined in 7.1.2, and shall send the HOBS value of WAITING, as defined in 10.2.3.4, on Bus x. Thus the PLCP shall generate a valid PLCP frame with the appropriate PLCP framing octets and PLCP Path Overhead octets and shall perform the stuffing mechanism.
  • If the DQDB Layer is not capable of becoming HOB (i.e., the HOB_CAPABLE Flag is set to NO; see 11.6.2), the PLCP Layer shall continue to generate the Jam signal on Bus x.

16.5 PLCP behavior during DQDB Layer out-of-service

The Physical Layer subsystem of a DQDB node connected to an E4-based transmission system shall always be powered up in normal operation. However, if for some reason the Physical Layer subsystem of a DQDB node is powered down, the stations downstream and upstream of this node would immediately detect this condition as a transmission system fault and the DQDB subnetwork would begin the fault detection process, as defined in 16.4, to reconfigure around the powered-down node.

The DQDB Layer shall have the ability of going in and out of service without interrupting the operation of the Physical Layer. When the DQDB Layer goes out-of-service, the PLCP, as well as the transmission system, shall continue normal operation (i.e., perform the stuffing mechanism and process/generate the PLCP Path Overhead octets). The DQDB Layer Management Information octets (M1 and M2) shall be relayed unmodified through the PLCP.

16.6 PLCP framing

The transition diagram for the PLCP Framing State Machine is defined in figure 16-4. Each DQDB node has two PLCP Framing State Machines, one at the receiver for each bus.

PLCP Framing transition diagram

The state machine can be in one of four states: In-Frame (INF3), Out-Of-Frame (OOF1a), Out-Of-Frame_Jam (OOF_J1b), and Loss-Of-Frame (LOF2). The state machine is powered up in the Loss-Of-Frame (LOF2) state.

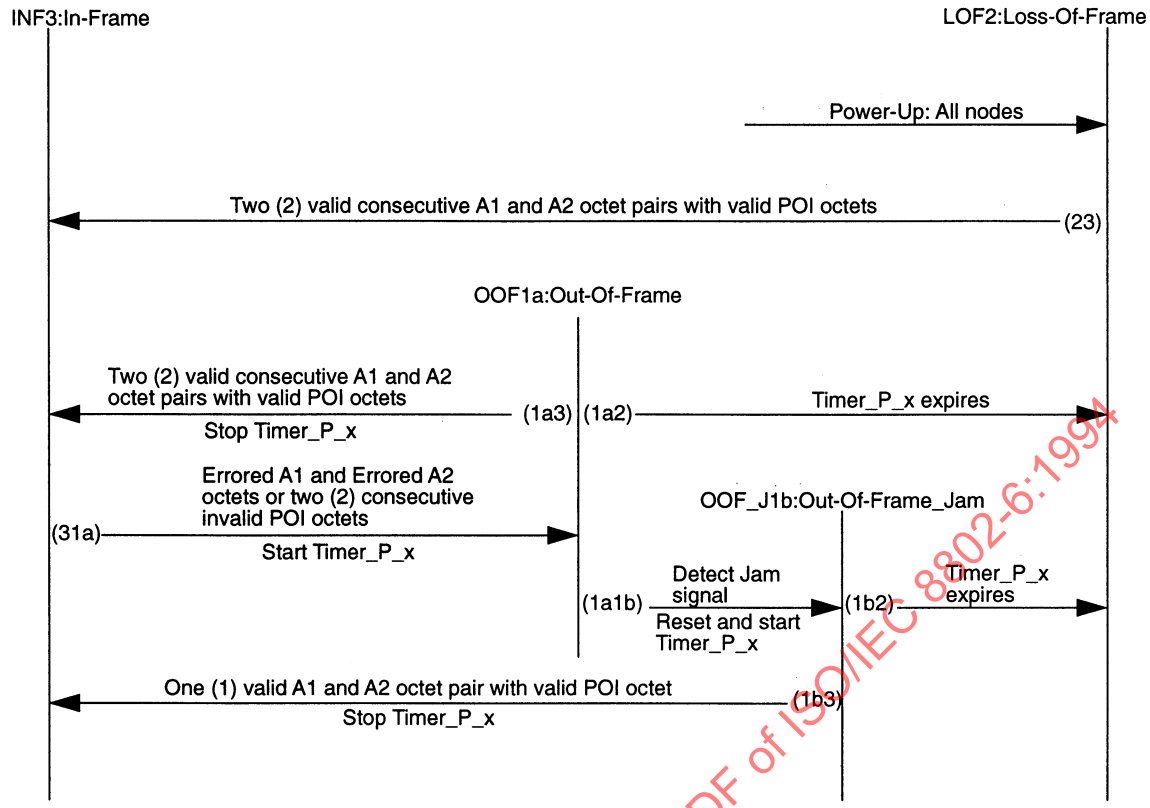


Figure 16-4—PLCP Framing State Machine transition diagram

State OOF1a: Out-Of-Frame

When entering this state, the PLCP Out-Of-Frame timer, Timer_P_x, shall be started, and the PLCP shall start generating the Jam signal on Bus x. However, if the node contains an HOB_OPERATION value of HEAD_OF_BUS_x (see 7.5.2), the PLCP shall continue to generate a framed PLCP signal on Bus x. The PLCP shall remain in this state until the Timer_P_x expires, the Jam signal is detected on Bus x, or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x.

Transition 1a3

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential Path Overhead Identifier (POI) octets, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

Transition 1a2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1a1b

If the PLCP detects a Jam signal for at least 8 μs, then the state machine shall enter the OOF_J1b state. The PLCP shall reset and start the timer, Timer_P_x.

State OOF_J1b: Out-Of-Frame_Jam

The PLCP shall remain in this state until the Timer_P_x expires or until valid framing is found. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. The PLCP shall continue to generate the Jam signal. However, if the node contains a HOB_OPERATION value of HEAD_OF_BUS_x, then the PLCP shall continue to generate a framed PLCP signal.

Transition 1b2

If the Timer_P_x expires, then the state machine shall enter the LOF2 state.

Transition 1b3

If the PLCP detects one valid A1 and A2 octet pair with a valid POI octet, then the state machine shall enter the INF3 state. The PLCP shall stop and reset the timer, Timer_P_x.

State LOF2: Loss-Of-Frame

When entering this state, the PLCP shall transmit on Bus y an LSS equal to rx_link_dn and shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to YES, the PLCP shall generate a framed PLCP signal on Bus x starting with an A1 and A2 framing pattern sequence. If the HOB_CAPABLE Flag is set to NO, the PLCP shall generate the Jam signal on Bus x. The PLCP shall remain in this state until valid PLCP framing is found.

Transition 23

If the PLCP detects two consecutive valid A1 and A2 octet pairs with two consecutive valid and sequential POI octets, then the state machine shall enter the INF3 state. The PLCP shall send to the DQDB Layer a Ph-STATUS indication equal to UP at Ph-SAP_x.

State INF3: In-Frame

The PLCP shall process/generate PLCP framing octets, POI octets, and PLCP Path Overhead octets as in normal operations and shall perform the stuffing mechanism. When the PLCP enters this state, it shall always begin transmission of the PLCP frame on Bus x with an A1 and A2 framing pattern sequence, and the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT (M2 and M1 octets) marked as INVALID at Ph-SAP_x until the PLCP detects the P36 octet. After the PLCP has detected the P36 octet, the PLCP shall send to the DQDB Layer Ph-DATA indications of type DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. The PLCP shall remain in this state until errored A1 and errored A2 octets or two consecutive invalid or nonsequential POI octets are detected.

Transition 31a

If the PLCP detects one or more errors in the A1 octet, and one or more errors in the A2 octet of an A1 and A2 octet framing pair or two consecutive invalid or nonsequential POI octets, then the state machine shall enter the OOF1a state. The PLCP shall start the Timer_P_x.

16.6.1 LSS operations table

The operations table for the LSS is defined in table 16-5. The operations table determines the status of the transmission link according to the state of the PLCP Framing State Machine, the incoming LSS, and the Physical Layer Connection State Machine (PLCSM) control. This table supplements table 11-1. Additional states from table 11-1, corresponding to rows 4 to 6, are highlighted in boldface font.

Table 16-5—LSS operations table

INPUT				OUTPUT	
PLCP Frame State	**PLCSM Control**	**Incoming LSS**	**Detect Jam**	**Ph-STATUS**	**Outgoing LSS**
INF3	NORMAL	Connected	X	UP	connected
INF3	NORMAL	rx_link_up	X	UP	connected
INF3	NORMAL	rx_link_dn	X	DOWN	rx_link_up
OOF1a/OOF_J1b	**NORMAL**	**X**	**X**	**no change**	**rx_link_up**
LOF2	**NORMAL**	**X**	**NO**	**DOWN**	**rx_link_dn**
LOF2	**NORMAL**	**X**	**YES**	**DOWN**	**rx_link_up**
X	FORCE_DN	X	X	DOWN	rx_link_dn

KEY: X = Don't care

If a DQDB node with HOB_CAPABLE Flag set to NO receives an LSS code equal to rx_link_dn on Bus x, then the PLCP shall transmit on Bus x an LSS code equal to rx_link_dn, irrespective of the incoming LSS code on Bus y. This node adjacent to a failure would, therefore, be isolated from the DQDB subnetwork.

16.6.2 Physical Layer Frame Timing operations table

The Physical Layer Frame Timing operations table, table 16-6, determines whether an unframed Jam signal or a framed PLCP signal will be transmitted on Bus x (x = A or B) and which 125 µs timing will be used to generate the framed PLCP signal. The transmission on Bus x is dependent on three inputs:

Table 16-6—Physical Layer Frame Timing operations supplementary table

INPUT			OUTPUT	
Bus x PLCP Frame State	**HOB OPERATION_z**	**HOB_CAPABLE Flag**	**Bus x Tx Frame**	**125 µs Timing**
OOF1a/ OOF_J1b	not HEAD_ OF_BUS_x	X	Jam	N/A
OOF1a/ OOF_J1b	HEAD_OF_ BUS_x	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)
LOF2	not HEAD_ OF_BUS_x	NO	Jam	N/A
LOF2	X	YES	PLCP signal	NODE_CLOCK or EXTERNAL_ CLOCK (note 1)

KEY: X = Don't care
N/A = Not applicable

NOTES

1—Selection between NODE_CLOCK and EXTERNAL_CLOCK is determined by tables 10-10b), 10-11, and 10-12 when the Bus x PLCP Frame State is OOF1a, OOF_J1b, or LOF2.
2—The HOB_OPERATION_z column refers to the value of HOB_OPERATION for the CC_z associated with Ph-SAP_x at the node.

a) The state of the PLCP Framing State Machine for both incoming buses (see figure 16-4).
b) The Timing Source and HOB_OPERATION_z (z = 1, 2, or Default) outputs from the CC_z operations tables in the DQDB Layer, defined in tables 10-10b), 10-11, and 10-12, respectively.
c) The value of the HOB_CAPABLE Flag, defined in 11.6.2.

This table supplements the Timing Source information of tables 10-10b), 10-11, and 10-12.

17. PLCP for CCITT Recommendations G.707, G.708, and G.709 SDH-based systems (155.520 Mbit/s)

17.1 Overview

This clause defines a convergence procedure for transfer of DQDB slots using the Synchronous Digital Hierarchy (SDH) at a 155.520 Mbit/s physical medium rate. The rates, formats, and other attributes of SDH are defined in CCITT Recommendations G.707, G.708, and G.709. DQDB slots are mapped into VC-4 virtual containers, and the VC-4s are transported using Synchronous Transport Modules.[69]

17.1.1 SDH relationship to the PLCP

A mapping of Asynchronous Transfer Mode (ATM) cells into VC-4 can be found in CCITT Recommendation I.432. As ATM cells and DQDB slots are identical in length (53 octets) and nearly identical in format, the mapping of DQDB slots into VC-4 is identical to the ATM cell mapping into VC-4 except for the following matters:

— The use of the User Channel (F2) and Growth (Z3) octets for carrying DQDB Layer Management Information octets (M1 and M2).
— The use of Reserve bit positions in the Multiframe Indicator (H4) octet for providing the Link Status Signal (LSS).
— The use of VC-4 for propagating the DQDB Layer 125 µs timing along the DQDB buses.
— The optional use of either 6 bit positions in the Multiframe Indicator (H4), or the Header Check Sequence (HCS) method for providing slot boundary indication. The HCS method for slot delineation is identical to the Header Error Control (HEC) method for ATM cell delineation described in CCITT Recommendation I.432, 4.5.1.1, except for the fact that the HCS is calculated over 3 octets of the DQDB slot header, whereas the ATM HEC is calculated over 4 octets of the ATM cell header.

CCITT Recommendations G.707, G.708, and G.709 shall be the normative references for providing an SDH-based Physical Layer for DQDB, with the above modifications. Descriptions of SDH Path Overhead (POH) field definitions in this clause other than the M1–M2 and H4 fields are included for clarity and completeness only.

The SDH PLCP makes use of the optional status parameter in Ph-DATA indication primitives (see 4.2). Hence, the status parameter is mandatory for the service provided by the SDH PLCP.

17.2 The PLCP frame format

The PLCP frame format is a virtual container VC-4 that consists of 9 rows by 261 octets. The VC-4 has a nominal duration of 125 µs. The VC-4 frame rate shall provide the 125 µs timing information. The VC-4 frames are transported between peer PLCPs by the SDH transmission system.

DQDB slots are mapped into the VC-4 as illustrated in figure 17-1. The VC-4 consists of one column (nine octets) of POH plus a 9 row by 260 column payload capacity.

The DQDB slots are located horizontally (by row) in the VC-4 payload capacity with the slot boundaries aligned with the VC-4 octet boundaries. Because the VC-4 payload capacity (2340 octets) is not an integer multiple of the DQDB slot length (53 octets), a slot is allowed to cross the VC-4 boundary. Slot boundary

[69]The PLCP mapping described in this clause applies to both the SDH STM-1 rate signals and the ANSI SONET STS-3c rate signals. The primary reference, however, is the SDH STM-1 mapping. The PLCP mapping for the ANSI SONET STS-3c rate signal uses the STS-3c Synchronous Payload Envelope (SPE) as the equivalent of the SDH VC-4.

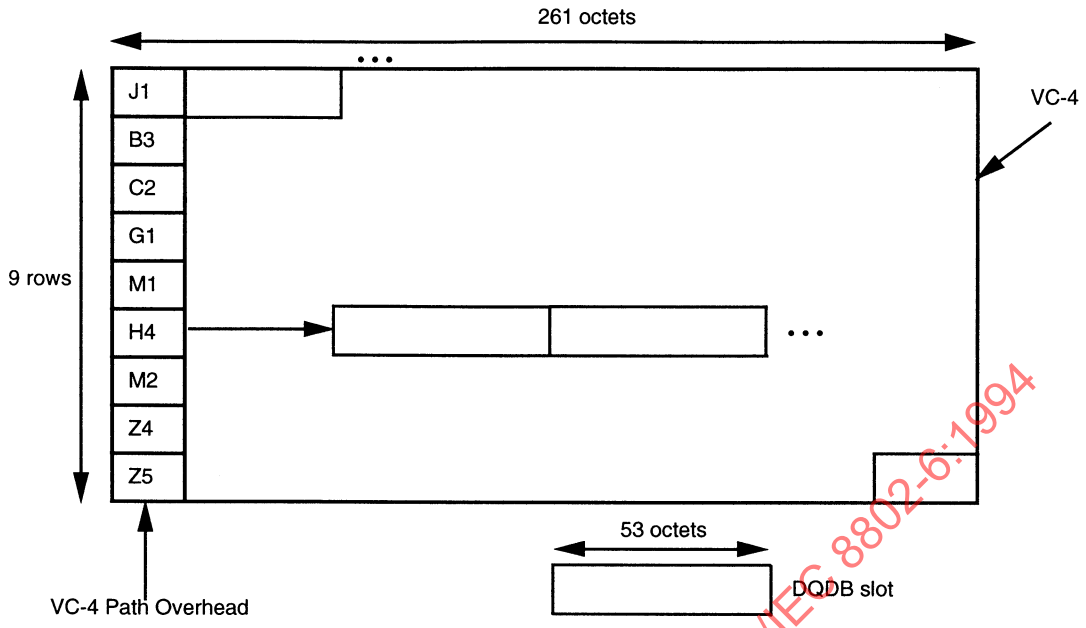


Figure 17-1—SDH VC-4 PLCP mapping

indication shall be provided using the Header Check Sequence (HCS) method or on a 125 μs basis by use of the POH H4 octet.

The slot format is illustrated in figure 17-2. The segment payload of 48 octets shall be scrambled before VC-4 framing. The scrambler operates for the duration of the 48-octet segment payload. Operation is suspended and the scrambler state is retained at all other times. A self-synchronous scrambler with generator polynomial $x^{43} + 1$ shall be used. In the reverse operation, following termination of the VC-4 signal and slot delineation, the segment payload shall be descrambled. The descrambler operates for the duration of the assumed segment payload according to the derived segment delineation (see 17.6.1.1). Operation is suspended and the descrambler state is retained at all other times.

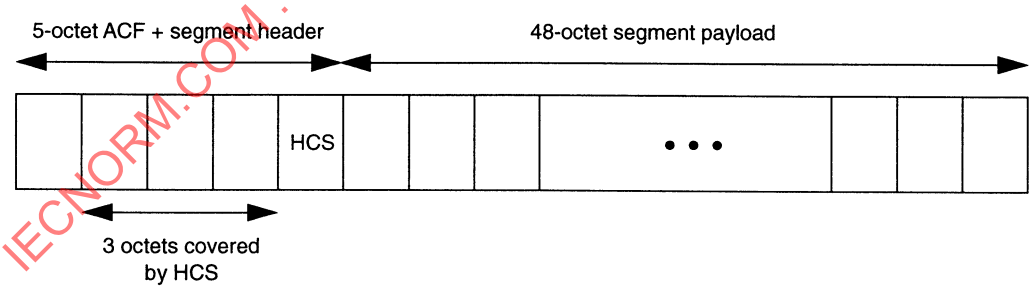


Figure 17-2—DQDB slot format

At the transmitting PLCP, an 8-bit pattern shall be added (modulo 2) to the HCS field of the segment headers. At the receiving PLCP, the same bit pattern shall be subtracted (equal to add modulo 2) from the HCS field of the assumed segment headers. The bit pattern shall be (01010101).

17.3 PLCP Path Overhead (POH) field definitions

The first column of the VC-4 contains the POH octets. The following subclauses describe each of the VC-4 POH octets and their functions. As previously noted, these descriptions are consistent with CCITT Recommendation G.709 except for the use of the User Channel (F2), Growth (Z3), and Multiframe Indicator (H4) octets. Values of octets are described as bit patterns. The left-most bit of each octet is the most significant.

The PLCP Path is defined between two adjacent peer PLCP entities. All POH octets other than M1 and M2 are related to PLCP operation and are terminated/generated at each PLCP on the subnetwork. The M1 and M2 octets are provided for the transport of DQDB Layer Management Information octets and shall not be processed by the PLCP.

17.3.1 Path trace (J1)

The J1 octet is used to repetitively transmit a 64-octet, fixed length string so that a receiving PLCP can verify its continued connection to the intended transmitter PLCP.

17.3.2 Bit Interleaved Parity-8 (B3)

The B3 octet is allocated for PLCP Path error monitoring. This function shall be a Bit Interleaved Parity-8 (BIP-8) code using even parity. The PLCP Path BIP-8 is calculated over all bits of the previous VC-4. The computed BIP-8 is placed in the B3 octet of the current VC-4. The BIP-8 is calculated after the PLCP scrambling of the segment payload.

A BIP-8 is an 8-bit code in which the first bit of the BIP-8 code calculates even parity over the first bit of each octet in the VC-4, the second bit of the BIP-8 code calculates even parity over the second bit of each octet in the VC-4, etc. Therefore, the BIP-8 code provides for 8 separate even parity codes covering the corresponding bit of each octet in the VC-4.

17.3.3 Signal label (C2)

The C2 octet is allocated to indicate the construction of the VC-4 payload. The value of this octet shall be set to code $(14)_{HEX}$, which indicates an IEEE 802.6 payload and overhead structure.

17.3.4 Path status (G1)

The G1 octet is allocated to convey the received PLCP status and performance to the transmitting PLCP. This octet permits the status and performance of the full duplex PLCP Path to be monitored at either PLCP entity. Figure 17-3 depicts the G1 octet.

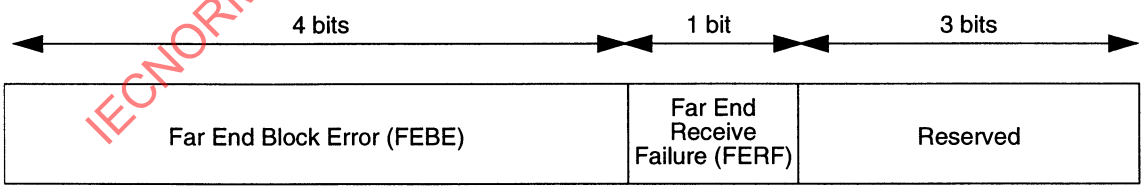


Figure 17-3—PLCP Path status (G1) coding

The four most significant bits of the G1 octet are the Far End Block Error (FEBE) code, which shall be used to convey the count of interleaved-bit blocks that have been detected to be in error by the PLCP BIP-8 code in the preceding VC-4. This count has nine legal values, namely zero (0000) to eight (1000) errors. The remaining seven possible codes (1001 through 1111) shall be interpreted as zero errors.

The fifth bit is the Far End Receive Failure (FERF) signal. The FERF alerts the transmitting PLCP that a received failure indication has been declared along the PLCP Path. When an incoming failure (i.e., the Framing State Machine in Loss-Of-Frame or Loss-Of-Slot-Delineation states (see 17.6.1.2) is detected on Bus x (x = A or B) that persists for 2.5 ± 0.5 s, a FERF shall be generated on Bus y (y = B or A) by setting the fifth bit of the G1 octet to one (1). FERF shall be detected by a one (1) in the fifth bit of the G1 octet for ten consecutive VC-4s. When the incoming failure has ceased for 15 ± 5 s, FERF shall be removed from Bus y by setting the fifth bit of the G1 octet to zero (0). Removal of FERF is detected by a zero (0) in the fifth bit of the G1 octet for ten consecutive VC-4s.

The remaining three least significant bits are reserved for future standardization. The transmitting PLCP shall encode these bits to the default code of (000). The receiver PLCP shall be capable of ignoring the values contained in these bits.

17.3.5 Multiframe Indicator (H4)

The H4 octet is the Multiframe Indicator for payloads. The coding of the H4 octet is illustrated in figure 17-4.

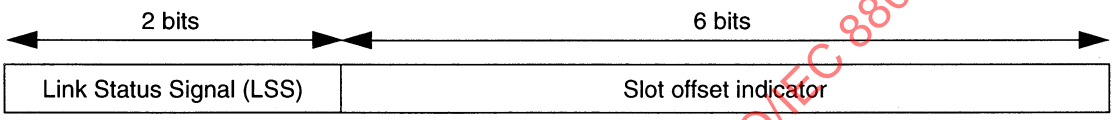


Figure 17-4—Multiframe Indicator (H4) coding

The two most significant bits are used for the Link Status Signal (LSS) as described in 11.3.2. The LSS is used to communicate information about the status of the transmission link between two adjacent PLCP entities.

The LSS codes for the H4 octet are shown in table 17-1.

Table 17-1—LSS codes

LSS code	LSS name	Link status
00	connected	Received link connected
11	rx_link_dn	Received link down, no input or FORCE_DOWN
01	rx_link_up	Received link up
10	hob_incapable	Lack of upstream head of bus capability

The six least significant bits of the H4 octet form the slot offset indicator. The slot offset indicator shall contain a binary number indicating the offset in octets between the H4 octet and the first slot boundary following the H4 octet. The valid range of the slot offset indicator value shall be 0 to 52.

17.3.6 DQDB Layer Management Information octets (M2, M1)

The octets M1 and M2 carry the DQDB Layer Management Information octets, which are described in 10.1. The DQDB Layer Management Information octets are generated at the head of a bus as described in 4.2, and are operated on by the DQDB Layer Management Protocol Entity as described in 5.4.3.3, 10.2, and 10.3. There need be no correlation between TYPE = 0 or 1 octets and the M1 or M2 octets.

17.3.7 Growth octets

The Z4 and Z5 octets are reserved for future standardization. The transmitter PLCP shall encode these octets to the default code of (00000000). The receiver PLCP shall be capable of ignoring the values contained in these octets.

17.4 PLCP behavior during faults

There are two types of conditions that directly influence the operation of the PLCP: DQDB Layer out-of-service and faults introduced by the PLCP or the SDH transmission system. Faults shall force the PLCP Out-Of-Frame or Out-Of-Slot-Delineation (see 17.6.1.2).

The Out-Of-Frame state is entered when a Loss-Of-Pointer or Alarm Indication Signal is declared by the SDH transmission system Pointer Interpretation State Machine (see CCITT Recommendation G.783, annex B). The Out-Of-Slot-Delineation state is entered when lost slot synchronism is declared by the Slot Delineation State Machine (see 17.6.1.1).

When the PLCP declares a Out-Of-Frame or Out-Of-Slot-Delineation condition, it shall start the Timer_P_x (x = A or B). This timer (Timer_P_x) shall be set to 1 ms ± 10 µs. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. This will lead to the transmission of void slots on Bus x (see 17.6.2).

— If the PLCP enters the In-Slot-Delineation state before the timer, Timer_P_x, expires, the timer shall be stopped. The PLCP shall send to the DQDB Layer Ph-DATA indication octets marked as VALID at Ph-SAP_x. The PLCP shall resume its normal operation of processing and generating VC-4s, i.e., process/generate POH octets, perform the mapping of DQDB slots and DQDB Layer Management Information octets into and out of the VC-4s, scramble/descramble segment payloads, and add/subtract the (01010101) bit pattern to/from the HCS field.

— If the timer expires before the SDH transmission system Pointer Interpretation State Machine declares Normal Pointer or before slot synchronism is declared by the Slot Delineation State Machine (see 17.6.1.1), then the PLCP shall enter the Loss-Of-Frame state or Loss-Of-Slot-Delineation state, respectively. The PLCP shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. The PLCP shall transmit on Bus y (y = B or A) an LSS equal to rx_link_dn (see 17.6.3).
  - If the DQDB Layer is capable of becoming head of bus (i.e., the HOB_CAPABLE Flag is set to YES; see 11.6.2), then when the DQDB Layer receives the Ph-STATUS indication equal to DOWN at Ph-SAP_x, the DQDB Layer shall start the Head of Bus Arbitration Timer, Timer_H_w (w = 1 or 2), as defined in 7.1.2, and shall send the HOBS value of WAITING, as defined in 10.2.3.4, on Bus x. Thus the PLCP shall generate EMPTY octets of type SLOT_START, SLOT_DATA, and DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. Octets received at Ph-SAP_y from the DQDB Layer are transmitted on Bus x.
  - If the DQDB Layer is not capable of becoming head of bus (i.e., HOB_CAPABLE Flag is set to NO; see 11.6.2), then the PLCP shall transmit on Bus x an LSS code equal to hob_incapable irrespective of the incoming LSS code on Bus y (see 17.6.3).

17.5 PLCP behavior during DQDB Layer out-of-service

The Physical Layer subsystem of a DQDB node connected to an SDH transmission system shall always be powered up in normal operation. If for some reason, however, the Physical Layer subsystem of a DQDB node is powered down, the nodes downstream and upstream of this node would immediately detect this condition as a transmission system fault and the DQDB subnetwork would begin the fault detection process to reconfigure around the powered-down node.

The PLCP shall have the ability to bypass the DQDB Layer when the DQDB Layer is out of service. The PLCP can use this bypass function to isolate the DQDB node from the subnetwork when the conditions in 11.5.2 are met. When the PLCP bypasses the DQDB Layer, the PLCP, as well as the SDH transmission system, shall continue normal operation (i.e., process/generate the PLCP POH octets etc.). The DQDB slot octets and DQDB Layer Management Information octets (M1 and M2) shall be relayed unmodified through the PLCP. The PLCP shall assume EITHER_BUS as the clock source request.

17.6 PLCP operation

17.6.1 Receiver operation

The PLCP has a receiver function for each bus. In the following, the Bus x (x = A or B) receiver function is described.

17.6.1.1 Slot delineation

Slot delineation shall be achieved using either the H4 octet slot offset indicator method as described in 17.6.1.1.1, or the Header Check Sequence (HCS) method as described in 17.6.1.1.2.

17.6.1.1.1 The H4 octet slot offset indicator method

When using the H4 octet slot offset indicator method, the H4 slot offset indicator value provides slot boundary indication (see 17.3.5). As the VC-4 payload capacity (2340 octets) is not an integer multiple of the DQDB slot length (53 octets), the received H4 slot offset indicator value in two consecutive VC-4s shall be expected to increase by 45 modulo 53. An H4 slot offset indicator value out of range shall be regarded as an unexpected slot offset indicator value.

The transition diagram for the H4 Slot Delineation State Machine is defined in figure 17-5. If the H4 octet slot offset indicator method is used, then each DQDB node has two PLCP H4 Slot Delineation State Machines, one at the receiver for each bus.

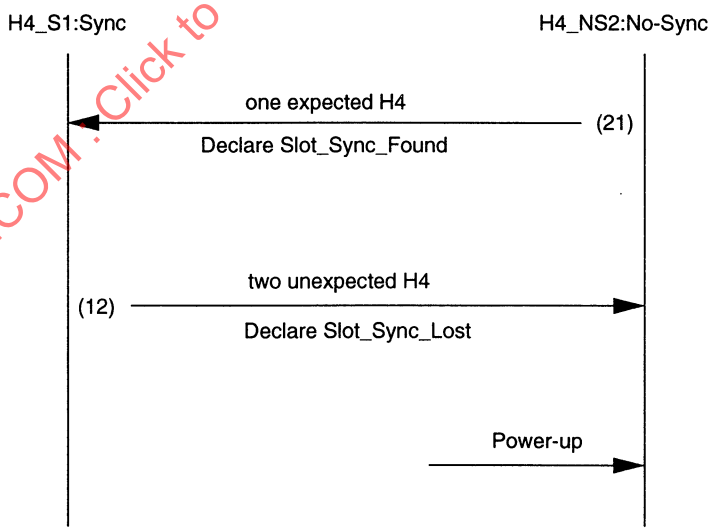


Figure 17-5—H4 Slot Delineation State Machine transition diagram

The H4 Slot Delineation State Machine can be in one of two states: Sync (H4_S1) and No-Sync (H4_NS2). The state machine is powered up in the No-Sync (H4_NS2) state.

State H4_S1: Sync

In this state, the expected slot offset indicator value is calculated by adding 45 modulo 53 to the previously received H4 slot offset indicator value. If the received H4 slot offset indicator value matches the expected H4 slot offset indicator value, this value is used. If a VC-4 is received with an unexpected slot offset indicator value, the Bus x (x = A or B) receiver function shall use the expected slot offset indicator value. If the slot offset indicator value of the next received VC-4 matches the expected slot offset indicator value calculated from either of the two previous received slot offset indicator values, this value shall be used.

Transition 12

If the slot offset indicator value of the received VC-4 does not match the expected slot offset indicator value calculated from either of the two previous received slot offset indicator values, the state machine shall enter the H4_NS2 state. The H4 Slot Delineation State Machine shall declare a Slot_Sync_Lost event to the framing state machine described in 17.6.1.2.

State H4_NS2: No-Sync

In this state, the expected slot offset indicator value is calculated by adding 45 modulo 53 to the previously received H4 slot offset indicator value.

Transition 21

If the slot offset indicator value of the received VC-4 matches the expected slot offset indicator value calculated from the previous received slot offset indicator value, the state machine shall enter the H4_S1 state. The H4 Slot Delineation State Machine shall declare a Slot_Sync_Found event to the Framing State Machine described in 17.6.1.2.

17.6.1.1.2 The HCS method

When using the HCS method, slot boundaries are derived within the VC-4 payload using the correlation between the 3 slot header octets that are protected by the HCS, and the slot header HCS octet itself (see figure 17-2). The HCS is a Cyclic Redundancy Check (CRC) with generating polynomial $x^8 + x^2 + x + 1$.

The transition diagram for the HCS Slot Delineation State Machine is defined in figure 17-6. If the HCS method is used, then each DQDB node has two PLCP HCS Slot Delineation State Machines, one at the receiver for each bus.

The HCS Slot Delineation State Machine can be in one of three states: Sync (HCS_S1), Presync (HCS_P2), and Hunt (HCS_H3). The state machine shall be powered up in the Hunt (HCS_H3) state.

Values of ALPHA=7 and DELTA=6 in suggested in 4.5.1.1 of CCITT Recommendation I.432.

State HCS_S1: Sync

In this state, the Bus x receiver function checks the HCS coding law slot by slot.

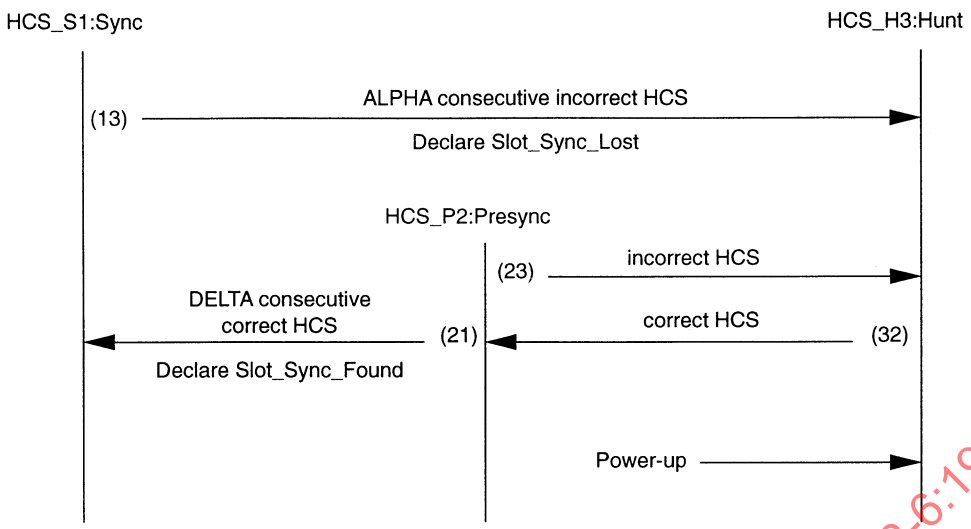


Figure 17-6—HCS Slot Delineation State Machine transition diagram

Transition 13

If the HCS coding law is recognized incorrectly ALPHA times consecutively, then the state machine shall enter the HCS_H3 state. The state machine shall declare a Slot_Sync_Lost event to the Framing State Machine described in 17.6.1.2.

State HCS_P2: Presync

In this state, the Bus x receiver function checks the HCS coding law slot by slot.

Transition 23

If the HCS coding law is recognized incorrectly, then the state machine shall enter the HCS_H3 state.

Transition 21

If the HCS coding law is recognized correctly DELTA times consecutively, then the state machine shall enter the HCS_S1 state. The state machine shall declare a Slot_Sync_Found event to the framing state machine described in 17.6.1.2.

State HCS_H3: Hunt

In this state, the Bus x receiver function checks the HCS coding law octet by octet.

Transition 32

If the HCS coding law is recognized correctly, then the state machine shall enter the HCS_P2 state.

17.6.1.2 Framing State Machine

The transition diagram for the Framing State Machine is defined in figure 17-7. Each DQDB node has two Framing State Machines, one at the receiver for each bus. In the following, the Framing State Machine at the receiver for Bus x (x = A or B) is described. References are made to the Loss-Of-Pointer, Alarm Indication

Signal, and Normal Pointer states of the SDH transmission system Pointer Interpretation State Machine (see CCITT Recommendation G.783, annex B).

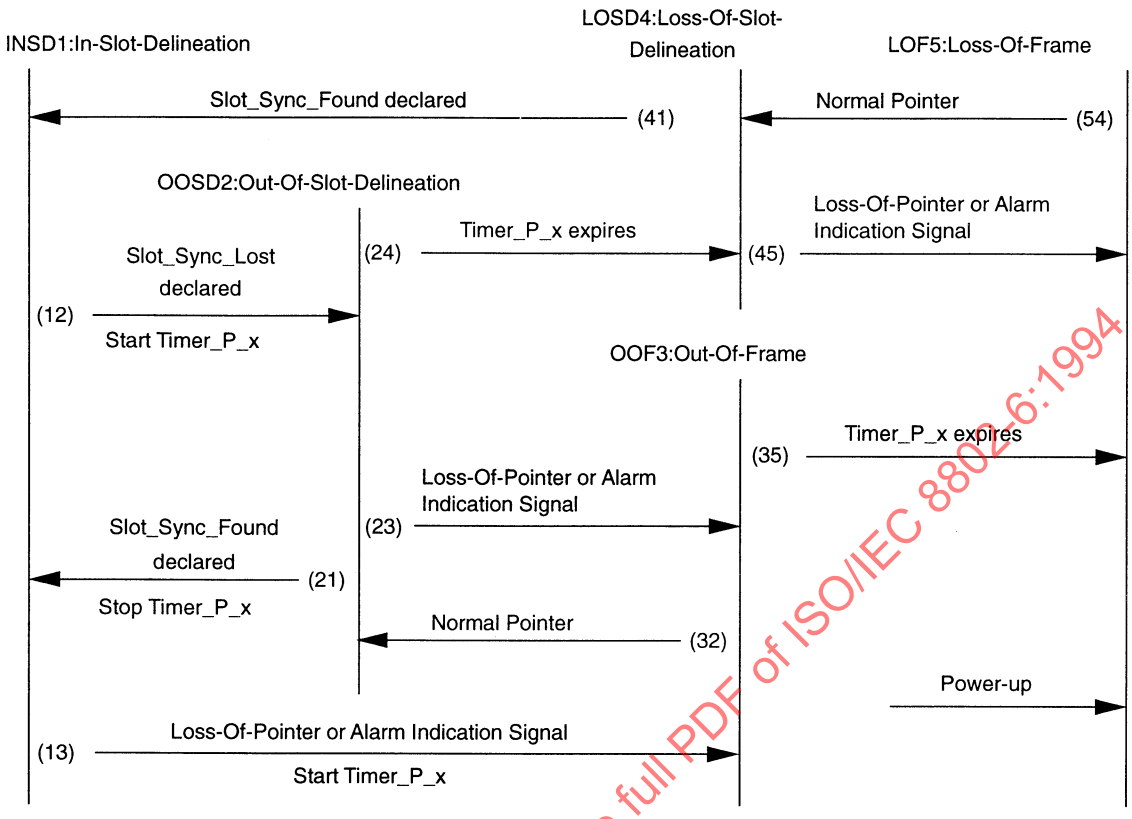


Figure 17-7—Framing State Machine transition diagram

The state machine may be in one of five states: In-Slot-Delineation (INSD1), Out-Of-Slot-Delineation (OOSD2), Out-Of-Frame (OOF3), Loss-Of-Slot-Delineation (LOSD4), and Loss-Of-Frame (LOF5). The state machine shall be powered up in the Loss-Of-Frame (LOF5) state.

State INSD1: In-Slot-Delineation

In this state, the Bus x (x = A or B) receiver function shall process VC-4s, i.e., process POH octets, perform the mapping of DQDB slots and DQDB Layer Management Information octets out of the VC-4s, subtract the (01010101) bit pattern from the HCS field of slot headers, and descramble segment payloads. The Bus x receiver function shall send to the DQDB Layer Ph-DATA indication octets of types SLOT_START, SLOT_DATA, and DQDB_MANAGEMENT marked as VALID at Ph-SAP_x.

Transition 12

If a Slot_Sync_Lost event is declared by the Slot Delineation Machine (see 17.6.1.1), then the state machine shall enter the OOSD2 state. The Bus x receiver function shall start the Timer_P_x.

Transition 13

If a Loss-Of-Pointer or Alarm Indication Signal is declared by the SDH transmission system, then the state machine shall enter the OOF3 state. The Bus x receiver function shall start the Timer_P_x.

State OOSD2: Out-Of-Slot-Delineation

The Bus x (x = A or B) receiver function shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID, except for octets of type DQDB_MANAGEMENT (M1 and M2) marked as VALID, at Ph-SAP_x. This will lead to the transmission of Void slots on Bus x (see 17.6.2).

Transition 21

If a Slot_Sync_Found event is declared by the Slot Delineation State Machine (see 17.6.1.1), then the state machine shall enter the INSD1 state. The Bus x receiver function shall stop and reset the timer, Timer_P_x.

Transition 23

If a Loss-Of-Pointer or Alarm Indication Signal is declared by the SDH transmission system, then the state machine shall enter the OOF3 state. The timer, Timer_P_x, shall remain active.

Transition 24

If the Timer_P_x expires, then the state machine shall enter the LOSD4 state.

State OOF3: Out-Of-Frame

The Bus x receiver function shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. This will lead to the transmission of Void slots on Bus x (see 17.6.2).

Transition 32

If Normal Pointer is declared by the SDH transmission system, then the state machine shall enter the OOSD2 state. The timer, Timer_P_x, shall remain active.

Transition 35

If the Timer_P_x expires, then the state machine shall enter the LOF5 state.

State LOSD4: Loss_Of-Slot-Delineation

When entering this state, the Bus x (x = A or B) receiver function shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to YES, the Bus x receiver function shall send to the DQDB Layer EMPTY Ph-DATA indication octets of types SLOT_START, SLOT_DATA, and DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. If the HOB_CAPABLE Flag is set to NO, the Bus x receiver function shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. This will lead to the transmission of Void slots on Bus x (see 17.6.2). Furthermore, the PLCP shall transmit on Bus x an LSS code equal to hob_incapable, irrespective of the incoming LSS code on Bus y.

Transition 41

If a Slot_Sync_Found event is declared by the Slot Delineation Machine (see 17.6.1.1), then the state machine shall enter the INSD1 state.

Transition 45

If a Loss-Of-Pointer or Alarm Indication Signal is declared by the SDH transmission system, then the state machine shall enter the LOF5 state.

State LOF5: Loss-Of-Frame

When entering this state, the Bus x (x = A or B) receiver function shall send to the DQDB Layer a Ph-STATUS indication equal to DOWN at Ph-SAP_x. If the HOB_CAPABLE Flag is set to YES, the Bus x receiver function shall send to the DQDB Layer EMPTY Ph-DATA indication octets of types SLOT_START, SLOT_DATA, and DQDB_MANAGEMENT marked as VALID at Ph-SAP_x. If the HOB_CAPABLE Flag is set to NO, the Bus x receiver function shall send to the DQDB Layer Ph-DATA indication octets marked as INVALID at Ph-SAP_x. This will lead to the transmission of Void slots on Bus x (see 17.6.2). Furthermore, the PLCP shall transmit on Bus x an LSS code equal to hob_incapable, irrespective of the incoming LSS code on Bus y.

Transition 54

If Normal Pointer is declared by the SDH transmission system, then the state machine shall enter the LOSD4 state.

17.6.2 Transmitter operation

The PLCP has a transmitter function for each bus. In the following, the Bus x (x = A or B) transmitter function will be described.

The Bus x transmitter function shall continuously generate VC-4s, i.e. generate POH octets, and perform the mapping of DQDB slots and DQDB Layer Management Information octets received from Ph-SAP_y (y = B or A) into VC-4s. The VC-4s shall be generated at the rate of the selected 125 µs timing mark (see 17.6.4).

Octets received from the DQDB Layer at Ph-SAP_y of type DQDB_MANAGEMENT shall be transmitted in the M1 and M2 POH octets on Bus x.

Octets received from the DQDB Layer at Ph-SAP_y of type SLOT_START and SLOT_DATA shall be transmitted on Bus x. Slots shall be mapped into the VC-4 payload as described in 17.2.

Continuous octets marked as INVALID or no octets received from the DQDB Layer at Ph-SAP_y shall cause Void slots to be transmitted on Bus x. A Void slot is defined as 53 octets each with the default code of (00000000). Slots shall be mapped into the VC-4 payload as described in 17.2.

The H4 slot offset indicator shall provide slot boundary information, and the PLCP shall add the (01010101) bit pattern to the HCS field and scramble segment payloads.

17.6.3 LSS operations table

The operations table for the LSS is defined in table 17-2. The operations table determines the status of the transmission link according to the state of the Framing State Machine, the incoming LSS, and the Physical Layer Connection State Machine (PLCSM) control. This table supplements table 11-1. Additional states from table 11-1, corresponding to rows 4 and 5, are highlighted in boldface font.

If a DQDB node with HOB_CAPABLE Flag set to NO declares a Ph-STATUS indication equal to DOWN at Ph-SAP_x, then the PLCP shall transmit on Bus x an LSS code equal to hob_incapable, irrespective of the incoming LSS code on Bus y. This node adjacent to a failure would, therefore, be isolated from the DQDB subnetwork.

Table 17-2—LSS operations table

INPUT			OUTPUT	
PLCP Frame State	**PLCSM Control**	**Incoming LSS at Bus x**	**Ph-STATUS at Ph-SAP_x**	**Outgoing LSS at Bus y**
INSD1	NORMAL	Connected	UP	connected
INSD1	NORMAL	rx_link_up	UP	connected
INSD1	NORMAL	rx_link_dn/ hob_incapable	DOWN	rx_link_up
OOSD2/OOF3	**NORMAL**	**X**	**no change**	**rx_link_up**
LOSD4/LOF5	**NORMAL**	**X**	**DOWN**	**rx_link_dn**
X	FORCE_DN	X	DOWN	rx_link_dn

KEY: X = Don't care

17.6.4 Physical Layer Frame Timing operations table

The Physical Layer Frame Timing operations table, table 17-3, determines which 125 µs timing shall be used when the timing source used so far is no longer available. The 125 µs timing is dependent on three inputs:

a) The state of the Framing State Machine at the receiver for Bus A (see figure 17-7).
b) The state of the Framing State Machine at the receiver for Bus B.
c) The Timing Source Request outputs from the CC_z (z = 1, 2, or Default) operations tables in the DQDB Layer, defined in tables 10-12, 10-11, and 10-10b), respectively.

This table supplements the TIming Source information of tables 10-10b), 10-11, and 10-12.

Table 17-3—Physical Layer Frame Timing operations supplementary table

INPUT			OUTPUT
Ph-TIMING-SOURCE request	**Receiver Bus A Framing State Machine state**	**Receiver Bus B Framing State Machine state**	**125 µs Timing**
EXTERNAL_CLOCK	X	X	EXTERNAL_CLOCK
NODE_CLOCK	X	X	NODE_CLOCK
BUS_A or EITHER_BUS currently using BUS_A	INSD1/OOSD2/ LOSD4	X	BUS_A
BUS_A or EITHER_BUS currently using BUS_A	OOF3/LO5	X	NODE_CLOCK or EXTERNAL_CLOCK (note 1)
BUS_B or EITHER_BUS currently using BUS_B	X	INSD1/OO2D2/ LOSD4	BUS_B
BUS_B or EITHER_BUS currently using BUS_B	X	OOF3/LO5	NODE_CLOCK or EXTERNAL_CLOCK (note 1)

KEY: X = Don't care

NOTE—Selection between NODE_CLOCK and EXTERNAL_CLOCK is determined by tables 10-10b), 10-11, and 10-12.

Annex A[1]
Protocol Implementation Conformance Statement (PICS) proforma

(normative)

A.1 Introduction

The supplier of a protocol implementation that is claimed to conform to ISO/IEC 8802-6 : 1994 [ANSI/ IEEE Std 802.6, 1994 Edition] shall complete the following Protocol Implementation Conformance Statement (PICS) proforma.

A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use

— By the protocol implementor, as a checklist to reduce the risk of failure to conform to the standard through oversight
— By the supplier and acquirer, or potential acquirer, of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma
— By the user, or potential user, of the implementation, as a basis for initial checking the possibility of interworking with another implementation (note that, while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICS proformas)
— By a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation

A.2 Instructions for completing the PICS proforma

A.2.1 Status symbols

In this PICS proforma, the following abbreviations are used in defining the status type of a feature, parameter, or capability:

M	=	mandatory field/function
O	=	optional field/function
O.<n>	=	optional field/function, but support of at least one of the group of options labeled by the same numeral <n> is required
C	=	conditional
N/A	=	not applicable
PR	=	prohibited

[1] *Copyright release for PICS proformas:* Users of this standard may freely reproduce the PICS proforma in this annex so that it can be used for its intended purpose and may further publish the completed PICS.

A.2.2 Other symbols

When used in the column labeled Value, the following symbols are used:

xx–yy = from number xx to number yy inclusive

A.2.3 General structure of the PICS proforma

The first part of the PICS proforma, Implementation Identification and Protocol Summary, is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma is a fixed-format questionnaire that is divided into subclauses, each containing a group of individual items. Answers to the questionnaire items are to be provided in the rightmost column, either by simply marking an answer to indicate a restricted choice (usually Yes, No, or Not Applicable), or by entering a value or a set or range of values. (Note that there are some items in which two or more choices from a set of possible answers can apply: all relevant choices are to be marked.

Each item is identified by an item reference in the first column. The second column contains the reason to be answered. The third column contains the reference or references to the material that specifies the item in the main body of the standard. The remaining columns record the status of the item, i.e., whether support is mandatory, optional, prohibited, or conditional, and provide the space for the answers (see also A.2.6 below).

A supplier may also provide, or be required to provide, further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labelled A<i> or X<i>, respectively, for cross-referencing purposes. Where <i> is any unambiguous identification for the item (e.g., simply a numeral), there are no other restrictions on its format and presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformance Statement for the implementation in question.

NOTE—Where an implementation is capable of being configured in more than one way, according to the items listed under Major capabilities (see A.4), a single PICS may be able to describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering some subset of the implementation's configuration capabilities, if this makes for easier and clearer presentation of the information. An example might be for a node that can be configured to support the Default Configuration Control (CC_D) function.

A.2.4 Additional Information

Items of Additional Information allow a supplier to provide further information intended to assist the interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS can be considered complete without any such information. Examples right be an outline of the ways in which a (single) implementation can be set up to operate in a variety of environments and configurations; or a brief rationale, based perhaps on specific application needs, for the exclusion of features that, although optional, are nonetheless commonly present in implementations of the Distributed Queue Dual Bus (DQDB) protocol.

References to items of Additional Information may be entered next to any answer in the questionnaire, and may be included in terms of Exception Information.

A.2.5 Exception Information

It may happen occasionally that a supplier will wish to answer an item with mandatory or prohibited status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No pre-printed answer will be found in the Support column for this. Instead, the supplier shall write the missing answer into the Support column, together with an X<i> reference to an item of Exception Information, and shall provide the appropriate rationale in the Exception item itself.

An implementation for which an Exception item is required in this way does not conform to ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition].

NOTE— A possible reason for the situation described above is that a defect in the standard has been reported, a correction for which is expected to change the requirement not met by the standard.

A.2.6 Conditional status

A.2.6.1 Conditional items

The PICS proforma contains a number of conditional items. These are items for which the status (mandatory, optional, or prohibited) that applies is dependent upon whether or not certain other items are supported, or upon the values supported for other items.

In many cases, whether or not the item applies at all is conditional in this way, as well as the status when the item does apply.

Individual conditional items are indicated by one or more conditional symbols (on separate lines) in the Status column.

A conditional item is indicated with "C<s>" in the Status column where "<s>" is one of M, O, or O.<n>, as described in A.2.1. The Predicate column will contain a predicate, "<pred>," as described in A.2.6.2.

If the value of the predicate in any line of a conditional item is true (see A.2.6.2), the conditional item is applicable, and its status is that indicated by the status symbol following the predicate: the answer column is to be marked in the usual way. If the value of a predicate is false, no answer is required.

A.2.6.2 Predicates

A predicate is one of the following:

a) An item-reference for an item in the PICS proforma: the value of the predicate is true if the item is marked as supported, and is false otherwise.

b) The logical negation symbol "NOT" prefixed to an item-reference. The value of the predicate is true if the value of the predicate formed by omitting the "NOT" symbol is false, and vice versa.

c) A BOOLEAN expression constructed by combining simple predicates using the BOOLEAN operators, AND, OR, and NOT, and parentheses, in the usual way. The value of such a predicate is true if the BOOLEAN expression evaluates to true when the simple predicates are interpreted as described above.

A.3 Identification

A.3.1 Implementation identification

Supplier	
Contact point for queries about the PICS	
Implementation Name(s) and Version(s)	
Other information necessary for full identification, e.g., name(s) and version(s) of machines and/or operating systems; System Name(s)	

NOTES

1—Only the first three items are required for all implementations. Other information may be completed as appropriate in meeting the requirement for full identification.

2—The terms Name and Version should be interpreted appropriately to correspond with a supplier's terminology (e.g., Type, Series, Model).

A.3.2 Protocol summary, ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition]

Identification of Protocol Specification	
Identification of Amendments and Corrigenda to this PICS proforma that have been completed as part of this PICS	
Protocol Version(s) Supported	
Have any Exception items been required (see A.2.5)? No[ ] Yes[ ] (The answer "Yes" means that the implementation does not conform to ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition].)	

Date of Statement: (dd/mm/yy)	

A.3.3 Claimed conformance to ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition]

Item	Feature	References	Status	Predicate	Value	Support
3.1	Are all mandatory features implemented?		M			Y[ ]

(The answer "No" means that the implementation does not conform to ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition].)

A.4 Major capabilities and features commonly used as predicates

A.4.1 Major capabilities

This subclause contains conformance requirements for major capabilities of any implementation for which compliance with ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition] is claimed.

Item	Feature	References	Status	Predicate	Value	Support
MS	Service to LLC	3.1, 5.1	M			Y[ ] N[ ] X__
IS	Isochronous service	3.2, 5.2	O*			Y[ ] N[ ]
CODS	Connection-Oriented Data Service	3.3, 5.3.1	O*			Y[ ] N[ ]

*Specification is not completely contained in this edition of ISO/IEC 8802-6.

A.4.2 Features commonly used as predicates

This subclause contains major optional and conditional features that are commonly used as predicates in the remainder of the document.

Item	Feature	References	Status	Predicate	Value	Support
CC_D	Default Configuration Control	10.2.1, 10.2.2.1, 5.4.2.1, clause FR3	O*			Y[ ] N[ ]
HOB_CC	Head of Bus function	10.2.2.1, 5.4.2.3, 7.5.2, 11.6.2	C: M	CC_D		Y[ ] N[ ] X__
HOB	Head of Bus function without Default Configuration Control	10.2.2.2, 10.2.2.3, 5.4.2.3, 5.4.2.5, 7.5.2, 11.6.2	C: O	NOT CC_D		Y[ ] N[ ]
ETS	External Timing Source	5.4.2.4.1, 7.4.5	O			Y[ ] N[ ]
ACLVCI	Additional Connectionless VCIs	6.3.1.1.1.1, 5.1.1.1.4	O			Y[ ] N[ ]
HEXT	Use of the Header Extension field	6.5.1.3	O			Y[ ] N[ ]
ADD_16	16-bit addressing	6.5.1.2.1	O			Y[ ] N[ ]
ADD_60	60-bit addressing	6.5.1.2.1	O			Y[ ] N[ ]
A_ADD	Support of other than universally administered 48-bit addressing	6.5.1.2.1	O			Y[ ] N[ ]

*Implementation of this feature in a node is optional for a supplier. However, it is mandatory that the user ensure that at least one node on each subnetwork has this capability and that exactly one node has this capability enabled.

A.5 DQDB node functional description

An implementation is required to conform to the functional models defined in the standard in regard to all of the externally observable effects.

A.5.1 Provision of MAC service to LLC

The provision of MAC Service to LLC is mandatory for all implementations conforming to this standard.

A.5.1.1 MAC Convergence Function (MCF) block

A.5.1.1.1 MCF Transmit functions

Item	Feature	References	Status	Predicate	Value	Support
5.1	MCF Transmit function	5.1.1.1, figure 5-3	M			Y[ ] N[ ] X__
5.2	Creation of IMPDU	5.1.1.1, 6.5.1	M			Y[ ] N[ ] X__
5.3	Segmentation of IMPDUs	5.1.1.1	M			Y[ ] N[ ] X__
5.4	Creation of DMPDUs	5.1.1.1, 6.5.2	M			Y[ ] N[ ] X__

A.5.1.1.2 MCF Receive functions

Item	Feature	References	Status	Predicate	Value	Support
5.5	MCF Receive functions	5.1.1.2, figure 5-6	M			Y[ ] N[ ] X__
5.6	Validation of the DMPDU	5.1.1.2	M			Y[ ] N[ ] X__
5.7	Reassembly of the IMPDU	5.1.1.2, 8.2.1	M			Y[ ] N[ ] X__
5.8	Validation of the IMPDU	5.1.1.2	M			Y[ ] N[ ] X__
5.9	Extraction of the MSDU	5.1.1.2.5	M			Y[ ] N[ ] X__

A.5.1.2 Queued Arbitrated (QA) functions block

A.5.1.2.1 QA Transmit functions

Item	Feature	References	Status	Predicate	Value	Support
5.10	QA Transmit functions	5.1.2.1, figure 5-8	M			Y [ ] N[ ] X__
5.11	QA segment creation	5.1.2.1	M			Y[ ] N[ ] X__
5.12	Distributed queuing of QA segments	5.1.2.1, 8.1	M			Y[ ] N[ ] X__

A.5.1.2.2 QA Receive functions

Item	Feature	References	Status	Predicate	Value	Support
5.13	QA Receive functions	5.1.2.2, figure 5-11	M			Y[ ] N[ ] X__
5.14	Busy, QA slot processing	5.1.2.2	M			Y[ ] N[ ] X__
5.15	QA segment header validation	5.1.2.2	M			Y[ ] N[ ] X__
5.16	Convergence function selection	5.1.2.2	M			Y[ ] N[ ] X__

A.5.1.3 MAC Sublayer Service Management functions

All nodes conforming to this standard shall support the Message Identifier Management functions and managed objects described in 9.4.1 through 9.4.5. All other DQDB Layer Management Interface functions are optional for the support of the MAC Sublayer service to the LLC Sublayer.

A.5.2 Provision of isochronous service

The provision of isochronous service is optional. However, if it is provided, all functions with mandatory status have to be implemented.

A.5.2.1 Isochronous Convergence Function (ICF) block

Item	Feature	References	Status	Predicate	Value	Support
5.17	ICF Transmit functions	5.2.1.1	C: M* N/A	IS NOT IS		Y[ ] N[ ] X__
5.18	ICF Receive functions	5.2.1.2	C: M* N/A	IS NOT IS		Y[ ] N[ ] X__

*Note that the ICF could be performing a null function.

A.5.2.2 Pre-Arbitrated (PA) Functions block

Item	Feature	References	Status	Predicate	Value	Support
5.19	PA Transmit functions	5.2.2.1	C: M N/A	IS NOT IS		Y[ ] N[ ] X__
5.20	PA Receive functions	5.2.2.2	C: M N/A	IS NOT IS		Y[ ] N[ ] X__

A.5.2.3 Isochronous service provider management functions

A conforming node that supports the optional isochronous service must support the managed objects and operations described in 9.2.3 and 9.2.5.

A.5.3 Provision of other services

A.5.3.1 Connection-Oriented Data Service

The provision of a Connection-Oriented Data Service is optional. However, if it is provided, all functions with mandatory status have to be implemented.

Item	Feature	References	Status	Predicate	Value	Support
5.21	Interactions between COCF Block and QA Functions block	5.3.1.2	C: M N/A	CODS NOT CODS		Y[ ] N[ ] X__

A.5.4 Common functions

A.5.4.1 Relaying of slot octets and Management Information octets

This subclause does not define any externally observable features.

A.5.4.2 Subnetwork Configuration Control function

These functions are required for the correct operation of a DQDB subnetwork. Not all nodes attached to a DQDB subnetwork need be capable of performing all of these functions.

Item	Feature	References	Status	Predicate	Value	Support
5.22	Support for the fundamental subnetwork requirements	5.4.2.1	M*			Y[ ] N[ ] X__
5.23	Startup of nodes not supporting Default Slot Generation function	5.4.2.2	C: M N/A	NOT CC_D CC_D		Y[ ] N[ ] X__
5.24	Summary of subnetwork head of bus requirements	5.4.2.3	M*			Y[ ] N[ ] X__
5.25	Subnetwork Timing Reference Configuration Control functions	5.4.2.4	M*			Y[ ] N[ ] X__
5.26	Hierarchy for selection of primary subnetwork timing reference	5.4.2.4	M*			Y[ ] N[ ] X__
5.27	Primary Timing Reference function for node not at Head_of_Bus	5.4.2.4	M*			Y[ ] N[ ] X__
5.28	Primary Subnetwork Timing Reference functions for node at Head_of_Bus	5.4.2.4.3	M*			Y[ ] N[ ] X__
5.29	Other Timing Reference function for HOB	5.4.2.4	M*			Y[ ] N[ ] X__
5.30	Nodes that cannot support Head of Bus functions	5.4.2.5	C: M N/A	(NOT HOB) and (NOT CC_D) HOB or CC_D		Y[ ] N[ ] X__
5.31	Default Slot Generator signalling	5.4.2.6, 10.1.2.1	C: M* C: M*	CC_D HOB	DSGS= PRESENT DSGS= NOT PRESENT	Y[ ] N[ ] X__

Item	Feature	References	Status	Predicate	Value	Support
5.32	Head of bus signalling	5.4.2.6, 10.1.2.2	C: M* C: M*	HOB_CC HOB	HOBS= STABLE HOBS= WAITING, NO ACTIVE HOB, or STABLE	Y[ ] N[ ] X__
5.33	External Timing Source signalling	5.4.2.6, 10.1.2.3	C: M* C: M*	HOB_CC OR HOB ETS	ETSS= NOT_ PRESENT ETSS= PRESENT	Y[ ] N[ ] X__

*These requirements are mandatory for a DQDB subnetwork. The role of a particular node may change.

A.5.4.3 Head of Bus functions

These functions are mandatory for a node that is to support Head of Bus functions. However, not all nodes are required to support the Head of Bus functions.

Item	Feature	References	Status	Predicate	Value	Support
5.34	Slot Marking functions	5.4.3.1, 6.2.1, 6.3, 6.4, 9.2.6, 9.2.7	C: M	HOB OR HOB_CC		Y[ ] N[ ] X__
5.35	Bus Identification Functions	5.4.3.3, 10.1.1	C: M	HOB OR CC_D		Y[ ] N[ ] X__

A.5.4.4 MID Page Allocation functions

Item	Feature	References	Status	Predicate	Value	Support
5.36	Head of Bus functions	5.4.4.1	C: M	HOB OR HOB_CC		Y[ ] N[ ] X__
5.37	Page Counter subfield generation	5.4.4.1, 7.2.4, 10.1.3.3, 10.3.1	C: M	HOB OR HOB_CC		Y[ ] N[ ] X__
5.38	Page Reservation subfield generation	5.4.4.1, 10.1.3.1, 10.3.2	C: M	HOB OR HOB_CC		Y[ ] N[ ] X__
5.39	Page Counter Modulus subfield generation	5.4.4.1, 10.1.3.2, 10.3.3	C: M	HOB OR HOB_CC		Y[ ] N[ ] X__
5.40	Node functions	5.4.4.2	M			Y[ ] N[ ] X__
5.41	MID Page List Maintenance	5.4.4.2, 7.3.5, 9.4, 10.3.6	M			Y[ ] N[ ] X__
5.42	Page Counter Operations	5.4.4.2, 10.3.4	M			Y[ ] N[ ] X__
5.43	Keeping MID Page	5.4.4.2, 10.3.5	M			Y[ ] N[ ] X__
5.44	Getting MID Pages	5.4.4.2, 10.3.6	M			Y[ ] N[ ] X__

A.6 DQDB Layer Protocol Data Unit (PDU) formats

This clause describes the formats of the data units as a sequence of fields.

A.6.1 Ordering principles

Item	Feature	References	Status	Predicate	Value	Support
6.1	Octet fields transmitted left-most first	6.1	M			Y[ ] N[ ] X__

A.6.2 Slot

Item	Feature	References	Status	Predicate	Value	Support
6.2	1-octet Access Control field	6.2	M			Y[ ] N[ ] X__
6.3	52-octet segment	6.2	M			Y[ ] N[ ] X__
6.4	ACF busy and SL_TYPE bits	6.2.1, table 6-1	M		Table 6-1	Y[ ] N[ ] X__
6.5	PSR bit set to 1 if previous slot can be cleared, 0 if it cannot be cleared	6.2.1	M			Y[ ] N[ ] X__
6.6	The ACF has 3 REQ_I bits that are set to zero by the Slot Marking function at the head of bus	6.2.1	M			Y[ ] N[ ] X__

A.6.3 Queued Arbitrated (QA) slot

This subclause describes the format of Queued Arbitrated slots.

Item	Feature	References	Status	Predicate	Value	Support
6.7	All bits of QA slot set to 0 at the head of bus	6.3	M			Y[ ] N [ ] X__
6.8	Access unit sets the BUSY bit to 1 to transfer to a QA segment	6.3	M			Y[ ] N[ ] X__
6.9	QA segment has 4-octet segment header, 48-octet segment payload	6.3.1	M			Y[ ] N[ ] X__
6.10	Segment header has a VCI field, Payload_type field, Segment_ Priority field, HCS field	6.3.1.1	M			Y[ ] N[ ] X__
6.11	All-zeros VCI	6.3.1.1.1	PR			Y[ ] N[ ]
6.12	Default connectionless VCI	6.3.1.1.1.1	M		all 1's	Y[ ]N[ ] X__
6.13	Payload_type field	6.3.1.1.2	C: M C: M	NOT ACLVCI ACLVCI	00 default= 00	Y[ ] N[ ] X__ Y[ ] N[ ] X__
6.14	Segment-priority field	6.3.1.1.3	M		00	Y[ ] N[ ] X__
6.15	HCS generator polynomial	6.3.1.1.4	M			Y[ ] N[ ] X__

Item	Feature	References	Status	Predicate	Value	Support
6.16	HCS error checking	6.3.1.1.4	M			Y[ ] N[ ] X__
6.17	HCS error correction	6.3.1.1.4	O			Y[ ] N[ ]
6.18	QA segment payload 48 octets	6.3.1.2	M			Y[ ] N[ ] X__

A.6.4 Pre-Arbitrated (PA) slots

This subclause describes the properties of PA slots.

Item	Feature	References	Status	Predicate	Value	Support
6.19	PA slot will be generated with BUSY set 1, the SL_TYPE 1 and other ACF bits 0	6.4.1	M			Y[ ] N[ ] X__
6.20	PA slot has 4-octet segment header and 48-octet segment payload	6.4.1	M			Y[ ] N[ ] X__
6.21	Segment header has a VCI field, Payload_type field, Segment _Priority field, HCS field	6.4.1.1	M			Y[ ] N[ ] X__
6.22	All-zeros VCI value	6.4.1.1.1	PR			Y[ ] N[ ]
6.23	All-1's VCI value	6.4.1.1.1	PR			Y[ ] N[ ]
6.24	Payload_data field	6.4.1.1.2	M		00	Y[ ] N[ ] X__
6.25	Segment_priority field	6.4.1.1.3	M		00	Y[ ] N[ ] X__
6.26	HCS generation polynomial	6.4.1.1.4	M			Y[ ] N[ ] X__
6.27	HCS error checking	6.4.1.1.4	M			Y[ ] N[ ] X__
6.28	HCS error correction	6.4.1.1.4	O			Y[ ] N[ ]
6.29	PA segment payload 48 octets	6.3.1.2	M			Y[ ] N[ ] X__

A.6.5 Transfer of MAC Service Data Unit (MSDU)

This subclause describes the format and transfer of MSDUs.

A.6.5.1 Initial MAC Protocol Data Unit (IMPDU)

Item	Feature	References	Status	Predicate	Value	Support
6.30	IMPDU format	6.5.1, figure 6-8	M			Y[ ] N[ ] X__
6.31	IMPDU header format	6.5.1	M			Y[ ] N[ ] X__
6.32	Common PDU header format	6.5.1, 6.5.1.1	M			Y[ ] N[ ] X__
6.33	BA size field	6.5.1.1.3	M			Y[ ] N[ ] X__

Item	Feature	References	Status	Predicate	Value	Support
6.34	48-bit MSAP addresses universally administered (destination and source)	6.5.1.2.1	M			Y[ ] N[ ] X__
6.35	48-bit MSAP addresses locally administered (destination and source)	6.5.1.2.1	O			Y[ ] N[ ] X__
6.36	60-bit MSAP addresses public and private (destination and source)	6.5.1.2.1	C: M	ADD_60		Y[ ] N[ ] X__
6.37	16-bit MSAP addresses (destination and source)	6.5.1.2.1	C: M	ADD_16		Y[ ] N[ ] X__
6.38	DA and SA field format	6.5.1.2.2, 6.5.1.2.3	M			Y[ ] N[ ] X__
6.39	Protocol Identification field	6.5.1.2.4.1	M		1	Y[ ] N[ ] X__
6.40	PAD Length field	6.5.1.2.4.1	M			Y[ ] N[ ] X__
6.41	Quality of Service: Delay field	6.5.1.2.5.1	M			Y[ ] N[ ] X__
6.42	Quality of Service: Loss field	6.5.1.2.5.2	M		0	Y[ ] N[ ] X__
6.43	CRC32 Indicator Bit	6.5.1.2.5.3	M			Y[ ] N[ ] X__
6.44	Header Extension Length field	6.5.1.2.5.4	M			Y[ ] N[ ] X__
6.45	Bridging field	6.5.1.2.6	M		0	Y[ ] N[ ] X__
6.46	The Header Extension (HE) field is reserved	6.5.1.3	M			Y[ ] N[ ] X__
6.47	All nodes shall recognize the HE field	6.5.1.3	M			Y[ ] N[ ] X__
6.48	Nodes must be able to receive INFO fields up to 9188 octets for PI = 1	6.5.1.4	M			Y[ ] N[ ] X__
6.49	PAD field	6.5.1.5	M		all 0's	Y[ ] N[ ] X__
6.50	CRC32 field	6.5.1.6	O			Y[ ] N[ ] X__
6.51	The CRC32 field must be recognized by all nodes when CIB is set	6.5.1.6	M			Y[ ] N[ ] X__
6.52	Common PDU trailer	6.5.1.7	M			Y[ ] N[ ] X__

A.6.5.2 Derived MAC Protocol Data Unit (DMPDU)

Item	Feature	References	Status	Predicate	Value	Support
6.53	The DMPDU format	6.52, figures 6-17–6-18, table 6-5	M			Y[ ] N[ ] X__
6.54	The DMPDU header consists of a 2-octet Segment_Type field, a 4-bit sequence number field, and a 10-bit Message Identification (MID) field	6.5.2.1, figure 6-18	M			Y[ ] N[ ] X__
6.55	The Segment_Type field is 00 for COMS, 01 for EOMS, 10 for BOMS, and 11 for SSMs	6.5.2.1.1, table 6-5	M			Y[ ] N[ ] X__
6.56	Sequence Number subfield	6.5.2.1.2, 7.2.6	M			Y[ ] N[ ] X__
6.57	Message Identifier subfield the same for all BOMs, COMs, and EOMs derived from the same IMPDU	6.5.2.1.3	M			Y[ ] N[ ] X__
6.58	Message Identifier subfield for BOMS, COMS, and EOMs	6.5.2.1.3, 5.4.4.2	M			Y[ ] N[ ] X__
6.59	Message Identifier subfield for SSMs	6.5.2.1.3	M		0	Y[ ] N[ ] X__
6.60	Payload Length subfield	6.5.2.2.1	M		4–44	Y[ ] N[ ] X__
6.61	The Payload_CRC subfield used for error detection at the receiving node	6.5.2.2.2	M			Y[ ] N[ ] X__
6.62	Payload_CRC subfield used for error correction of the DMPDU header	6.5.2.2.2	O			Y[ ] N[ ] X__
6.63	Payload_CRC subfield used for error correction of the DMPDU segmentation unit or trailer	6.5.2.2.2	PR			Y[ ] N[ ] X__

A.7 DQDB Layer facilities

This clause contains the conformance requirements for the DQDB Layer facilities of any implementation for which compliance with ISO/IEC 8802-6 : 1994 [ANSI/IEEE Std 802.6, 1994 Edition] is claimed.

A.7.1 Timers

A.7.1.1 Reassembly IMPDU Timer (RIT)

Item	Feature	References	Status	Predicate	Value	Support
7.1	Timer RIT	7.1.1	M			Y[ ] N[ ] X__
7.2	Setting RIT_PERIOD	7.3.1	O			Y[ ] N[ ]
7.3	Start RIT	7.1.1	M			Y[ ] N[ ] X__
7.4	Clear RIT	7.1.1	M			Y[ ] N[ ] X__

A.7.1.2 Head of Bus Arbitration Timer (Timer_H)

Item	Feature	References	Status	Predicate	Value	Support
7.5	Timer_H	7.1.2	C: M*	HOB OR HOB_CC		Y[ ] N[ ] X__
7.6	Setting Timer_H_PERIOD	7.3.1	O			Y[ ] N[ ]
7.7	Start/Stop Timer_H	10.2.3.4	C: M*	HOB OR HOB_CC		Y[ ] N[ ] X__

*Two Head of Bus Arbitration Timers (one for each bus) shall be available for each CC function that can support a Head of Bus function.

A.7.2 Counters

A.7.2.1 Request Counter (REQ_I_CNTR)

Item	Feature	References	Status	Predicate	Value	Support
7.8	6x REQ_I_CNTR*	7.2.1	M		0–65535	Y[ ] N[ ] X__
7.9	Initialize counters	7.2.1	M		0	Y[ ] N[ ] X__
7.10	Minimum saturation of counters	7.2.1	M		0	Y[ ] N[ ] X__
7.11	Maximum saturation of counters	7.2.1	M		65535	Y[ ] N[ ] X__

*Two independent request counters (one for each bus) shall be available at each priority level (0, 1, 2).