Recognized as an

American National Standard (ANSI)

IEEE Std 802.2, 1998 Edition(R2003)

(Incorporating ANSI/IEEE Stds 802.2c-1997,
802.2f-1997, and 802.2h-1997)

(Adopted by ISO/IEC and redesignated as
ISO/IEC 8802-2:1998)

IEEE Standard for Information technology—

Telecommunications and information exchange between systems—

Local and metropolitan area networks—

Specific requirements

Part 2: Logical Link Control

Adopted by the ISO/IEC and redesignated as ISO/IEC 8802-2:1998

Sponsor

LAN/MAN Standards Committee
of the
IEEE Computer Society

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 Basic Reference Model. The functions, features, protocol, and services of the Logical Link Control (LLC) sublayer, which constitutes the top sublayer in the data link layer of the ISO/IEC 8802 LAN protocol, are described. The services required of, or by, the LLC sublayer at the logical interfaces with the network layer, the medium access control (MAC) sublayer, and the LLC sublayer management function are specified. The protocol data unit (PDU) structure for data communication systems is defined using bit-oriented procedures, as are three types of operation for data communication between service access points. In the first type of operation, PDUs are exchanged between LLCs without the need for the establishment of a data link connection. In the second type of operation, a data link connection is established between two LLCs prior to any exchange of information-bearing PDUs. In the third type of operation, PDUs are exchanged between LLCs without the need for the establishment of a data link connection, but stations are permitted to both send data and request the return of data simultaneously.
Keywords: local area networks, protocols; logical link control

---

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

ISBN 1-55937-959-6

7 May 1998 *SH94562*

ANSI/IEEE Std 802.2, 1998 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 that 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 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:

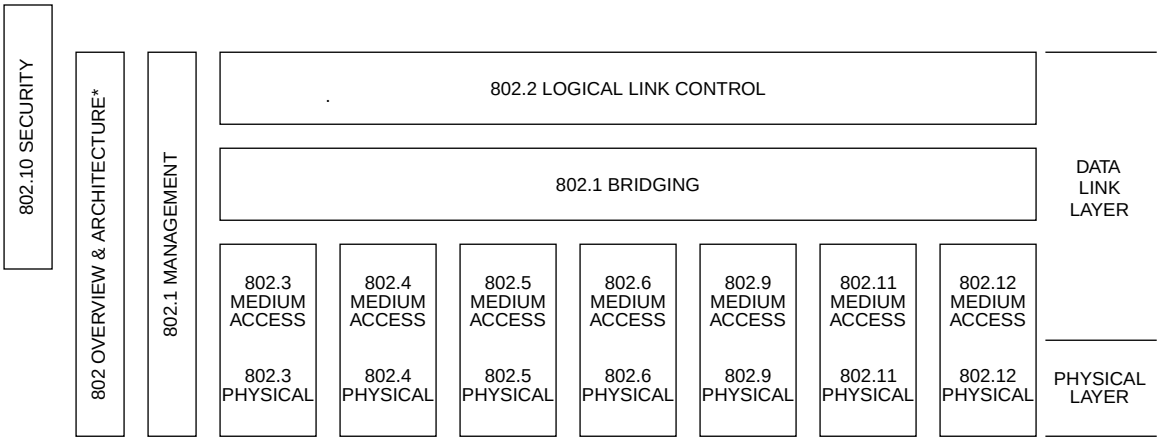
Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Introduction to ANSI/IEEE Std 802.2, 1998 Edition

(This introduction is not a part of ANSI/IEEE Std 802.2, 1998 Edition or of ISO/IEC 8802-2 : 1998.)

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 (OSI) Basic Reference Model (ISO/IEC 7498-1 : 1994). The access standards define seven 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 technologies noted above are as follows:

- IEEE Std 802 — *Overview and Architecture*. This standard provides an overview to the family of IEEE 802 Standards.
- ANSI/IEEE Std 802.1B and 802.1k [ISO/IEC 15802-2] — *LAN/MAN Management.* Defines an OSI management-compatible architecture, and services and protocol elements for use in a LAN/MAN environment for performing remote management.
- ANSI/IEEE Std 802.1D [ISO/IEC 10038] — *Media Access Control (MAC) Bridges.* Specifies an architecture and protocol for the interconnection of IEEE 802 LANs below the MAC service boundary.
- ANSI/IEEE Std 802.1E [ISO/IEC 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.
- ANSI/IEEE Std 802.1G [ISO/IEC 15802-5] — *Remote Media Access Control (MAC) Bridging.* Specifies extensions for the interconnection, using non-LAN communication technologies, of geographically separated IEEE 802 LANs below the level of the logical link control protocol.
- ANSI/IEEE Std 802.2 [ISO/IEC 8802-2] — *Logical Link Control*
- ANSI/IEEE Std 802.3 [ISO/IEC 8802-3] — *CSMA/CD Access Method and Physical Layer Specifications*

- ANSI/IEEE Std 802.4 [ISO/IEC 8802-4] *Token Passing Bus Access Method and Physical Layer Specifications*
- ANSI/IEEE Std 802.5 [ISO/IEC 8802-5] *Token Ring Access Method and Physical Layer Specifications*
- ANSI/IEEE Std 802.6 [ISO/IEC 8802-6] *Distributed Queue Dual Bus Access Method and Physical Layer Specifications*
- ANSI/IEEE Std 802.9 [ISO/IEC 8802-9] *Integrated Services (IS) LAN Interface at the Medium Access Control (MAC) and Physical (PHY) Layers*
- ANSI/IEEE Std 802.10 *Interoperable LAN/MAN Security*
- IEEE Std 802.11 [ISO/IEC DIS 8802-11] *Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications*
- ANSI/IEEE Std 802.12 [ISO/IEC DIS 8802-12] *Demand Priority Access Method, Physical Layer and Repeater Specifications*

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

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

The following additional working group has authorized standards projects under development:

- IEEE 802.14 *Standard Protocol for Cable-TV Based Broadband Communication Network*

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. Thus the conformance test documents for 802.3 are numbered 1802.3.

ANSI/IEEE Std 802.2, 1998 Edition [ISO/IEC 8802-2 : 1998]

This edition of the standard incorporates three supplements: 802.2c-1997, *Conformance Requirements* (ISO/IEC Amendment 3); 802.2f-1997, *Managed Objects Definition for Logical Link Control (LLC)* (ISO/IEC Amendment 6) along with Technical Corrigendum 001; and 802.2h-1997, *Optional Toleration of Duplicate Information Transfer Format Protocol Data Units* (ISO/IEC Amendment 7). In the previous edition, the following supplements were incorporated: 802.2a-1993, *Standard for Flow Control Techniques for Bridged Local Area Networks* (ISO/IEC Amendment 1); 802.2b-1993, *Standard for Acknowledged Connectionless-Mode Service and Protocol* (Type 3 Operation) (ISO/IEC Amendment 2); 802.2d-1993, *Editorial Changes and Technical Corrections* (ISO/IEC Amendment 4); 802.2e-1993, *Bit Delivery Referencing* (ISO/IEC Defect Report 001); and 802.5p-1993, *Standard for Route Determination Entity* (ISO/IEC Amendment 5). The base standard with supplements incorporated into the 1994 edition was reaffirmed by IEEE on 16 September 1997.

This standard contains state-of-the-art material. The area covered by this standard is undergoing evolution. Revisions are possible 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
445 Hoes Lane
P.O. Box 1331
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

The following individuals were participants in the work of this IEEE Project 802.2 Working Group:

David E. Carlson, *Chair*

Om Agrawal
Phil Arneth
Jeff Bobzin
Mark Bauer
Le Biu
Clyde Boenke
Bob Bowen
Bob Bridge*
Chuck Brill
Wayne Brodd*
Fred Burg***
Werner Bux
Jim Campbell
Tony Capel
Ron Cates
Rao Cherukuri
Po Chen***
Jade Chien
Mike Clader
Jerry Clancy*
Rich Collins
Steve Cooper
Mike Coy**
Bob Crowder*
Kirit Dave
John Davidson
Em Delahostria*
Jan Dolphin
Bob Donnan
Bob Douglas
Bill Durrenberger
Rich Fabbri
Eldon Feist*
James Fields*
Larry Foltzer
Ron Floyd
Ingrid Fromm***
Darrell Furlong
Mel Gable
Mike Garvey
Bud Glick
Arie Goldberg
Pat Gonia***
Larry Green***
Gordon Griffiths
Bob Grow
Maris Graube
Ed Harada
Lo Hsieh
Karen Hsing
Kevin Hughes
Marco Hurtado
Bob Husak
Dittmar Janetzky
Ross Jaibaji
George Jelatis
Gabor Kardos
Peggy Karp*
Hal Keen***
Kristin Kocan
Zak Kong*
Sy Korowitz
George Koshy
Don Kotas
Tony Kozlik
Mike Kryskow*
Dave Laffitte
Terry Lawell*
Ron Leuchs
Peter Lin
Jim Lindgren
Laurie Lindsey*
Bill Livingston
Then Tang Liu
Don C. Loughry
Don J. Loughry
Bruce Loyer
Jerry Lurtz
Arthur Miller***
Bill Miller
Ken Miller
Lou Mitta
Bob Moles
Jim Mollenauer
Ware Myers
Lee Neitzel**
Gene Nines
Bill Northup
Brian O'Neil*
Kul Padda
Mahendra Patel
Tom Phinney*
Juan Pimentel
Lavern Pope
Dave Potter
Denis Quy
James Ragsdale**
John Rance
Dan Ratner
Richard Read
Ted Rebenko
John Ricketson
Edouard Rocher
Rob Rosenthal*
Chip Schnarel
Walter Schreuer
Gerard Segarra
Dennis Sosnoski
Robert C. Smith
Mark Stahlman
Monica Stahl
Steve Stearns
Garry Stephens*
Mark Steiglitz*
Kathleen Sturgis
Bob Stover*
Bart Stuck
Dave Sweeton*
Dan Sze*
Vic Tarassov***
Angus Telfer*
Dave Thompson
Fouad Tobagi
Jean-Marie Tourret
Bo Viklund
Bruce Watson
Don Weir*
Dan Sendling
Walter Wheeler
Hugh White
Steve Whiteside
Earl Whitaker*
Ping Wu
Esin Ulug
Hiroshi Yoshida
Wayne Zakowski***
Hank Zannini

*Principal contributors to Project 802.2 at time of initial approval (1989).
**Members of Project 802.2 at time of 1993 supplements' approval.
***Members of Project 802.2 at time of 1997 supplements' approval and reaffirmation of base text.

Additional individuals who made significant contributions were the following:

Don Andrews
Phil Arst
Ron Crane
Walt Elden
Atul Garg
Bryan Hoover
Andrew Huang
Tony Lauck
Andy Luque
Dan Maltbie
Jane Munn
Wendell Nakamine
Liston Neely
Dan Pitt
Robert Printis
Stephen Soto
Joshua Weiss

The following persons were on the original balloting committee that approved this document for submission to the IEEE Standards Board:

William B. Adams
Kit Athul
Chih-Tsai Chen
Michael H. Coden
Robert S. Crowder
George S. Curon
Mitchell Duncan
John E. Emrich
John W. Fendrich
Hal Folts
Harvey Freeman
D. G. Gan
Patrick Gonia
Ambuj Goyal
Maris Graube
J. Scott Haugdahl
Paul L. Hutton
Raj Jain
David M. Kollm
Anthony B. Lake
Mike Lawler
Jaiyong Lee
F. C. Lim
R. S. Little
William D. Livingston
Donald C. Loughry
Andy J. Luque
Richard Miller
Nirode C. Mohanty
John E. Montague
Kinji Mori
David J. Morris
M. Ravindranath Nayak
Arne A. Nilsson
Charles Oestereicher
Young Oh
Udo W. Pooch
John P. Riganati
Gary S. Robinson
Robert Rosenthal
Floyd Ross
S. I. Samoylenko
Julio Gonzalez Sanz
Norman Schneiderwind
D. A. Sheppard
John Spragins
Carel M. Stillebroer
Fred Strauss
Peter Sugar
Efstathios D. Sykas
Daniel Sze
Nathan Tobol
L. David Umbaugh
Thomas A. Varetoni
James Vorhies
Don Weir
Earl J. Whitaker
George B. Wright
Oren Yuen

When the IEEE Standards Board approved IEEE Std 802.2 on 17 August 1989, it had the following membership:

Dennis Bodson, *Chair* **Marco W. Migliaro,** *Vice Chair*
Andrew G. Salem, *Secretary*

Arthur A. Blaisdell
Fletcher J. Buckley
Allen L. Clapp
James M. Daly
Stephen R. Dillon
Donald C. Fleckenstein
Eugene P. Fogarty
Jay Forster*
Thomas L. Hannan
Kenneth D. Hendrix
Theodore W. Hissey, Jr.
John W. Horch
David W. Hutchins
Frank D. Kirschner
Frank C. Kitzantides
Joseph L. Koepfinger*
Edward Lohse
John E. May, Jr.
Lawrence V. McCall
L. Bruce McClung
Donald T. Michael*
Richard E. Mosher
Stig Nilsson
L. John Rankine
Gary S. Robinson
Donald W. Zipse

* Member emeritus

IEEE Std 802.2-1989 was approved by the American National Standards Institute on 12 January 1990.

The following persons were on the balloting committee that approved supplements 802.2a, 802.2b, 802.2d, and 802.2e for submission to the IEEE Standards Board:

William B. Adams
Don Aelmore
Hasan Alkhatib
Kit Athul
Yong Myung Baeg
Alan L. Bridges
Richard Caasi
George Carson
Robert A. Ciampa
Michael H. Coden
Robert Crowder
Jose A. Cueto
Andrew M. Dunn
Philip H. Enslow
Changxin Fan
John W. Fendrich
Harvey A. Freeman
Robert Gagliano
Patrick Gonia
Maris Graube
Craig Guarnieri
Paul L. Hutton
Raj Jain
Jens Kolind
Peter Kornerup
Anthony B. Lake
Jai Yong Lee
Michael E. Lee
Lewis E. Leinenweber
F. C. Lim*
Randolph S. Little
Donald C. Loughry
Nam C. Low
Andy J. Luque
Peter Martini
William McDonald
Darrell B. McIndoe
Richard H. Miller
David S. Millman
C. B. Madhar Mishra
Wen Hsien Lim Moh
John E. Montague
Kinji Mori
Gerald Moseley
Donal O'Mahony
Charles Oestereicher
Art J. Pina
Udo W. Pooch
David Propp
Andris Putnins
Thad L. D. Regulinski
Gary S. Robinson
Philip T. Robinson
Julio Gonzalez Sanz
Norman Schneidewind
Gregory D. Schumacher
Jeffrey R. Schwab
Donald A. Sheppard
Fred J. Strauss
Efstathiois Sykas
Ahmed N. Tantawi
Geoffrey O. Thompson
Robert Tripi
L. David Umbaugh
James T. Vorhies
Donald F. Wier
Raymond Wenig
Earl J. Whitaker
Paul A. Willis
Jen-Kun Yang
Oren Yuen
Stephen Zebrowski

*Did not vote on 802.2a.

Those who participated in the development of IEEE Std 802.5p were as follows:

Robert A. Donnan, *Chair, 802.5*
Phillip Emer, *Chair, Route Determination Entity Task Group*

Floyd Backes
Robert Barrett
Stephen Belisle
Laura Bridge
Fred Burg
Dave Carlson
Claude A. Cartee
Alan Chambers
Johnny A. Chang
Thomas Coradetti
Michael Coy
Robert Dalgleish
Roy C. Dixon
Rick Downs
Candace C. Elder
Richard Fox
William T. Futral
Lionel Geretz
Harry Gold
Larry Green
Tom Gulick
Sharam Hakimi
David Hammond
Charles F. Hanes
John Hart
Douglas Ingraham
Tony Jeffree
Hal Keen
Choon Lee
Chao-yu Liang
George Lin
Arthur Miller
John E. Montague
Lee Neitzel
Alan Oppenheimer
Richard Patti
John Pickens
Dennis Picker
Daniel A. Pitt
Venkat Prasad
Kirk Preiss
Jim Ragsdale
Everett O. Rigsbee III
Phil Robinson
Paul Rosenblum
Bob Ross
Floyd Ross
Jacques Roth
Chris Roussel
Mick Seaman
Himanshu Shah
Richard Siefert
Somsubhra Sikdar
W. Earl Smith
Magnus Stallknecht
Richard Sweatt
Andre Szczepanek
Peter Tan
Jeff Tong
Ric Waller
Chang-Jung Wang
Robert Wu
Amnon Yacoby
Carolyn Zimmer

The following persons were on the balloting committee that approved supplement 802.5p for submission to the IEEE Standards Board:

William B. Adams
Ian F. Akyildiz
Bernhard Albert
Hasan S. Alkhatib
Pat J. Angarano
Kit Athul
William E. Ayen
Tim Batten
George Carson
George C. Chachis
Robert A. Ciampa
Robert Crowder
Robert Donnan
John Emrich
Philip H. Enslow
John W. Fendrich
Harvey A. Freeman
Robert Gagliano
Isaac Ghansah
Patrick Gonia
Scott J. Haugdahl
Richard J. Iliff
Raj Jain
Gary C. Kessler
Farrokh Khatibi
Youngbum Kim
Randolph S. Little
Donald C. Loughry
Joseph F. P. Luhukay
William McDonald
David S. Millman
Kinji Mori
David J. Morris
Ellis S. Nolley
Charles Oestereicher
Jeffrey L. Paige
Art J. Pina
R. I. Prince
Brian Ramelson
Philip T. Robinson
Edouard Y. Rocher
Daniel Rosich
Floyd E. Ross
Julio Gonzalez Sanz
Manoj Kunar Saxena
Gregory D. Schumacher
Donald A. Sheppard
Robert K. Southard
Fred J. Strauss
Efstathiois Sykas
Daniel Sze
Hao Tang
Patricia Thaler
Geoffrey O. Thompson
Mark-Rene Uchida
David L. Umbaugh
James T. Vorhies
Donald F. Weir
Raymond Wenig
Paul A. Willis
Oren Yue
Stephen Zebrowski

When the IEEE Standards Board approved Std 802.5p on 15 September 1993, and Stds 802.2a, 802.2b, 802.2d, and 802.2e on 2 December 1993, it had the following membership:

Wallace S. Read, *Chair* **Donald C. Loughry,** *Vice Chair*
Andrew G. Salem, *Secretary*

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

IEEE Std 802.5p-1993 was approved by the American National Standards Institute on 24 February 1994. IEEE Stds 802.2a-1993, 802.2b-1993, 802.2d-1993, and 802.2e-1993 were approved by the American National Standards Institute on 3 June 1994.

The following persons were on the balloting committees of 802.2c, 802.2f, and 802.2h. The superscripted letters c, f, and h, corresponding to the supplement letter, indicate that the individual balloted only those documents. Those listed without any superscripted letter balloted all three supplements.

William B. Adams
Don Aelmore[c]
Paul D. Amer[c]
Kit Athul[cf]
William E. Ayen
Thomas W. Bailey[cf]
Frederic Bauchot
Manuel J. Betancor[cf]
Kathleen L. Briggs
Peter K. Campbell
James T. Carlo
David E. Carlson
Alan M. Chambers
Frederick N. Chase[c]
Robert S. Crowder
Edward A. Dunlop[c]
Sourav K. Dutta[c]
Paul S. Eastman[ch]
Philip H. Enslow
Changxin Fan[h]
John W. Fendrich
Michael A. Fischer
Harvey A. Freeman
Robert J. Gagliano
D. G. Gan[h]
Gautam Garai
Harry Gold
Julio Gonzalez Sanz[cf]
Maris Graube[c]
Richard J. Iliff
Neil A. Jarvis[fh]
Henry D. Keen[cf]
Peter M. Kelly
Gary C. Kessler
Stephen Barton Kruger
William G. Lane
Lanse M. Leach
Randolph S. Little
Robert D. Love
Joseph G. Maley[c]
Richard McBride
John L. Messenger[fh]
Bennett Meyer
Richard H. Miller
David S. Millman[h]
Warren Monroe
John E. Montague
David J. Morris
James R. Moulton
Wayne D. Moyers
Bongnam Noh[c]
Charles Oestereicher[cf]
Robert O'Hara[fh]
Donal O'Mahony[fh]
Joerg Ottensmeyer[fh]
Roger Pandanda
Ronald C. Petersen
Thomas L. Phinney[cf]
David L. Propp
Vikram Punj[fh]
Edouard Y. Rocher
James W. Romlein
Floyd E. Ross
Michael Salzman
S. I. Samoylenko[cf]
Norman Schneidewind[c]
Lee A. Sendelbach[c]
Donald A. Sheppard
Joseph S. Skorupa[c]
Rosemary Slager[c]
Michael A. Smith[c]
Alex Soceanu[ch]
Fred J. Strauss
Efstathios D. Sykas
Geoffrey O. Thompson[c]
Robert C. Tripi
Mark-Rene Uchida[c]
Yun-Che Wang[c]
Frank J. Weisser[h]
Raymond P. Wenig[c]
Paul A. Willis[c]
Qian-li Yang[c]
Oren Yuen[c]
Jonathan M. Zweig[h]

When the IEEE Standards Board reaffirmed IEEE Std 802.2 and approved IEEE Stds 802.2c, 802.2f, and 802.2h on 16 September 1997, it had the following membership:

Donald C. Loughry, *Chair* **Richard J. Holleman,** *Vice Chair*
Andrew G. Salem, *Secretary*

Clyde R. Camp
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster*
Thomas F. Garrity
Donald N. Heirman
Jim Isaak
Ben C. Johnson
Lowell Johnson
Robert Kennelly
E. G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Lawrence V. McCall
L. Bruce McClung
Marco W. Migliaro
Louis-François Pau
Gerald H. Peterson
John W. Pope
Jose R. Ramos
Ronald H. Reimer
Ingo Rüsch
John S. Ryan
Chee Kiow Tan
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Alan H. Cookson

Kristin Dittmann
IEEE Standards Project Editor

ISO/IEC 8802-2 : 1998 [ANSI/IEEE Std 802.2, 1998 Edition] was approved by the American National Standards Institute (ANSI) on 15 April 1998.

Contents

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

Part 2: Logical Link Control

1. Overview

1.1 Scope and purpose

This International Standard is one of a set of international standards produced to facilitate the interconnection of computers and terminals on a Local Area Network (LAN). It is related to the other international standards by the Reference Model for Open Systems Interconnection (OSI).

NOTE—The exact relationship of the layers described in this International Standard to the layers defined by the OSI Reference Model is under study.

This International Standard describes the functions, features, protocol, and services of the Logical Link Control (LLC) sublayer in the ISO/IEC 8802 LAN Protocol. The LLC sublayer constitutes the top sublayer in the data link layer (see figure 1) and is common to the various medium access methods that are defined and supported by the ISO/IEC 8802 activity. Separate International Standards describe each medium access method individually and indicate the additional features and functions that are provided by the Medium Access Control (MAC) sublayer in each case to complete the functionality of the data link layer as defined in the LAN architectural reference model.

This International Standard describes the LLC sublayer service specifications to the network layer (Layer 3), to the MAC sublayer, and to the LLC sublayer management function. The service specification to the network layer provides a description of the various services that the LLC sublayer, plus underlying layers and sublayers, offer to the network layer, as viewed from the network layer. The service specification to the MAC sublayer provides a description of the services that the LLC sublayer requires of the MAC sublayer. These services are defined so as to be independent of the form of the medium access methodology, and of the nature of the medium itself. The service specification to the LLC sublayer management function provides a description of the management services that are provided to the LLC sublayer. All of the above service specifications are given in the form of primitives that represent in an abstract way the logical exchange

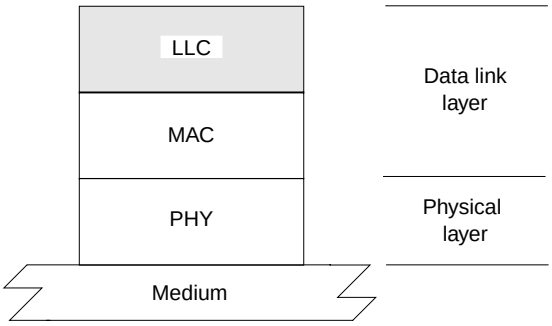


Figure 1—Relationship to LAN reference model

of information and control between the LLC sublayer and the identified service function (network layer, MAC sublayer, or LLC sublayer management function). They do not specify or constrain the implementation of entities or interfaces.

This International Standard provides a description of the peer-to-peer protocol procedures that are defined for the transfer of information and control between any pair of data link layer service access points on a LAN. The LLC procedures are independent of the type of medium access method used in the particular LAN.

To satisfy a broad range of potential applications, three types of data link control operation are included (see clause 4). The first type of operation (see clause 6) provides a data-link-connectionless-mode service across a data link with minimum protocol complexity. This type of operation may be useful when higher layers provide any essential recovery and sequencing services so that these do not need replicating in the data link layer. In addition, this type of operation may prove useful in applications where it is not essential to guarantee the delivery of every data link layer data unit. This type of service is described in this International Standard in terms of "logical data links." The second type of operation (see clause 7) provides a data-link-connection-mode service across a data link comparable to existing data link control procedures provided in International Standards such as HDLC (see ISO/IEC 13239 : 1997[1]). This service includes support of sequenced delivery of data link layer data units, and a comprehensive set of data link layer error recovery techniques. This second type of service is described in this International Standard in terms of "data link connections." The third type of operation (see clause 8) provides an acknowledged-connectionless-mode data unit exchange service, which permits a station to both send data and request the return of data at the same time. Although the exchange service is connectionless, in-sequence delivery is guaranteed for data sent by the initiating station.

This International Standard identifies four distinct "classes" of LLC operation. Class I provides data-link-connectionless-mode service only. Class II provides data-link-connection-mode service plus data-link-connectionless-mode service. Class III provides acknowledged-connectionless-mode service plus data-link-connectionless-mode service. Class IV provides acknowledged-connectionless-mode service plus data-link-connection-mode service plus data-link-connectionless-mode service. Any one of these classes of operation may be supported.

The basic protocols described herein are peer protocols for use in multistation, multiaccess environments. Because of the multistation, multiaccess environment, it shall be possible for a station to be involved in a multiplicity of peer protocol data exchanges with a multiplicity of different stations over a multiplicity of different logical data links and/or data link connections that are carried by a single physical layer (PHY) over a single physical medium. Each unique to-from pairing at the data link layer shall define a separate logical

[1]Information about references can be found in 1.3.

data link or data link connection with separate logical parameters and variables. Except where noted, the procedures described shall relate to each data link layer logical data link or data link connection separately and independently from any other logical data link or data link connection that might exist at the stations involved.

ISO/IEC 10038 : 1993, annex C, provides additional services to allow the MAC service user the ability to determine and use multiple routes through a bridged LAN. This International Standard specifies the provision for an optional Route Determination Entity (RDE) within the LLC sublayer. This entity provides for the discovery and selection of a path (bridged route) for each required data link through the bridged LAN. It does not preclude the LLC service user from providing its own method of discovery and selection of routes.

To evaluate conformance of a particular implementation, it is necessary to have a statement of which capabilities and options have been implemented. Such a statement is called a Protocol Implementation Conformance Statement (PICS), as defined in ISO/IEC 9646-1 : 1994. This International Standard provides such a PICS proforma (Annex A) in compliance with the relevant requirements, and in accordance with the relevant guidance given in ISO/IEC 9646-2 : 1994.

1.2 Standards compatibility

The peer protocol procedures defined in clause 5 utilize some of the concepts and principles, as well as commands and responses, of the balanced data link control procedures known as Asynchronous Balanced Mode (ABM), as defined in ISO/IEC 13239 : 1997. (The ABM procedures provided the basis upon which the ITU-T Recommendation X.25 Level 2 LAPB procedures were defined.) The frame structure defined for the data link layers procedures as a whole is defined in part in clause 3 of this International Standard and in part in those International Standards that define the various MAC procedures. The combination of a MAC sublayer address and an LLC sublayer address is unique to each data link layer service access point in the LAN.

NOTE—This division of data link layer addressing space into separate MAC and LLC address fields is not presently a part of any present ISO data link layer International Standard.

The RDE procedures defined in clause 9 utilize some of the concepts and principles as defined in ISO/IEC 10038 : 1993, annex C.

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 indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

IEC 60955 : 1989, Process data highway, Type C (PROWAY C), for distributed process control systems.[2]

ISO/IEC 7498-1 : 1994, Information technology—Open Systems Interconnection—Basic Reference Model—The Basic Model.[3]

ISO/IEC 7498-4 : 1989, Information processing systems—Open Systems Interconnection—Basic Reference Model—Part 4: Management framework.

---

[2]IEC publications are available from IEC Sales Department, Case Postale 131, 3 rue de Varembé, CH-1211, Genève 20, Switzerland/Suisse. IEC 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.

[3]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 and ISO/IEC 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.

ISO 8824 : 1990, Information technology—Open Systems Interconnection—Specification of Abstract Syntax Notation One (ASN. 1) (provisionally retained edition).

ISO/IEC 8886 : 1996, Information technology—Open Systems Interconnection—Data link service definition.

ISOIIEC 9595 : 1991, Information technology—Open Systems Interconnection—Common management information service definition.

ISO/IEC 9596-1 : 1991, Information technology—Open Systems Interconnection—Common management information protocol—Part 1: Specification.

ISO/IEC 9646-1 : 1994, Information technology—Open Systems Interconnection—Conformance testing methodology and framework—Part 1: General concepts.

ISO/IEC 9646-2 : 1994, Information technology—Open Systems Interconnection—Conformance testing methodology and framework—Part 2: Abstract Test Suite specification.

ISO/IEC 10038 : 1993 [ANSI/IEEE Std 802.1D, 1993 Edition], Information technology—Telecommunications and information exchange between systems—Local area networks—Media access control (MAC) bridges.[4]

ISO/IEC 10040 : 1992, Information technology—Open Systems Interconnection—Systems management overview.

ISO/IEC 10164-1 : 1993, Information technology—Open Systems Interconnection—Systems Management: Object Management Function.

ISO/IEC 10164-2 : 1993, Information technology—Open Systems Interconnection—Systems Management: State Management function.

ISO/IEC 10164-3 : 1993, Information technology—Open Systems Interconnection—Systems Management: Attributes for representing relationships.

ISO/IEC 10164-4 : 1992, Information technology—Open Systems Interconnection—Systems management: Alarm reporting function.

ISO/IEC 10164-5 : 1993, Information technology—Open Systems Interconnection—Systems management: Event Report Management Function.

ISO/IEC 10164-6 : 1993, Information technology—Open Systems Interconnection—Systems Management: Log control function.

ISO/IEC 10165-1 : 1993, Information technology—Open Systems Interconnection—Management information services—Structure of management information: Management Information Model.

ISO/IEC 10165-2 : 1992, Information technology—Open Systems Interconnection—Structure of management information: Definition of management information.

ISO/IEC 10165-4 : 1992, Information technology—Open Systems Interconnection—Structure of management information—Part 4: Guidelines for the definition of managed objects.

[4]This publication is available from the ISO Central Secretariat. It is also available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

ISO/IEC 10165-5 : 1994, Information technology—Open Systems Interconnection—Structure of management information: Generic management information.

ISO/IEC TR 10171 : 1994, Information technology—Telecommunications and information exchange between systems—List of standard data link layer protocols that utilize high-level data link control (HDLC) classes of procedures and list of standardized XID format identifiers and private parameter set identification values.

ISO/IEC 10742 : 1994, Information technology—Telecommunications and information exchange between systems—Elements of management information related to OSI Data Link Layer standards.

ISO/IEC 11575 : 1995, Information technology—Telecommunications and information exchange between systems—Protocol mappings for the OSI Data Link service.

ISO/IEC 13239 : 1997, Information technology—Telecommunications and information exchange between systems—High-level data link control (HDLC) procedures.

ITU-T Recommendation X.25, Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit.[5]

ITU-T Recommendation X.200, Reference model on open systems interconnection for CCITT applications.

1.4 Acronyms and definitions

1.4.1 Acronyms and abbreviations

ABM	Asynchronous Balanced Mode
ACK	ACKnowledge
ADM	Asynchronous Disconnected Mode
ARE	All Routes Explorer
C	Command
C/R	Command/Response
DA	Destination Address
DCE	Data Circuit-terminating Equipment
DISC	DISConnect
DL	Data Link
DLE	Data Link Entity
DM	Disconnected Mode
DSAP	Destination Service Access Point
DTE	Data Terminal Equipment
F	Final
FCS	Frame Check Sequence
FRMR	FRaMe Reject
HDLC	High-level Data Link Control
I	Information
I	Information transfer format
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization

[5]All ITU-T publications are available from the International Telecommunications Union, Sales Section, Place des Nations, CH-1211, Genève 20, Switzerland/Suisse. They are also available in the United States from the U.S. Department of Commerce, Technology Administration, National Technical Information Service (NTIS), Springfield, VA 22161, USA.

ITU-T	International Telecommunications Union—Telecommunications
LAN	Local Area Network
LAPB	Link Access Procedure, Balanced
LLC	Logical Link Control
LSAP	Link Service Access Point
LSB	Least Significant Bit
LSDU	Link layer Service Data Unit
M	Modifier function bit
MAC	Medium Access Control
MO	Managed Object
N(R)	Receive sequence Number
N(S)	Send sequence Number
NSR	Non Source Routed
OSI	Open Systems Interconnection
P	Poll
PDU	Protocol Data Unit
P/F	Poll/Final
PHY	PHYsical
R	Response
RCC	Route Control Component
RDC	Route Determination Component
RDE	Route Determination Entity
REJ	Reject
RIF	Routing Information Field
RII	Routing Information Indicator
RNR	Receive Not Ready
RQ	Route Query
RQC	Route Query Command
RQR	Route Query Response
RR	Receive Ready
RS	Route Selected
RSC	Route Selected Command
S	Supervisory format
S	Supervisory function bit
SA	Source Address
SABME	Set Asynchronous Balanced Mode Extended
SAP	Service Access Point
SRF	Specifically Routed Frame
SRT	Source Routing Transparent (bridge)
SSAP	Source Service Access Point
STE	Spanning Tree Explorer
STR	Spanning Tree Route
TEST	Test
TRR	Timer, Route Response
TRS	Timer, Route Select
U	Unnumbered format
UA	Unnumbered Acknowledgment
UI	Unnumbered Information
V(R)	Receive state Variable
V(S)	Send state Variable
XID	eXchange IDentification

Within the Managed Object definitions and GDMO templates, the following abbrevations are used in the standard-name element of a document identifier when making references to other documents.

DMI CCITT Rec.X.721 (1992)|ISO/IEC 10165-2 : 1992
GMI CCITT Rec.X.723 (1992)|ISO/IEC 10165-5 : 1994
DML ISO/IEC 10742 : 1994

1.4.2 Definitions

For the purpose of this International Standard, the following definitions shall apply:

1.4.2.1 accept: The condition assumed by an LLC upon accepting a correctly received PDU for processing.

1.4.2.2 address fields (DSAP and SSAP): The ordered pair of service access point (SAP) addresses at the beginning of an LLC PDU that identifies the LLC(s) designated to receive the protocol data unit (PDU) and LLC sending the PDU. Each address field is one octet in length.

1.4.2.3 all routes explorer (ARE): A frame that traverses every path and combination of paths through a bridged network.

1.4.2.4 basic status: The capability of an LLC to send or receive a PDU containing an information field.

1.4.2.5 command: In data communications, an instruction represented in the control field of a PDU and sent by an LLC. It causes the addressed LLC(s) to execute a specific data link control function.

1.4.2.6 command PDU: All PDUs sent by an LLC in which the C/R bit in the SSAP address field is equal to “0”.

1.4.2.7 control field (C): The field immediately following the DSAP and SSAP address fields of a PDU. The content of the control field is interpreted by the receiving destination LLC(s) designated by the DSAP address field:

1) As a command, from the source LLC designated by the SSAP address field, instructing the performance of some specific function; or
2) As a response, from the source LLC designated by the SSAP address field.

1.4.2.8 data link: An assembly of two or more terminal installations and the interconnecting communications channel operating according to a particular method that permits information to be exchanged; in this context the term *terminal installation* does not include the data source and the data sink.

1.4.2.9 data link layer: The conceptual layer of control or processing logic existing in the hierarchical structure of a station that is responsible for maintaining control of the data link. The data link layer functions provide an interface between the station higher layer logic and the data link. These functions include address/control field interpretation, channel access and command PDU/response PDU generation, sending, and interpretation.

1.4.2.10 descriptor: The portion of the routing information field that indicates the individual segment and bridge of the network path. A series of descriptors therefore describe a path through the network.

1.4.2.11 exception condition: The condition assumed by an LLC upon receipt of a command PDU that it cannot execute due to either a transmission error or an internal processing malfunction.

1.4.2.12 global (broadcast) DSAP address: The predefined LLC DSAP address (all ones) used as a broadcast (all parties) address. It can never be the address of a single LLC on the data link.

1.4.2.13 group (multicast) DSAP address: A destination address assigned to a collection of LLCs to facilitate their being addressed collectively. The least significant bit shall be set equal to “1”.

1.4.2.14 higher layer: The conceptual layer of control or processing logic existing in the hierarchical structure of a station that is above the data link layer and upon which the performance of data link layer functions are dependent; for example, device control, buffer allocation, LLC station management, etc.

1.4.2.15 information field: The sequence of octets occurring between the control field and the end of the LLC PDU. The information field contents of I, TEST, and UI PDUs are not interpreted at the LLC sublayer.

1.4.2.16 invalid frame: A PDU that either

1) Does not contain an integral number of octets,
2) Does not contain at least two address octets and a control octet, or
3) Is identified by the physical layer or MAC sublayer as containing data bit errors.

1.4.2.17 LLC: That part of a data station that supports the logical link control functions of one or more logical links. The LLC generates command PDUs and response PDUs for sending and interprets received command PDUs and response PDUs. Specific responsibilities assigned to an LLC include

1) Initiation of control signal interchange,
2) Organization of data flow,
3) Interpretation of received command PDUs and generation of appropriate response PDUs, and
4) Actions regarding error control and error recovery functions in the LLC sublayer.

1.4.2.18 MAC: That part of a data station that supports the medium access control functions that reside just below the LLC sublayer. The MAC procedures include framing/deframing data units, performing error checking, and acquiring the right to use the underlying physical medium.

1.4.2.19 N-layer: A subdivision of the architecture, constituted by subsystems of the same rank (N).

1.4.2.20 N-user: An N+1 entity that uses the services of the N-layer, and below, to communicate with another N+1 entity.

1.4.2.21 non-source routed (NSR): Indicates that the frame does not make use of a routing information field (i.e., the RIF is null and the RII is not set).

1.4.2.22 octet: A bit-oriented element that consists of eight contiguous binary bits.

1.4.2.23 path: A bridged route between a source and a destination.

1.4.2.24 peer protocol: The sequence of message exchanges between two entities in the same layer that utilize the services of the underlying layers to effect the successful transfer of data and/or control information from one location to another location.

1.4.2.25 priority (use in primitives): A parameter used to convey the priority required or desired.

1.4.2.26 protocol data unit (PDU): The sequence of contiguous octets delivered as a unit to the MAC sublayer or received as a unit from the MAC sublayer. A valid LLC PDU is at least 3 octets in length, and contains two address fields and a control field. A PDU may or may not include an information field in addition.

1.4.2.27 protocol type (PTYPE): A field in the RDE PDU information field that describes the protocol function of the PDU.

1.4.2.28 remote MAC (RMAC): The MAC component at the remote end of the data link as specified by its unique 48-bit address.

1.4.2.29 remote SAP (RSAP): The SAP at the remote end of a data link as specified by its LLC address.

1.4.2.30 response: In data communications, a reply represented in the control field of a response PDU. It advises the addressed destination LLC of the action taken by the source LLC to one or more command PDUs.

1.4.2.31 response PDU: All PDUs sent by a LLC in which the C/R bit in the SSAP address field is equal to “1”.

1.4.2.32 route: Denotes the information employed to generate routing information. It becomes a routing_information parameter when placed in the MAC primitive. The route explicitly describes the path a frame takes through a bridged network.

1.4.2.33 route query (RQ): An RDE PDU used to explore possible paths between two stations developing a data link. The route query consists of a command PDU (RQC) and a response PDU (RQR).

1.4.2.34 route selected (RS): An RDE PDU used to announce the selection of a path between two stations developing a data link.

1.4.2.35 routing information: The data that explicitly describes the route a frame takes through a bridged network. The routing_information parameter is included in the MA_UNITDATA request and MA_UNITDATA indication MAC primitives.

1.4.2.36 routing information field (RIF): Denotes the routing information field of the source-routed frame format.

1.4.2.37 routing information indicator (RII): An indication that the frame format contains a routing information field (RIF).

1.4.2.38 service: The capabilities and features provided by an N-layer to an N-user.

1.4.2.39 service class (use in primitives): A parameter used to convey the class of service required or desired.

1.4.2.40 source routing: The capability for a source to specify the path that a frame will use to traverse the bridged network.

1.4.2.41 Source Routing Transparent (SRT): The bridging technology defined by ISO/IEC 10038 : 1993, annex C, as an extension to the transparent bridging rules allowing the source station to specify the path through the bridged network (source routing).

1.4.2.42 spanning tree explorer (STE): A type of source-routed frame that will traverse the network following the spanning tree path created by the transparent bridging rules.

1.4.2.43 spanning tree route (STR): A term used to denote the configuration of transparent bridges such that every segment is connected to the root of the network through exactly one path. A frame sent without routing information (NSR) traverses the network on the spanning tree path according to the rules for transparent bridging. A frame sent with a routing type of STE is forwarded through the network on the spanning tree path, but is forwarded by the rules for SRT bridges (note that a bridge that does not support source routing will not forward STE frames).

1.4.2.44 specifically routed frame (SRF): A frame sent with a routing information field that describes the exact path that the frame will take through the bridged network.

This International Standard uses the following term as defined in ISO/IEC 9646-1 : 1994:

— Protocol implementation conformance statement (PICS) proforma

1.4.3 Basic reference model

This part of ISO/IEC 8802 makes use of the following terms defined in ISO/IEC 7498-1 : 1994:

— Data Link layer
— Open system
— (N)-entity
— (N)-protocol
— (N)-service access point

1.4.4 Management framework

This part of ISO/IEC 8802 makes use of the following term defined in ISO/IEC 7498-4 : 1989:

— Managed object

1.4.5 Systems management overview

This part of ISO/IEC 8802 makes use of the following terms defined in ISO/IEC 10040 : 1992:

— Managed object class
— Notification

1.4.6 Common management information service definition

This part of ISO/IEC 8802 makes use of the following term defined in ISO/IEC 9595 : 1991:

— Attribute

1.4.7 Information model

This part of ISO/IEC 8802 makes use of the following terms defined in ISO/IEC 10165-1 : 1993:

— Attribute type
— Behaviour
— Containment
— Distinguished name
— Inheritance
— Name binding
— Package
— Parameter
— Relative distinguished name
— Subclass
— Superclass

1.4.8 Guidelines for the definition of managed objects

This part of ISO/IEC 8802 makes use of the following terms defined in ISO/IEC 10165-4 : 1992:

— Managed object class definition
— Template
— Parameter

1.5 Conformance

1.5.1 Static conformance

1.5.1.1 General requirements

An implementation that claims conformance to this International Standard shall implement the following:

a) LLC Type 1 operation as specified in 6.9.

1.5.1.2 Optional requirements

An implementation that claims conformance to this International Standard may implement any of the following options:

a) LLC Type 2 operation, or LLC Type 3 operation, or LLC Type 2 operation and LLC Type 3 operation. LLC Type 2 operation is specified in 7.9, and LLC Type 3 operation is specified in 8.7, or
b) The flow control technique as specified in annex B, or
c) Initiation of the Duplicate Address Check procedure as specified in 6.9, or
d) Initiation of the TEST function as specified in 6.7, or
e) The RDE as specified in clause 9.

1.5.2 Dynamic conformance

For each function that the PICS states to be supported, the implementation shall exhibit behavior consistent with the implementation of the following:

a) The corresponding data link layer procedures, and
b) The encoding of any transmitted frames

as specified in the clauses to which the PICS proforma entry for the function refers.

1.5.3 PICS proforma

The supplier of a protocol implementation that is claimed to conform to this International Standard shall complete a copy of the PICS proforma in annex A, including the information necessary to identify fully both the supplier and the implementation.

2. LLC sublayer service specifications

2.1 General

This clause covers the services required of, or by, the LLC sublayer at the logical interfaces with the network layer, the MAC sublayer, and the LLC sublayer management function.

In general, the services of a layer (or sublayer) are the capabilities it offers to a 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 2 illustrates this notion of service hierarchy and shows the relationship of the two correspondent N-users and their associated N-layer (or sublayer) peer protocol entities.

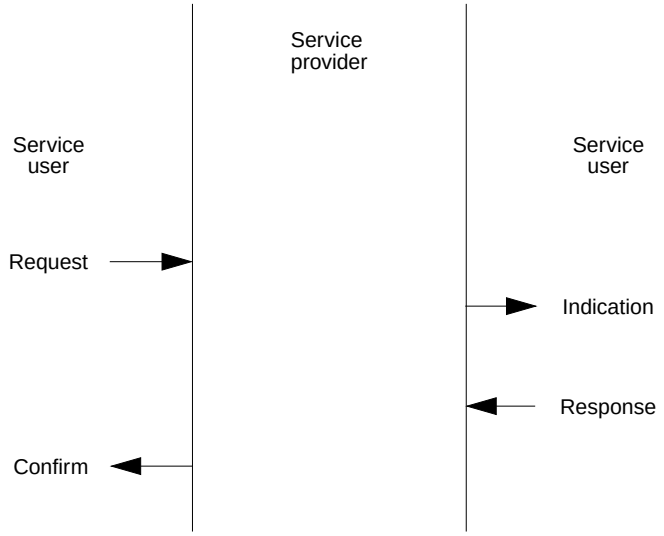


Figure 2—Service primitives

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 an N-layer (or sublayer) service access point 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.

Primitives are of four generic types:

1) **REQUEST:** The request primitive is passed from the N-user to the N-layer (or sublayer) to request that a service be initiated.
2) **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).

3) **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.

4) **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).

Possible relationships among primitive types are illustrated by the time-sequence diagrams shown in figure 3. The figure also indicates the logical relationship of the primitive types. Primitive types that occur earlier in time and are connected by dotted lines in the diagrams are the logical antecedents of subsequent primitive types.

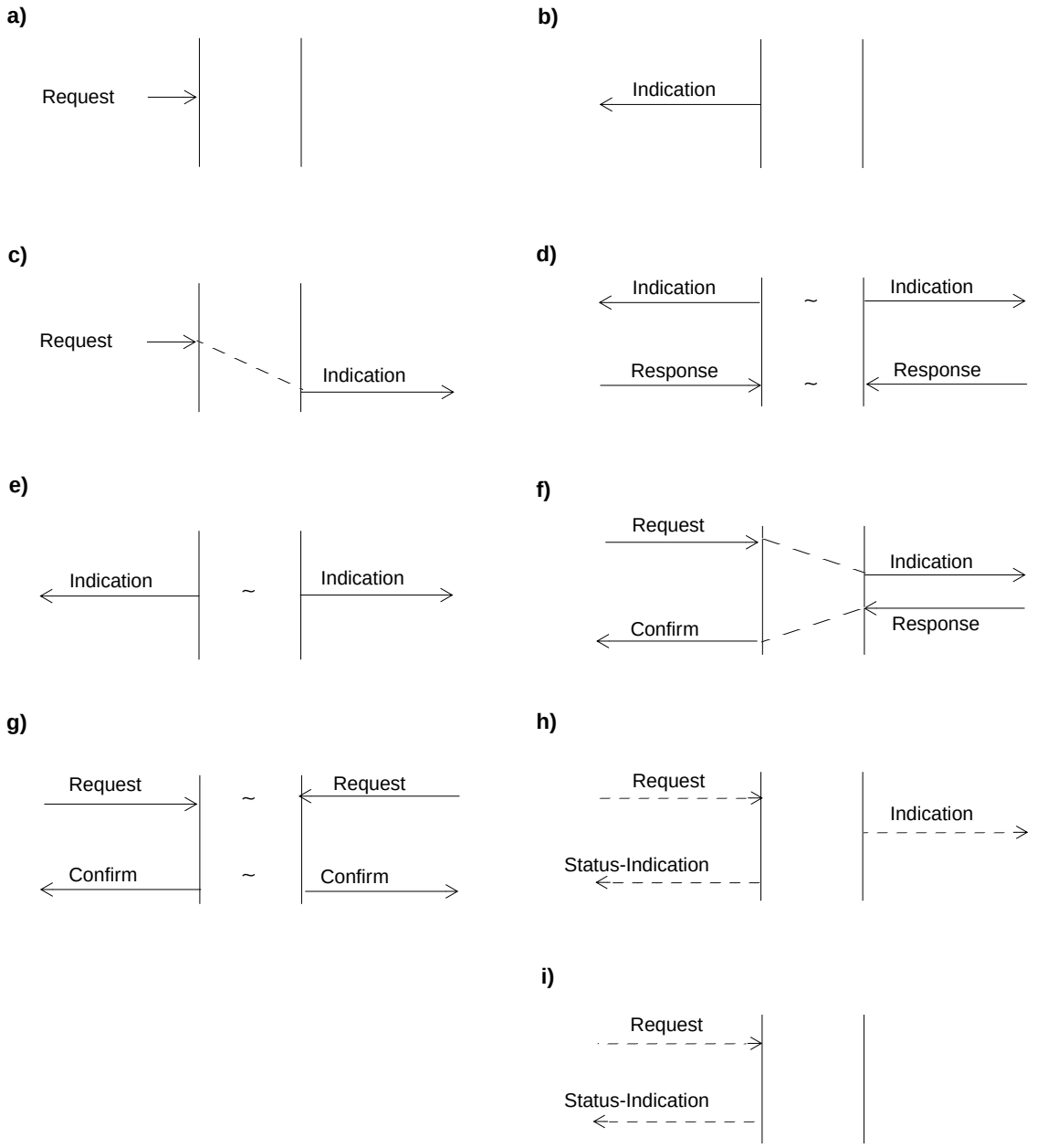


Figure 3—Time-sequence diagrams

2.2 Network layer/LLC sublayer interface service specification

This subclause specifies the services required of the LLC sublayer by the network layer, as viewed from the network layer, to allow a local network layer entity to exchange packets with remote peer network layer entities. The services are described in an abstract way and do not imply any particular implementation or any exposed interface.

Three forms of service are provided: *unacknowledged connectionless-mode*, *connection-mode*, and *acknowledged connectionless-mode*.

1) **Unacknowledged connectionless-mode services:** This set of data transfer services provides the means by which network entities can exchange link service data units (LSDUs) without the establishment of a data link level connection. The data transfer can be point-to-point, multicast, or broadcast.
2) **Connection-mode services:** This set of services provides the means for establishing, using, resetting, and terminating data link layer connections. These connections are point-to-point connections between LSAPs.
    a) The *connection establishment service* provides the means by which a network entity can request, or be notified of, the establishment of data link layer connections.
    b) The *connection-oriented data transfer service* provides the means by which a network entity can send or receive LSDUs over a data link layer connection. This service also provides data link layer sequencing, flow control, and error recovery.
    c) The *connection reset service* provides the means by which established connections can be returned to the initial state.
    d) The *connection termination service* provides the means by which a network entity can request, or be notified of, the termination of data link layer connections.
    e) The *connection flow control service* provides the means to control the flow of data associated with a specified connection, across the network layer/data link layer interface.
3) **Acknowledged connectionless-mode services:** The acknowledged connectionless-mode data unit exchange services provide the means by which network layer entities can exchange link service data units (LSDUs) that are acknowledged at the LLC sublayer, without the establishment of a data link connection. The services provide a means by which a network layer entity at one station can send a data unit to another station, request a previously prepared data unit from another station, or exchange data units with another station. The data unit transfer is point-to-point.

2.2.1 Overview of interactions

2.2.1.1 Unacknowledged connectionless-mode services

2.2.1.1.1 Unacknowledged connectionless-mode data transfer

The primitives associated with unacknowledged connectionless-mode data transfer are as follows:

— DL-UNITDATA request
— DL-UNITDATA indication

The DL-UNITDATA request primitive is passed to the LLC sublayer to request that an LSDU be sent using unacknowledged connectionless-mode procedures. The DL-UNITDATA indication primitive is passed from the LLC sublayer to indicate the arrival of an LSDU.

2.2.1.2 Connection-mode services

2.2.1.2.1 Connection establishment

The primitives associated with connection establishment are as follows:

— DL-CONNECT request
— DL-CONNECT indication
— DL-CONNECT response
— DL-CONNECT confirm

The DL-CONNECT request primitive is passed to the LLC sublayer to request that a data link connection be established between a local LSAP and a remote LSAP. The DL-CONNECT indication primitive is passed from the LLC sublayer to indicate the request by a remote entity to establish a connection to a local LSAP. The DL-CONNECT response primitive is passed to the LLC sublayer to signal acceptance of a connection. The DL-CONNECT confirm primitive is passed from the LLC sublayer to convey the results of the previous associated DL-CONNECT request primitive.

2.2.1.2.2 Connection-mode data transfer

The primitive associated with connection-mode data transfer are as follows:

— DL-DATA request
— DL-DATA indication

The DL-DATA request primitive is passed to the LLC sublayer to request that an LSDU be sent using connection-mode procedures. The DL-DATA indication primitive is passed from the LLC sublayer to indicate the arrival of an LSDU.

2.2.1.2.3 Connection termination

The primitives associated with connection termination are as follows:

— DL-DISCONNECT request
— DL-DISCONNECT indication

The DL-DISCONNECT request primitive is passed to the LLC sublayer to request the immediate termination of a data link connection. The DL-DISCONNECT indication primitive is passed from the LLC sublayer to indicate to the network that a connection has been terminated.

2.2.1.2.4 Connection reset

The primitives associated with connection resetting are as follows:

— DL-RESET request
— DL-RESET indication
— DL-RESET response
— DL-RESET confirm

The DL-RESET request primitive is passed to the LLC sublayer to request that a connection be immediately reset to the initial state. The DL-RESET indication primitive is passed from the LLC sublayer to indicate a connection reset attempt by either a remote entity or the local LLC sublayer. The DL-RESET response primitive is passed to the LLC sublayer to signal acceptance of the reset condition. The DL-RESET confirm

primitive is passed from the LLC sublayer to convey the results of the previous associated DL-RESET request primitive.

2.2.1.2.5 Connection flow control

The primitives associated with connection flow control are as follows:

— DL-CONNECTION-FLOWCONTROL request
— DL-CONNECTION-FLOWCONTROL indication

The DL-CONNECTION-FLOWCONTROL request primitive is passed to the LLC sublayer to control the flow from the LLC sublayer of DL-DATA indication primitives related to a connection. The DL-CONNECTION-FLOW-CONTROL indication primitive is passed from the LLC sublayer to control the flow from the network layer of DL-DATA request primitives related to a connection.

2.2.1.3 Acknowledged connectionless-mode services

2.2.1.3.1 Acknowledged connectionless-mode data unit transmission service

The primitives associated with the acknowledged connectionless-mode data unit transmission service are as follows:

— DL-DATA-ACK request
— DL-DATA-ACK indication
— DL-DATA-ACK-STATUS indication

The DL-DATA-ACK request primitive is passed to the LLC sublayer to request that an LSDU be sent to a remote LLC using acknowledged connectionless-mode data unit transmission procedures. The DL-DATA-ACK indication primitive is passed from the LLC sublayer to indicate the arrival of a command PDU except in the case where this PDU is used only for resynchronization. The DL-DATA-ACK-STATUS indication primitive is passed from the LLC sublayer to convey the results of the previous associated DL-DATA-ACK request primitive.

2.2.1.3.2 Acknowledged connectionless-mode data unit exchange service

The primitives associated with the acknowledged connectionless-mode data unit exchange service are as follows:

— DL-REPLY request
— DL-REPLY indication
— DL-REPLY-STATUS indication

The DL-REPLY request primitive is passed to the LLC sublayer to request that an LSDU be returned from a remote station or that LSDUs be exchanged between stations using acknowledged connectionless-mode data unit exchange procedures. The DL-REPLY indication primitive is passed from the LLC sublayer to indicate the arrival of a command PDU. The DL-REPLY-STATUS indication primitive is passed from the LLC sublayer to convey the results of the previous associated DL-REPLY request primitive.

2.2.1.3.3 Reply data unit preparation

The primitives associated with reply data unit preparation are as follows:

— DL-REPLY-UPDATE request
— DL-REPLY-UPDATE-STATUS indication

The DL-REPLY-UPDATE request primitive is passed to the LLC sublayer with an LSDU to be held by LLC and sent out at a later time when requested to do so by some other station. The DL-REPLY-UPDATE-STATUS indication primitive is passed from the LLC sublayer to convey the results of the previous associated DL-REPLY request primitive.

2.2.2 Detailed service specifications

This subclause describes in detail the primitives and parameters associated with the identified services. Note that the parameters are specified in an abstract sense. The parameters specify the information that must be available to the receiving entity. A specific implementation is not constrained in the method of making this information available.

The “source_address” and “destination_address” parameters provide at a minimum the logical concatenation of the MAC address field (SA and/or DA) and the LLC address field (SSAP and/or DSAP). An implementation of connection-mode services may make use of a locally significant connection identifier to imply source and destination address parameters. The “data” parameter may be provided by actually passing the link service data unit, by passing a pointer, or by other means. The “priority” parameter provides the priority associated with the data unit transfer. The “priority” parameter is passed transparently to the underlying MAC sublayer via the appropriate LLC/MAC primitives; see 2.3. The “reason” parameter provides an explanation of the disconnection, including a request by the remote entity, or an error internal to the LLC sublayer. The “amount” parameter provides information regarding the amount of data that the LLC entity is allowed to pass. The “service_class” parameter indicates whether or not an acknowledge capability in MAC sublayer is to be used for the data unit transfer. The “status” parameter indicates the success or failure of the previous associated data unit transfer request.

2.2.2.1 DL-UNITDATA request

2.2.2.1.1 Function

This primitive is the service request primitive for the unacknowledged connectionless-mode data transfer service.

2.2.2.1.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-UNITDATA request   (
                      source_address,
                      destination_address,
                      data,
                      priority
                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit transfer. The destination_address may specify either an individual or group address. The data parameter specifies the link service data unit to be transferred by the data link layer entity. The priority parameter specifies the priority desired for the data unit transfer.

2.2.2.1.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request that an LSDU be sent to one or more remote LSAP(s) using unacknowledged connectionless-mode procedures.

2.2.2.1.4 Effect on receipt

Receipt of this primitive causes the LLC sublayer to attempt to send the LSDU using unacknowledged connectionless-mode procedures.

2.2.2.1.5 Additional comments

This primitive is independent of any connection with the remote LSAP. A possible logical sequence of primitive associated with successful unacknowledged connectionless-mode data unit transfer is illustrated in figure 3, item (c).

2.2.2.2 DL-UNITDATA indication

2.2.2.2.1 Function

This primitive is the service indication primitive for the unacknowledged connectionless-mode data unit transfer service.

2.2.2.2.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-UNITDATA indication  (
                        source_address,
                        destination_address,
                        data,
                        priority
                        )
```

The source_address and destination_address parameters specify, respectively, the remote and local LSAPs involved in the data unit transfer. The destination address may be the address of a local LSAP, or may be a group address specifying multiple LSAPs, including a local LSAP. The data parameter specifies the link service data unit that has been received by the LLC sublayer entity. The priority parameter specifies the priority desired for the data unit transfer.

2.2.2.2.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate the arrival of an LSDU from the specified remote entity.

2.2.2.2.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.2.5 Additional comments

This primitive is independent of any connection within the remote LSAP. In the absence of errors, the contents of the data parameter are logically complete and unchanged relative to the data parameter in the associated DL-UNITDATA request primitive.

2.2.2.3 DL-CONNECT request

2.2.2.3.1 Function

This primitive is the service request primitive for the connection establishment service.

2.2.2.3.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECT request  (
                        source_address,
                        destination_address,
                        priority
                        )
```

The source_address and destination_address parameters specify the local and remote LSAPs that are to be connected. The priority parameter specifies the priority desired for the connection.

2.2.2.3.3 When generated

This primitive is passed from the network layer to the LLC sublayer when the network layer entity wishes to establish a logical link connection, of a given priority, to a remote LSAP.

2.2.2.3.4 Effect on receipt

The receipt of this primitive by the LLC sublayer causes the local LLC entity to initiate the establishment of a logical link connection with the remote LLC entity.

2.2.2.3.5 Additional comments

A possible logical sequence of primitives associated with successful connection establishment is illustrated in figure 3, item (f).

2.2.2.4 DL-CONNECT indication

2.2.2.4.1 Function

This primitive is the service indication primitive for the connection establishment service.

2.2.2.4.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECT indication   (
                        source_address,
                        destination_address,
                        priority
                        )
```

The source_address and destination_address parameters specify the remote and local LSAPs that are to be connected. The priority parameter indicates the priority desired for the connection.

2.2.2.4.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate that a connection of a certain priority is being requested.

2.2.2.4.4 Effect on receipt

The network layer entity shall issue either a DL-CONNECT response primitive to accept the connection, or a DL-DISCONNECT request primitive to refuse the connection.

2.2.2.4.5 Additional comments

None.

2.2.2.5 DL-CONNECT response

2.2.2.5.1 Function

This primitive is the service response primitive for the connection establishment service.

2.2.2.5.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECT response   (
                      source_address,
                      destination_address,
                      priority
                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs that are to be connected. The priority parameter indicates the priority provided for the connection.

2.2.2.5.3 When generated

This primitive is passed from the network layer to the LLC sublayer to indicate acceptance of the requested connection.

2.2.2.5.4 Effect on receipt

The receipt of this primitive by the LLC sublayer causes the local LLC entity to accept a logical link connection with the remote LLC entity.

2.2.2.5.5 Additional comments

The network layer entity can return the same priority as given in the DL-CONNECT indication primitive or it may select a lower priority. After returning a DL-CONNECT response primitive, the network layer entity assumes that the connection is established.

2.2.2.6 DL-CONNECT confirm

2.2.2.6.1 Function

This primitive is the service confirm primitive for the connection establishment service.

2.2.2.6.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECT confirm  (
                    source_address,
                    destination_address,
                    priority
                    )
```

The source_address and destination_address parameters specify the local and remote LSAPs that are to be connected. The priority parameter indicates the priority provided for the connection.

2.2.2.6.3 When generated

This primitive is passed by the LLC sublayer to the network layer to convey the results of the previous associated DL-CONNECT request primitive. The results indicate that the connection attempt was successful, and specify the priority obtained.

2.2.2.6.4 Effect on receipt

The network layer entity may use this connection for data unit transfer.

2.2.2.6.5 Additional comments

This primitive indicates that the remote network layer entity received and accepted the connection attempt.

2.2.2.7 DL-DATA request

2.2.2.7.1 Function

This primitive is the service request primitive for the connection-mode data unit transfer service.

2.2.2.7.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DATA request  (
                 source_address,
                 destination_address,
                 data
                 )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection. The data parameter specifies the link service data unit to be transferred by the LLC sublayer entity.

2.2.2.7.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request that an LSDU be transferred to a remote LSAP over an existing connection.

2.2.2.7.4 Effect on receipt

The receipt of this primitive by the LLC sublayer causes the LLC sublayer to transfer the LSDU over the specified logical link connection using connection-mode procedures.

2.2.2.7.5 Additional comments

The DL-DATA request primitive does not contain a priority parameter because priority must be uniform for all DL-DATA request primitive in a particular connection.

A possible logical sequence of primitives associated with successful connection-mode data unit transfer is illustrated in figure 3, item (c).

2.2.2.8 DL-DATA indication

2.2.2.8.1 Function

This primitive is the service indication primitive for the connection-mode data unit transfer service.

2.2.2.8.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DATA indication   (
                     source_address,
                     destination_address,
                     data
                     )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection. The data parameter specifies the link service data unit that has been received by the LLC sublayer entity.

2.2.2.8.3 When generated

This primitive is passed by the LLC sublayer to the network layer to indicate the arrival of an LSDU from the specified remote network layer entity over a particular connection.

2.2.2.8.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.8.5 Additional comments

In the absence of errors, the contents of the data parameter are logically complete and unchanged relative to the data parameter in the associated DL-DATA request primitive.

2.2.2.9 DL-DISCONNECT request

2.2.2.9.1 Function

This primitive is the service request primitive for the connection termination service.

2.2.2.9.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DISCONNECT request   (
                        source_address,
                        destination_address
                        )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection to be terminated.

2.2.2.9.3 When generated

This primitive is passed from the network layer to the LLC sublayer when the network layer entity wishes to terminate a connection.

2.2.2.9.4 Effect on receipt

Receipt of this primitive causes the LLC sublayer to immediately terminate the connection.

2.2.2.9.5 Additional comments

All unacknowledged LSDUs are discarded. The connection termination service is an abortive service. That is, no guarantee of delivery can be assumed about data that is not yet acknowledged at a higher layer. Thus, graceful disconnection (i.e., without loss of data) is the responsibility of a higher layer protocol.

A possible logical sequence of primitives associated with successful connection termination is illustrated in figure 3, item (c).

2.2.2.10 DL-DISCONNECT indication

2.2.2.10.1 Function

This primitive is the service indication primitive for the connection termination service.

2.2.2.10.2 Semantics of the service primitive

This primitive shall provide parameters as follows:

```
DL-DISCONNECT indication   (
                           source_address,
                           destination_address,
                           reason
                           )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the terminated connection. The reason parameter specifies the reason for the disconnection. The reasons for disconnection may include a request by the remote entity, or an error internal to the LLC sublayer.

2.2.2.10.3 When generated

This primitive is passed from the LLC sublayer to the network layer to inform the network layer that a connection has been terminated.

2.2.2.10.4 Effect on receipt

The network entity may no longer use this connection for data unit transfer.

2.2.2.10.5 Additional comments

All unacknowledged LSDUs are discarded. The connection termination service is an abortive service. That is, no guarantee of delivery can be assumed about data that is not yet acknowledged at a higher layer. Thus, graceful disconnection (i.e., without loss of data) is the responsibility of a higher layer protocol.

2.2.2.11 DL-RESET request

2.2.2.11.1 Function

This primitive is the service request primitive for the connection reset service.

2.2.2.11.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-RESET request   (
                   source_address,
                   destination_address
                   )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection to be reset.

2.2.2.11.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request that a connection be reset to the initial state.

2.2.2.11.4 Effect on receipt

Receipt of this primitive causes immediate resetting of the connection.

2.2.2.11.5 Additional comments

All unacknowledged LSDUs are discarded. The connection reset service is an abortive service. That is, no guarantee of delivery can be assumed about data that is not yet acknowledged at a higher layer. Thus, graceful reset (i.e., without loss of data) is the responsibility of a higher layer protocol.

A possible logical sequence of primitives associated with successful connection reset is illustrated in figure 3, item (f).

2.2.2.12 DL-RESET indication

2.2.2.12.1 Function

This primitive is the service indication primitive for the connection reset service.

2.2.2.12.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-RESET indication   (
                      source_address,
                      destination_address,
                      reason
                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection that is affected. The reason parameter specifies the cause for the reset indication primitive. One of the reason codes indicates that a reset was requested by a remote network layer entity or a remote LLC sublayer peer [as shown in figure 3, items (f) and (d), respectively]. All other codes indicate that the local LLC sublayer has detected the need for a connection reset [as shown in figure 3, item (b)].

2.2.2.12.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate either that a connection reset has been requested by the remote network layer entity or remote LLC sublayer peer, or that the local LLC sublayer has determined that the data link connection is in need of reinitialization [as shown in figure 3, item (d)].

2.2.2.12.4 Effect on receipt

For remote reset request, the network layer entity shall issue either a DL-RESET response primitive to signal acceptance of the connection reset, or a DL-DISCONNECT request primitive to terminate the connection. For local reset condition indication, the network layer entity shall issue either a DL-RESET request primitive to reinitialize the connection, or a DL-DISCONNECT request primitive to terminate the connection.

2.2.2.12.5 Additional comments

The reasons for the reset may include a request by the remote entity, or an error condition detected by the local LLC sublayer. All unacknowledged LSDUs are discarded. The connection reset service is an abortive service. That is, no guarantee of delivery can be assumed about data that is not yet acknowledged at a higher layer. Thus, graceful reset (i.e., without loss of data) is the responsibility of a higher layer protocol.

2.2.2.13 DL-RESET response

2.2.2.13.1 Function

This primitive is the service response primitive for the connection reset service.

2.2.2.13.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-RESET response   (
                    source_address,
                    destination_address
                    )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection that is affected.

2.2.2.13.3 When generated

This primitive is passed by the network layer to the LLC sublayer to indicate acceptance of the connection reset.

2.2.2.13.4 Effect on receipt

The receipt of this primitive by the LLC sublayer causes the local LLC sublayer entity to complete the connection reset.

2.2.2.13.5 Additional comments

All unacknowledged LSDUs are discarded. The connection reset service is an abortive service. That is, no guarantee of delivery can be assumed about data that is not yet acknowledged at a higher layer. Thus, graceful reset (i.e., without loss of data) is the responsibility of a higher layer protocol.

2.2.2.14 DL-RESET confirm

2.2.2.14.1 Function

This primitive is the service confirm primitive for the connection reset service.

2.2.2.14.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-RESET confirm   (
                   source_address,
                   destination_address
                   )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection that is affected.

2.2.2.14.3 When generated

This primitive is passed from the LLC sublayer to the network layer to inform the network layer that a connection reset has been completed.

2.2.2.14.4 Effect on receipt

The network layer entity may use this connection for data unit transfer.

2.2.2.14.5 Additional comments

This primitive indicates that the remote LLC sublayer entity has acknowledged the reset.

2.2.2.15 DL-CONNECTION-FLOWCONTROL request

2.2.2.15.1 Function

This primitive is the service request primitive for the connection flow control service.

2.2.2.15.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECTION-FLOWCONTROL request  (
                                   source_address,
                                   destination_address,
                                   amount
                                   )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection to be flow controlled. The amount parameter specifies the amount of data the LLC sublayer entity is permitted to pass.

2.2.2.15.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request control of the flow of DL-DATA indication primitives associated with a connection from the LLC sublayer.

2.2.2.15.4 Effect on receipt

Receipt of this primitive causes the LLC sublayer to adjust the amount of data that may be passed to the network layer.

2.2.2.15.5 Additional comments

Control of the flow of data on a connection is independent of control of the flow on other connections. The amount of data permitted to be passed is dynamically updated by each request. If amount is specified as zero, then the associated flow is stopped. Specific implementations may allow amount to be specified in implementation-specific units, and may allow amount to be specified as “infinite.”

A possible logical sequence of primitives associated with a DL-CONNECTION-FLOWCONTROL request is illustrated in figure 3, item (a).

2.2.2.16 DL-CONNECTION-FLOWCONTROL indication

2.2.2.16.1 Function

This primitive is the service indication primitive for the connection flow control service.

2.2.2.16.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-CONNECTION-FLOWCONTROL indication  (
                                      source_address,
                                      destination_address,
                                      amount
                                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs of the connection to be flow controlled. The amount parameter specifies the amount of data that the network layer entity is permitted to pass to avoid data loss.

2.2.2.16.3 When generated

This primitive is passed from the LLC sublayer to the network layer to request control of the flow of DL-DATA request primitive associated with a connection from the network layer.

2.2.2.16.4 Effect on receipt

Receipt of the primitive causes the network layer to adjust the amount of data that it is allowed to pass without data loss.

2.2.2.16.5 Additional comments

Control of the flow of data on a connection is independent of control of the flow on other connections. The amount of data permitted to be passed is dynamically updated by each indication. If amount is specified as zero, then the associated flow is stopped. Specific implementations may allow amount to be specified in implementation-specific units, and may allow amount to be specified as “infinite.”

A possible logical sequence of primitive associated with a DL-CONNECTION-FLOWCONTROL indication is illustrated in figure 3, item (b).

2.2.2.17 DL-DATA-ACK request

2.2.2.17.1 Function

This primitive is the service request primitive for the acknowledged connectionless-mode data unit transfer service. This primitive is used to send a data unit with acknowledgment to another station.

2.2.2.17.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DATA-ACK request   (
                      source_address,
                      destination_address,
                      data,
                      priority,
                      service_class
                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit transfer. The data parameter specifies the link service data unit to be transferred by the LLC sublayer entity. The priority parameter specifies the priority desired for the data unit transfer. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer is to be used for the data unit transfer.

2.2.2.17.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request that an LSDU be sent to a remote LSAP using acknowledged connectionless-mode data unit transfer procedures.

2.2.2.17.4 Effect on receipt

Receipt of this primitive causes the LLC sublayer to attempt to send the LSDU using acknowledged connectionless-mode procedures.

2.2.2.17.5 Additional comments

This primitive can be passed with a null (having zero length) data parameter for the purpose of resynchronizing the remote LLC. This primitive is independent of any connection with the remote LSAP. A possible sequence of DL-DATA-ACK primitives associated with a successful acknowledged connectionless-mode data unit transfer is illustrated in figure 3, item (h).

2.2.2.18 DL-DATA-ACK indication

2.2.2.18.1 Function

This primitive is the service indication primitive for the acknowledged connectionless-mode data unit transfer service.

2.2.2.18.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DATA-ACK indication  (
                        source_address,
                        destination_address,
                        data,
                        priority,
                        service_class
                        )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit transfer. The data parameter specifies the link service data unit that has been received by the LLC sublayer entity. The priority parameter specifies the priority provided for the data unit transfer. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer was used for the data unit transfer.

2.2.2.18.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate the arrival of a non-null, non-duplicate LSDU from a remote network layer entity.

2.2.2.18.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.18.5 Additional comments

This primitive is independent of any connection with the remote LSAP.

2.2.2.19 DL-DATA-ACK-STATUS indication

2.2.2.19.1 Function

This primitive is the service status indication primitive for the acknowledged connectionless-mode data unit transfer service.

2.2.2.19.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-DATA-ACK-STATUS indication   (
                                source_address,
                                destination_address,
                                priority,
                                service_class,
                                status
                                )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit transfer. The priority parameter specifies the priority provided for the data unit transfer. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer was used for the data unit transfer. The status parameter indicates the success or failure of the previous associated acknowledged connectionless-mode data unit transfer request. One status code must indicate complete compliance with the corresponding request. Other codes indicate failure to comply with the request and the reason for the failure.

2.2.2.19.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate the success or failure of the previous associated acknowledged connectionless-mode data unit transfer request.

2.2.2.19.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.19.5 Additional comments

This primitive is independent of any connection with the remote LSAP. For this primitive to convey useful information, it is required that sufficient implicit context information be passed with the DL-DATA-ACK request primitive and the DL-DATA-ACK-STATUS indication primitive to allow the network layer to correlate the status to the appropriate previous request. Success of the data unit transfer request indicates that an LSDU has been transmitted correctly, and for the case of a non-null LSDU, to the best of the LLC sublayer's knowledge (acknowledgment received), the remote LLC sublayer entity has initiated an DL-DATA-ACK indication primitive to a remote network layer entity.

Unsuccessful statuses may indicate various failures at the local or remote stations, such as inability to transfer the request PDU, or no response PDU received from the remote LLC.

2.2.2.20 DL-REPLY request

2.2.2.20.1 Function

This primitive is the service request primitive for the acknowledged connectionless-mode data unit exchange service. This primitive can be used to request a previously prepared data unit from another station, or to exchange data units with another station.

2.2.2.20.2 Semantics of the service primitive

The primitive shall provide parameters as follows:


```
DL-REPLY request   (
                   source_address,
                   destination_address,
                   data,
                   priority,
                   service_class
                   )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit exchange. The data parameter specifies the link service data unit to be transferred by the LLC sublayer entity. The priority parameter specifies the priority desired for the data unit exchange. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer is to be used for the data unit exchange.

2.2.2.20.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request a previously prepared data unit from another LSAP, or to exchange data units with another LSAP using acknowledged connectionless-mode data unit exchange procedures.

2.2.2.20.4 Effect on receipt

Receipt of this primitive causes the LLC sublayer to attempt to send the LSDU using acknowledged connectionless-mode data unit exchange procedures.

2.2.2.20.5 Additional comments

This primitive can be passed with a null (having zero length) data parameter for the purpose of requesting data only (without sending data). The support of the data unit exchange function in which a non-null LSDU may be contained in the DL-REPLY request primitive is optional in a Type 3 LLC. This primitive is independent of any connection with the remote LSAP. A possible sequence of DL-REPLY primitives associated with a successful acknowledged connectionless-mode data unit exchange is illustrated in figure 3, item (h).

2.2.2.21 DL-REPLY indication

2.2.2.21.1 Function

This primitive is the service indication primitive for the acknowledged connectionless-mode data unit exchange service.

2.2.2.21.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-REPLY indication   (
                      source_address,
                      destination_address,
                      data,
                      priority,
                      service_class
                      )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit exchange. The data parameter specifies the link service data unit that has been received by the LLC

sublayer entity. The priority parameter specifies the priority provided for the data unit exchange. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer was used for the data unit exchange.

2.2.2.21.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate either a successful request of an LSDU from the remote network layer entity, or exchange of LSDUs with a remote network layer entity.

2.2.2.21.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.21.5 Additional comments

This primitive is independent of any connection with the remote LSAP. The transfer of a previously prepared LSDU to a requesting station does not destroy the original copy of the LSDU. Subsequent requests for data by any station will cause the transfer of the same LSDU, until the DL-REPLY-UPDATE request primitive is used to replace the LSDU with new information. The support of the data unit exchange function in which a non-null LSDU may be contained in the DL-REPLY indication primitive is optional in a Type 3 LLC.

2.2.2.22 DL-REPLY-STATUS indication

2.2.2.22.1 Function

This primitive is the service status indication primitive for the acknowledged connectionless-mode data unit exchange service.

2.2.2.22.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-REPLY-STATUS indication   (
                             source_address,
                             destination_address,
                             data,
                             priority,
                             service_class,
                             status
                             )
```

The source_address and destination_address parameters specify the local and remote LSAPs involved in the data unit exchange. The data parameter specifies the link service data unit that has been received by the LLC sublayer entity. The priority parameter specifies the priority provided for the data unit exchange. The service_class parameter specifies whether or not an acknowledge capability in the medium access control sublayer was used for the data unit exchange. The status parameter indicates the success or failure of the previous associated acknowledged connectionless-mode data unit exchange request. One status code must indicate complete compliance with the corresponding request. Other codes indicate failure to comply with the request and the reason for the failure.

2.2.2.22.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate the success or failure of the previous associated acknowledged connectionless-mode data unit exchange request.

2.2.2.22.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.22.5 Additional comments

This primitive is independent of any connection with the remote LSAP. For this primitive to convey useful information, it is required that sufficient implicit context information be passed with the DL-REPLY request primitive and the DL-REPLY-STATUS indication primitive to allow the network layer to correlate the status to the appropriate previous request. Success of the data unit exchange request indicates that an LSDU has been transferred correctly, and for the case of a non-null LSDU, to the best of the LLC sublayer's knowledge (acknowledgment received), the remote LLC sublayer entity has initiated a DL-REPLY indication primitive to a remote network layer entity. Further, if an LSDU was requested from the remote LLC, success indicates that the required LSDU was successfully transferred. Unsuccessful statuses may indicate various failures at the local or remote stations, such as inability to transfer the request PDU, no response PDU received from the remote LLC, or no LSDU available at the remote LLC.

2.2.2.23 DL-REPLY-UPDATE request

2.2.2.23.1 Function

This primitive is the service request primitive for the reply data unit preparation service.

2.2.2.23.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-REPLY-UPDATE request   (
                          source_address,
                          data
                          )
```

The source_address parameter specifies the local LSAP to be associated with the data unit preparation. The data parameter specifies the link service data unit to be held by LLC, in preparation for transfer at a later time when requested.

2.2.2.23.3 When generated

This primitive is passed from the network layer to the LLC sublayer to request that an LSDU be associated with a local LSAP and held by LLC.

2.2.2.23.4 Effect on receipt

The receipt of this primitive by the LLC sublayer causes the LLC sublayer to associate the LSDU with a local LSAP and hold the LSDU for future access.

2.2.2.23.5 Additional comments

Once an LSDU has been associated with a local LSAP, that LSDU can be transferred to other stations using the acknowledged connectionless-mode response PDU as often as requested by other stations (without the need for additional DL-REPLY-UPDATE request primitives from the network layer). A subsequent DL-REPLY-UPDATE request primitive from the network layer for the specified LSAP serves to replace the currently associated LSDU with a new LSDU.

A possible sequence of DL-REPLY-UPDATE primitives associated with successful reply data unit preparation is illustrated in figure 3, item (i).

2.2.2.24 DL-REPLY-UPDATE-STATUS indication

2.2.2.24.1 Function

This primitive is the service indication primitive for the reply data unit preparation service.

2.2.2.24.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
DL-REPLY-UPDATE-STATUS indication   (
                                    source_address,
                                    status
                                    )
```

The source_address parameter specifies the local LSAP associated with the data unit preparation. The status parameter indicates the success or failure of the previous associated data unit preparation request.

2.2.2.24.3 When generated

This primitive is passed from the LLC sublayer to the network layer to indicate the success or failure of the previous associated data unit preparation request.

2.2.2.24.4 Effect on receipt

The effect on receipt of this primitive by the network layer is unspecified.

2.2.2.24.5 Additional comments

If the status is successful, this primitive indicates that the LLC sublayer has associated the LSDU with the local LSAP. A failure status indicates that the LSDU could not be associated with the local LSAP (possibly because of inability to access a shared data area). For this primitive to convey useful information, it is required that sufficient implicit context information be passed with the DL-REPLY-UPDATE request primitive and the DL-REPLY-UPDATE-STATUS indication primitive to allow the network layer to correlate the status to the appropriate previous request.

2.3 LLC sublayer/MAC sublayer interface service specification

This subclause specifies the services required of the MAC sublayer by the LLC sublayer to allow the local LLC sublayer entity to exchange LLC data units with peer LLC sublayer entities. The services are described in an abstract way and do not imply any particular implementation or any exposed interface.

NOTE—Work is in progress to produce a single-service specification that is common to all the MAC sublayers. When this is available, it will be referenced in this International Standard instead of the current MAC service text.

2.3.1 Overview of interactions

— MA-UNITDATA request
— MA-UNITDATA indication
— MA-UNITDATA-STATUS indication

2.3.2 Detailed service specification

2.3.2.1 MA-UNITDATA request

2.3.2.1.1 Function

This primitive requests the transfer of an MSDU from a local LLC sublayer entity to a single peer LLC sublayer entity, or multiple peer LLC sublayer entities in the case of group addresses.

2.3.2.1.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
MA-UNITDATA request   (
                      source_address,
                      destination_address,
                      routing_information,
                      data,
                      priority,
                      service_class
                      )
```

The source_address parameter shall specify an individual MAC sublayer entity address. The destination_address parameter shall specify either an individual or a group MAC sublayer entity address. Together they must contain sufficient information to create the SA and DA fields that are appended to the LLC SDU by the local MAC sublayer entity as well as any physical layer address information (e.g., transmit frequency in broadband applications). The routing_information parameter specifies the route desired for the data unit transfer (a null value indicates source routing is not to be used). The data parameter specifies the MAC service data unit to be transmitted by the MAC sublayer entity, which includes the DSAP, SSAP, C, and information (if present) fields as specified in clause 3, as well as sufficient information for the MAC sublayer entity to determine the length of the data unit. The priority parameter specifies the priority desired for the data unit transfer. The service_class parameter specifies the class of service desired for the data unit transfer.

2.3.2.1.3 When generated

This primitive is generated by the LLC sublayer entity to request a MSDU be transferred to a peer LLC sublayer entity or entities. This can be as a result of a request from higher layers of protocol, or from a LSDU generated internally in the LLC sublayer, such as required by Type 2 operation.

2.3.2.1.4 Effect on receipt

The receipt of this primitive shall cause the MAC sublayer entity to append all MAC-specified fields, including DA, SA, and any fields that are unique to the particular medium access method, and pass the properly formatted frame to the lower layers of protocol for transfer to the peer MAC sublayer entity or entities for subsequent transfer to the associated LLC sublayer(s).

2.3.2.1.5 Additional comments

A possible logical sequence of primitives associated with successful MAC service data unit transfer is illustrated in figure 3, item (c).

2.3.2.2 MA-UNITDATA indication

2.3.2.2.1 Function

This primitive defines the transfer of a MSDU from the MAC sublayer entity to the LLC sublayer entity, or entities in the case of group addresses. In the absence of errors, the contents of the data parameter are logically complete and unchanged relative to the data parameter in the associated MA-UNITDATA request primitive.

2.3.2.2.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
MA-UNITDATA indication   (
                         source_address,
                         destination_address,
                         routing_information,
                         data,
                         reception_status,
                         priority,
                         service_class
                         )
```

The source_address parameter shall specify an individual address as specified by the SA field of the incoming frame. The destination_address parameter shall be either an individual or a group address as specified by the DA field of the incoming frame. The routing_information parameter specifies the route used for the data unit transfer (for MAC sublayers that do not support source routing, this field will be set to null). The data parameter specifies the MAC service data unit as received by the local MAC entity. The reception_status parameter indicates the success or failure of the incoming frame. The priority parameter specifies the priority desired for the data unit transfer. The service_class parameter specifies the class of service desired for the data unit transfer.

2.3.2.2.3 When generated

The MA-UNITDATA indication primitive is passed from the MAC sublayer entity to the LLC sublayer entity or entities to indicate the arrival of a frame at the local MAC sublayer entity. Frames are reported only if at the MAC sublayer they are validly formatted, received without error, and their destination address designates the local MAC sublayer entity.

2.3.2.2.4 Effect on receipt

The effect on receipt of this primitive by the LLC sublayer is dependent on the validity and content of the frame.

2.3.2.2.5 Additional comments

If the local MAC sublayer entity is designated by the destination_address parameter of an MA-UNITDATA request primitive, the indication primitive will also be invoked by the MAC sublayer entity to the local LLC sublayer entity. This full duplex characteristic of the MAC sublayer may be due to unique functionality within the MAC sublayer or full duplex characteristics of the lower layers (e.g., all frames transmitted to the broadcast address will invoke MA-UNITDATA indication primitives at all stations in the network including the station that generated the request).

2.3.2.3 MA-UNITDATA-STATUS indication

2.3.2.3.1 Function

This primitive has local significance and shall provide the LLC sublayer with status information for a previous associated MA-UNITDATA request primitive.

2.3.2.3.2 Semantics of the service primitive

The primitive shall provide parameters as follows:

```
MA-UNITDATA-STATUS indication   (
                                source_address,
                                destination_address,
                                transmission_status,
                                provided_priority,
                                provided_service_class
                                )
```

The source_address parameter must be an individual MAC sublayer entity address as specified in the associated MA-UNITDATA request primitive. The destination_address parameter shall be either an individual or a group MAC sublayer entity address as specified in the associated MA-UNITDATA request primitive. The transmission_status parameter is used to pass status information back to the local requesting LLC sublayer entity. The types of status that can be associated with this primitive are dependent on the particular implementation as well as the type of MAC sublayer that is used (e.g., “excessive collisions” may be a status returned by a CSMA/CD MAC sublayer entity). The provided_priority parameter specifies the priority that was used for the associated data unit transfer. The provided_service_class parameter specifies the class of service provided for the data unit transfer.

2.3.2.3.3 When generated

The MA-UNITDATA-STATUS indication primitive is passed from the MAC sublayer entity to the LLC sublayer to indicate the status of the service provided for a previous associated MA-UNITDATA request primitive.

2.3.2.3.4 Effect on receipt

The effect on receipt of this primitive by the LLC sublayer is dependent upon the type of operation employed by the LLC sublayer entity.

2.3.2.3.5 Additional comments

It is assumed that sufficient information is available to the LLC sublayer entity to associate the status with the appropriate request.

2.4 LLC sublayer/LLC sublayer management function interface service specification

This matter is the subject of further ongoing study and resolution.

3. LLC PDU structure

3.1 General

This clause defines in detail the LLC PDU structure for data communication systems using bit-oriented procedures. It defines the relative positions of the various components of the PDU. It defines the method for representing data link layer service access point addresses (to or from network layer entities). It defines a partition of these addresses into individual and group addresses. Details of the control and information field allocation are specified in clause 5.

3.2 LLC PDU format

All LLC PDUs shall conform to the format shown in figure 4.

DSAP address	SSAP address	Control	Information
8 bits	8 bits	8 or 16 bits	M*8 bits

DSAP address	=	Destination service access point address field
SSAP address	=	Source service access point address field
Control	=	Control field [16 bits for formats that include sequence numbering, and 8 bits for formats that do not (see 5.2)]
Information	=	Information field
*	=	Multiplication
M	=	An integer value equal to or greater than 0. (Upper bound of M is a function of the medium access control methodology used.)

Figure 4—LLC PDU format

3.3 Elements of the LLC PDU

3.3.1 Address fields

Each LLC PDU shall contain two address fields—the Destination Service Access Point (DSAP) address field and the Source Service Access Point (SSAP) address field, in that order. Each address field shall contain only a single address. The DSAP address field shall identify the one or more service access points for which the LLC information field is intended. The SSAP address field shall identify the specific service access point from which the LLC information field was initiated.

3.3.1.1 Address field representation

The representation of each address field shall be as shown in figures 5 and 6.

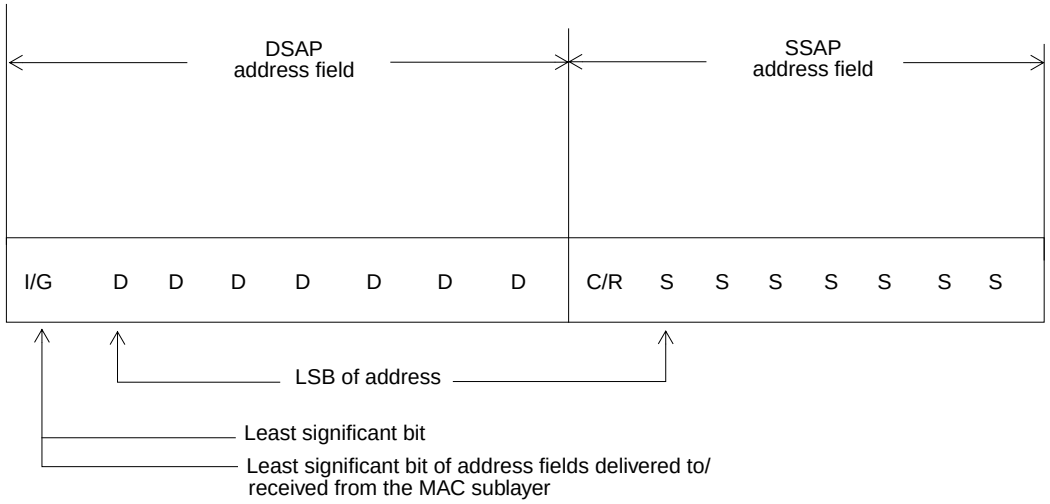


Figure 5—DSAP and SSAP address field formats

Figure 6—Global DSAP address field format

1) Each address field shall contain one octet.

2) Each address field shall contain 7 bits of actual address and one bit that shall be used in the DSAP address field to identify the DSAP address as either an individual or a group address (called the address type designation bit) and in the SSAP address field to identify that the LLC PDU is a command or a response (called the command/response identifier bit).

3) The address type designation bit shall be located in the least significant bit position of the DSAP address field. If this bit is “0”, it shall indicate that the address is an individual DSAP address. If this bit is “1”, it shall indicate that the address is a group DSAP address that identifies none, one or more, or all of the service access points that are serviced by the LLC entity.

4) The command/response identifier bit shall be located in the least significant bit position of the SSAP address field. If this bit is “0”, it shall indicate that the LLC PDU is a command. If this bit is “1”, it shall indicate that the LLC PDU is a response.

3.3.1.2 Address usage

An individual address shall be usable as both an SSAP and a DSAP address; a null address shall be usable as both an SSAP and a DSAP address; a group address shall be usable only as a DSAP address.

All “1”s in the DSAP address field (i.e., the address type designation bit set to “1”, and the seven address bits set to “1”) is predefined to be the “Global” DSAP address. This DSAP address designates a group consisting of all DSAPs actively being serviced by the underlying MAC service access point address(es).

All “0”s in the DSAP or SSAP address field, (i.e., the address type designation bit set to “0”, and the seven address bits set to “0”) is predefined to be the “Null” address. The Null service access point address designates the LLC that is associated with the underlying MAC service access point address, and is NOT used to identify any service access point to the network layer or any service access point to an associated layer management function.

Addresses 01000000 and 11000000 are designated as the individual and the group addresses, respectively, for the LLC sublayer management function at the station. LLC address 01100101 is designated as the RDE SAP address. Other addresses with the next to low-order bit set to “1” are reserved for ISO definition.

3.3.2 Control field

The control field shall consist of one or two octets that shall be used to designate command and response functions, and shall contain sequence numbers when required. The content of this field shall be as described in clause 5.

3.3.3 Information field

The information field shall consist of any integral number (including zero) of octets.

3.3.4 Invalid LLC PDU

An invalid LLC PDU shall be defined as one which meets at least one of the following conditions:

1) It is not an integral number of octets in length.
2) Its length is less than 3 octets (one-octet control field) or 4 octets (two-octet control field).

Invalid LLC PDUs shall be ignored.

4. LLC types and classes of procedures

4.1 General

LLC defines three types of operation for data communication between service access points.

1) *Type 1 Operation.* With Type 1 operation, PDUs shall be exchanged between LLCs without the need for the establishment of a data link connection. In the LLC sublayer these PDUs shall not be acknowledged, nor shall there be any flow control or error recovery in Type 1 operation.
2) *Type 2 Operation*. With Type 2 operation, a data link connection shall be established between two LLCs prior to any exchange of information-bearing PDUs. The normal cycle of communication between two Type 2 LLCs on a data link connection shall consist of the transfer of PDUs containing information from the source LLC to the destination LLC, acknowledged by PDUs in the opposite direction.
3) *Type 3 Operation.* With Type 3 operation, PDUs shall be exchanged between LLC entities without the need for the establishment of a data link connection. In the LLC sublayer, PDUs that may or may not bear information shall be acknowledged. The acknowledgment function shall be accomplished by the destination LLC returning to the source LLC a specific response in a separate PDU that contains status information and may or may not bear user information.

With Type 2 operation, the control of traffic between the source LLC and the destination LLC shall be effected by means of a numbering scheme, which shall be cyclic within a modulus of 128 and measured in terms of PDUs. An independent numbering scheme shall be used for each source/destination LLC pair. Each such pairing shall be defined to be a logical point-to-point data link connection between data link layer service access points and shall take into account the DA and SA addressing that is part of the MAC sublayer. The acknowledgment function shall be accomplished by the destination LLC informing the source LLC of the next expected sequence number. This shall be done either in a separate PDU, not containing information, or within the control field of a PDU containing information.

LLC Type 2 procedures shall be applicable to balanced data link connections. A balanced data link connection shall involve two participating LLCs. For control purposes, each LLC shall assume responsibility for the organization of its data flow and for the link recovery operations for the transmissions that it originates. Each LLC shall be capable of sending and receiving both command and response PDUs.

For the transfer of data between LLCs in Type 2 operation, figure 7 depicts the data link control functions utilized. The data source in each LLC shall control the data sink in the other LLC by the use of command PDUs. The information shall flow from the data source to the data sink, and any acknowledgments shall always be sent in the opposite direction. The poll-type command PDUs shall be utilized by each LLC station to solicit specific acknowledgments and status responses from the other LLC.

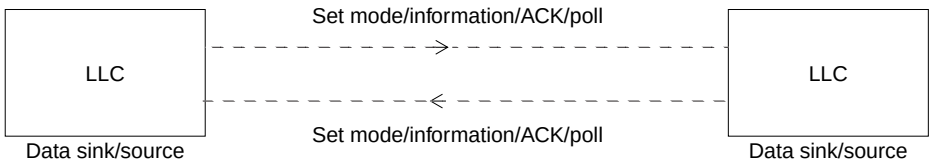


Figure 7—Balanced data link connection configuration

In normal Type 3 operation, each command PDU shall receive an acknowledgment PDU, and though the source LLC may retransmit a command PDU for recovery purposes, it will not send a new PDU from a given SSAP to an LLC with a given DA portion of the destination address at a given priority while waiting for an acknowledgment of a previous PDU with the same addresses and priority. The LLC entity will not

accept a new request primitive from the network layer until the receipt of the preceding “request” primitive LSDU has been acknowledged by the remote LLC entity. This restriction is necessary to allow higher layers to perform recovery operations before resuming normal data transmission in case LLC is unsuccessful in transferring a PDU (after retries).

A mechanism of alternating code-points in successive PDUs in Type 3 operation provides a one-bit sequence number functionality that allows the LLC receiving a command PDU to differentiate between a new PDU and a second copy of a previously received PDU. Further, the LLC receiving an acknowledgment PDU can ensure that the acknowledgment refers to the last sent information PDU. A previous acknowledgment that incurred excessive delay is thus ignored.

Depending on the service provided by the medium access control sublayer, the Type 3 operation specifies different amounts of state information that must be maintained at the stations involved in the information exchange. In the general case, each station must maintain for each SSAP-DA pair at each priority, a one-bit sequence number for sending and another for receiving. If, however, the medium access control sublayer can insure that during the period in which a first transmission and all retires of a single PDU occur, the destination station will not receive some other intervening Type 3 PDU, and the medium access control sublayer services its queues for each priority supported in FIFO (first in, first out) order within that priority, then only one sequence number is needed for received PDUs.

4.2 Classes of LLC (conformance clause)

Four classes of LLCs are defined. A Class I LLC shall support only Type 1 operation, a Class II LLC shall support only Type 1 and Type 2 operations, a Class III LLC shall support only Type 1 and Type 3 operations, and a Class IV LLC shall support Type 1, Type 2, and Type 3 operations, as illustrated in figure 8.

Class of LLC

Types of operation supported	I	II	III	IV
1	X	X	X	X
2		X		X
3			X	X

Xs indicate valid combinations

Figure 8—Classes of LLC

This means all LLCs on a local area network shall have in common at least one type of operation, namely Type 1. Furthermore, the support of Type 1 operation by a Class II LLC, a Class III LLC, or a Class IV LLC shall be totally independent of the modes or change of modes of the Type 2 or Type 3 operations in that same LLC. A Class II LLC, a Class III LLC, or a Class IV LLC shall be capable of going back and forth between Type 1 operation and Type 2 operation or Type 3 operation on a PDU-to-PDU basis, if necessary.

4.2.1 Class I LLC

Class I LLCs shall support Type 1 operation only. Class I service shall be applicable to individual, group, global, and null DSAP addressing, and applications requiring no data link layer acknowledgment or flow control procedures. The set of command PDUs and response PDUs supported in Class I service shall be

	Commands	Responses
Type 1:	UI	
	XID	XID
	TEST	TEST

4.2.2 Class II LLC

Class II LLCs shall support both Type 1 operation and Type 2 operation. In a Class II station, the operation of the Type 1 procedures and the Type 2 procedures are completely independent. The set of command PDUs and response PDUs supported in Class II service shall be

	Commands	Responses
Type 1:	UI	
	XID	XID
	TEST	TEST
Type 2:	I	I
	RR	RR
	RNR	RNR
	REJ	REJ
	SABME	UA
	DISC	DM
		FRMR

4.2.3 Class III LLC

Class III LLCs shall support both Type 1 and Type 3 operations. In a Class III station, the operation of the Type 1 procedures and the Type 3 procedures are completely independent. The set of command PDUs and response PDUs supported in Class III service shall be

	Commands	Responses
Type 1:	UI	
	XID	XID
	TEST	TEST
Type 3:	AC0	AC0
	AC1	AC1

4.2.4 Class IV LLC

Class IV LLCs shall support Type 1, Type 2, and Type 3 operations. In a Class IV station, the operation of the Type 1 procedures, the Type 2 procedures, and the Type 3 procedures are completely independent of one another. The set of command PDUs and response PDUs supported in Class IV service shall be

	Commands	Responses
Type 1:	UI	
	XID	XID
	TEST	TEST
Type 2:	I	I
	RR	RR
	RNR	RNR
	REJ	REJ
	SABME	UA
	DISC	DM
		FRMR
Type 3:	AC0	AC0
	AC1	AC1

4.3 Support of route determination entity (RDE) (conformance clause)

Support of RDE is optional. Support of RDE shall be totally independent of the modes or change of modes of operation for any type (Type 1, Type 2, etc.) operation. An LLC that supports RDE shall implement the RDE components consistent with clause 9, and

1) Shall respond to the route query command (RQC) with a route query response (RQR).
2) Shall exclude routes that do not meet the specified minimum criteria.
3) Shall default to the spanning tree route for
   a) A PDU addressed to the MAC broadcast address.
   b) A PDU addressed to a MAC group address.
   c) Any other case when no route meets the specified minimum criteria.

5. LLC elements of procedure

5.1 General

This clause specifies the elements of the LAN LLC procedures for code-independent data communication using the LLC PDU structure (see clause 3).

These LLC elements of procedure are defined specifically in terms of the actions that shall occur in the LLC on receipt of commands, and occasionally on receipt of a reply to a command, over a logical data link (Type 1 and Type 3) and a data link connection (Type 2). Each element of procedure is utilized by only one of the three types of operation (Type 1, Type 2, or Type 3) that are defined in clause 4.

5.2 Control field formats

The three formats defined for the control field (figure 9) shall be used to perform numbered information transfer, numbered supervisory transfer, unnumbered control, and unnumbered information transfer functions. The numbered information transfer and supervisory transfer functions apply only to Type 2 operation. The unnumbered control and unnumbered information transfer functions apply to Type 1, Type 2, or Type 3 operation depending upon the specific function selected.

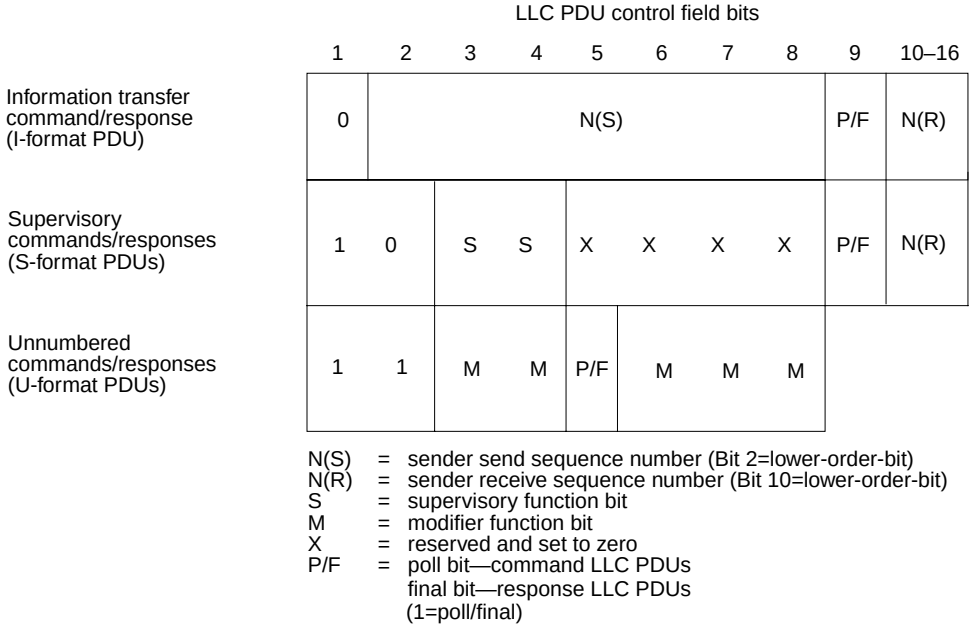


Figure 9—LLC PDU control field formats

5.2.1 Information transfer format—I

The I-format PDU shall be used to perform a numbered information transfer in Type 2 operation. Except where otherwise specified (e.g., UI, TEST, FRMR, and XID), it shall be the only LLC PDU that may contain an information field. The functions of N(S), N(R), and P/F shall be independent; i.e., each I-format PDU shall have an N(S) sequence number, an N(R) sequence number that shall or shall not acknowledge additional I-format PDUs at the receiving LLC, and a P/F bit that shall be set to “1” or “0”.

5.2.2 Supervisory format—S

The S-format PDU shall be used to perform data link supervisory control functions in Type 2 operation, such as acknowledging I-format PDUs, requesting retransmission of I-format PDUs, and requesting a temporary suspension of transmission of I-format PDUs. The functions of N(R) and P/F shall be independent; i.e., each S-format PDU shall have an N(R) sequence number that shall or shall not acknowledge additional I-format PDUs at the receiving LLC, and a P/F bit that shall be set to “1” or “0”.

5.2.3 Unnumbered format—U

The U-format PDUs shall be used in Type 1, Type 2, or Type 3 operation to provide additional data link control functions and to provide unsequenced information transfer. The U-format PDUs shall contain no sequence numbers, but shall include a P/F bit that shall be set to “1” or “0”.

5.3 Control field parameters

The various parameters associated with the control field formats are described in the following subclauses.

5.3.1 Type 1 operation parameters

The only parameter that exists in Type 1 operation is the Poll/Final (P/F) bit. The P/F bit set to “1” shall only be used in Type 1 operation with the XID and TEST command/response PDU functions. The Poll (P) bit set to “1” shall be used to solicit (poll) a correspondent response PDU with the F bit set to “1” from the addressed LLC. The Final (F) bit set to “1” shall only be used to indicate that response PDU that is sent by the LLC as the result of a soliciting (poll) command PDU (P bit set to “1”).

5.3.2 Type 2 operation parameters

The various parameters associated with the control field formats in Type 2 operation are described in the following subclauses.

5.3.2.1 Modulus

Each I PDU shall be sequentially numbered with a number that shall have a value between 0 and MODULUS minus ONE (where MODULUS is the modulus of the sequence numbers). The modulus shall equal 128 for the Type 2 LLC control field format. The sequence numbers shall cycle through the entire range.

The maximum number of sequentially numbered I PDUs that may be outstanding (i.e., unacknowledged) in a given direction on a data link connection at any given time shall never exceed one less than the modulus of the sequence numbers. This restrictions shall prevent any ambiguity in the association of sent I PDUs with sequence numbers during normal operation and/or error recovery action.

5.3.2.2 LLC PDU state variables and sequence numbers

A station LLC shall maintain a send state variable V(S) for the I PDUs it sends and a receive state variable V(R) for the I PDUs it receives on each data link connection. The operation of V(S) shall be independent of the operation of V(R).

5.3.2.2.1 Send state variable V(S)

The send state variable shall denote the sequence number of the next in-sequence I PDU to be sent on a specific data link connection. The send state variable shall take on a value between 0 and MODULUS minus ONE (where MODULUS equals 128 and the numbers cycle through the entire range). The value of the send state variable shall be incremented by one with each successive I PDU transfer on the associated data link connection, but shall not exceed N(R) of the last received PDU by more than MODULUS minus ONE.

5.3.2.2.2 Send sequence number N(S)

Only I PDUs shall contain N(S), the send sequence number of the sent PDU. Prior to sending an I PDU, the value of N(S) shall be set equal to the value of the send state variable for that data link connection.

5.3.2.2.3 Receive state variable V(R)

The receive state variable shall denote the sequence number of the next in-sequence I PDU to be received on a specific data link connection. The receive state variable shall take on a value between 0 and MODULUS minus ONE (where MODULUS equals 128 and the numbers cycle through the entire range). The value of the receive state variable associated with a specific data link connection shall be incremented by one whenever an error-free, in-sequence I PDU is received whose send sequence number N(S) equals the value of the receive state variable for the data link connection.

5.3.2.2.4 Receive sequence number N(R)

All I-format PDUs and S-format PDUs shall contain N(R), the expected sequence number of the next received I PDU on the specified data link connection. Prior to sending an I-format PDU or S-format PDU, the value of N(R) shall be set equal to the current value of the associated receive state variable for that data link connection. N(R) shall indicate that the station sending the N(R) has received correctly all I PDUs numbered up through N(R)-1 on the specified data link connection.

5.3.2.3 Poll/Final (P/F) bit

The poll (P) bit shall be used to solicit (poll) a response from the addressed LLC. The final (F) bit shall be used to indicate the response PDU sent as the result of a soliciting (poll) command.

The poll/final (P/F) bit shall serve a function in Type 2 operation in both command PDUs and response PDUs. In command PDUs the P/F bit shall be referred to as the P bit. In response PDUs it shall be referred to as the F bit. P/F bit exchange provides a distinct command/response linkage that is useful during both normal operation and recovery situations.

5.3.2.3.1 Poll bit functions

A command PDU with the P bit set to “1” shall be used on a data link connection to solicit a response PDU with the F bit set to “1” from the addressed LLC on that data link connection.

Only one PDU with a P bit set to “1” shall be outstanding in a given direction at a given time on the data link connection between any specific pair of LLCs. Before an LLC issues another PDU on the same link connection with the P bit set to “1”, the LLC shall have received a response PDU with the F bit set to “1” from the addressed LLC. If no valid response PDU is received within a system-defined P-bit timer time-out period, the (re)sending of a command PDU with the P bit set to “1” shall be permitted for error recovery purposes.

5.3.2.3.2 Final bit functions

A response PDU with the F bit set to “1” shall be used to acknowledge the receipt of a command PDU with the P bit set to “1”.

Following the receipt of a command PDU with the P bit set to “1”, the LLC shall send a response PDU with the F bit set to “1” on the appropriate data link connection at the earliest possible opportunity.

The LLC shall be permitted to send appropriate response PDUs with the F bit set to “0” at any medium access opportunity on an asynchronous basis (without the need for a command PDU).

5.3.3 Type 3 operation parameters

The various parameters associated with the control field formats in Type 3 operation are described in the following subclauses.

5.3.3.1 LLC PDU state variables

5.3.3.1.1 Transmit sequence state variable V(SI)

The LLC shall be able to maintain one transmit sequence state variable V(SI) for each unique combination of DA portion of the remote address, SSAP portion of the local address, and MAC priority used for sending Type 3 command PDUs. This variable shall only take on the values of zero and one and shall be set equal to bit eight of the code-point used for the last Type 3 response PDU received with the associated address pair and priority. The V(SI) variable permits the LLC to insure that a received acknowledgment applies to the currently outstanding transmission and allows the receiver to detect duplicate frames. V(SI) shall be created as specified in 8.5.1 and destroyed as specified in 8.4.2.

5.3.3.1.2 Receive sequence state variable V(RI)

The LLC shall be able to maintain one receive sequence state variables V(RI) for each unique combination of remote address (SA and SSAP), and MAC priority associated with received Type 3 command PDUs. This variable contains the complement of bit eight of the code-point used in the last received Type 3 command with the associated addresses at the associated priority. V(RI) allows the LLC to differentiate between a Type 3 command PDU received for the first time, and a received PDU that is a retransmission of a previously received PDU. V(RI) shall be created as specified in 8.5.2 and destroyed as specified in 8.4.3.

5.3.3.1.3 Reception status state variable V(RB)

The LLC shall be able to maintain one reception status state variable V(RB) for each unique combination of remote address (SA and SSAP), and MAC priority associated with received Type 3 command PDUs. This variable contains an indication of the success or failure of the reception of the information field from the last received Type 3 command with the associated address at the associated priority. V(RB) insures that the response to a duplicate command PDU contains the same reception status code as the response to the original command PDU.

5.3.3.2 Poll/Final (P/F) bit

The poll (P) bit shall be used to solicit a response PDU having an information field that contains both a status subfield and an LSDU subfield. The final (F) bit shall be used to indicate the response PDU sent as a result of a soliciting (poll) command.

5.3.3.2.1 Poll bit function

A command PDU with the P bit set to “0” shall be used on a data link to pass an LSDU (link service data unit) from the sending LLC to a receiving LLC. The command PDU with the P bit of “0” also requests the receiving LLC to return a response PDU having only a status subfield in its information field. A command PDU with the P bit set to “1” is used by the sending LLC to request the receiving LLC to return a response PDU having both a status subfield and an LSDU subfield in its information field.

5.3.3.2.2 Final bit function

A response PDU is always sent with the F bit set equal to the P bit in the command PDU for which the response is sent.

5.4 Commands and responses

This subclause defines the commands and associated responses. Subclauses 5.4.1, 5.4.2 and 5.4.3 contain the definitions of the set of commands and responses (listed below) for each of the control field formats for Type 1 operation, Type 2 operation, and Type 3 operation, respectively. The C/R bit, located in the low-order bit of the SSAP field, is used to distinguish between commands and responses. The following discussion of commands and responses assumes that the C/R bit has been properly decoded.

Information transfer format commands		**Information transfer format responses**	
I	Information	I	Information
Supervisory format commands		**Supervisory format responses**	
RR	Receive ready	RR	Receive ready
RNR	Receive not ready	RNR	Receive not ready
REJ	Reject	REJ	Reject
Unnumbered format commands		**Unnumbered format responses**	
UI	Unnumbered information	UA	Unnumbered Acknowledgment
DISC	Disconnect		
SABME	Set asynchronous balanced mode extended	DM	Disconnected Mode
		FRMR	Frame reject
XID	Exchange identification	XID	Exchange identification
TEST	Test	TEST	Test
AC0	Acknowledged connectionless information, Seq. 0	AC0	Acknowledged connectionless acknowledgment, Seq. 0
AC1	Acknowledged connectionless information, Seq. 1	AC1	Acknowledged connectionless acknowledgment, Seq. 1

NOTE—For brevity in the remaining text, the term “ACn” refers to both AC0 and AC1.

5.4.1 Type 1 operation commands and responses

The Type 1 commands and responses are all U-format PDUs.

5.4.1.1 Type 1 operation commands

The U-format PDU command encodings for Type 1 operation are listed in figure 10.

5.4.1.1.1 Unnumbered information (UI) command

The UI command PDU shall be used to send information to one or more LLCs. Use of the UI command PDU is not dependent on the existence of a data link connection between the destination and source LLCs, and its use will not affect the V(S) or V(R) variables associated with any data link connections. There is no LLC response PDU to the UI command PDU.

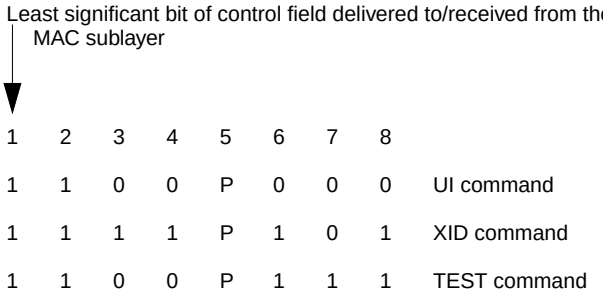


Figure 10—Type 1 operation command control field bit assignments

Reception of the UI command PDU is not acknowledged or sequence number verified; therefore, the data contained in a UI PDU may be lost if a logical data link exception (such as a transmission error or a receiver-busy condition) occurs during the sending of the UI command PDU.

A UI command PDU shall have either an individual, group, global, or null address as the destination DSAP address and the originator's individual address as the SSAP address.

5.4.1.1.2 Exchange identification (XID) command

The XID command PDU shall be used to convey the types of LLC services supported (for all LLC services) and the receive window size on a per data link connection basis to the destination LLC, and to cause the destination LLC to respond with the XID response PDU (see 5.4.1.2.1) at the earliest opportunity. The XID command PDU shall have no effect on any mode or sequence numbers maintained by the remote LLC. An XID command PDU shall have either an individual, group, global, or null address as the destination DSAP address and a null address or the originator's individual address as the SSAP address.

The information field of an XID LLC basic format command PDU shall consist of an 8-bit XID format identifier field plus a 16-bit parameter field, as shown in figure 11. The parameter field is encoded to identify the LLC services supported plus the XID sender's receive window size (RW).

The XID receiver should set its transmit window, k, to a value less than or equal to the receive window of the XID sender, to avoid overrunning the XID sender.

NOTE—Other uses of the XID PDU are for further study. In particular, the use of an unsolicited XID response PDU to announce the presence of a new LLC will be examined.

5.4.1.1.3 Test (TEST) command

The TEST command PDU shall be used to cause the destination LLC to respond with the TEST response PDU (see 5.4.1.2.2) at the earliest opportunity, thus performing a basic test of the LLC to LLC transmission path. An information field is optional with the TEST command PDU. If present, however, the received information field shall be returned, if possible, by the addressed LLC in the TEST response PDU. The TEST command PDU shall have no effect on any mode or sequence numbers maintained by the remote LLC and may be used with an individual, group, global, or null DSAP address with an individual or null SSAP address, and with an individual group, or global DA address.

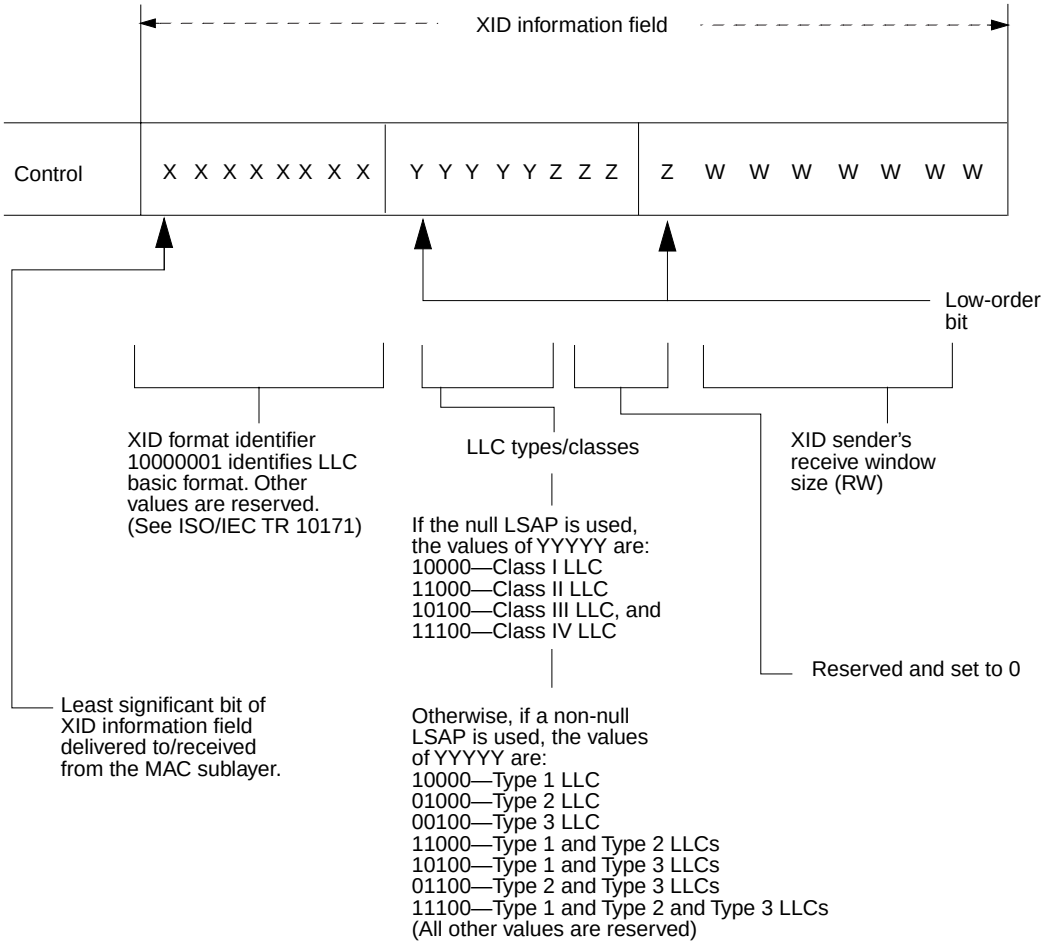


Figure 11—XID information field basic format

5.4.1.2 Type 1 operation responses

The U-format PDU response encodings for Type 1 operation are listed in figure 12.

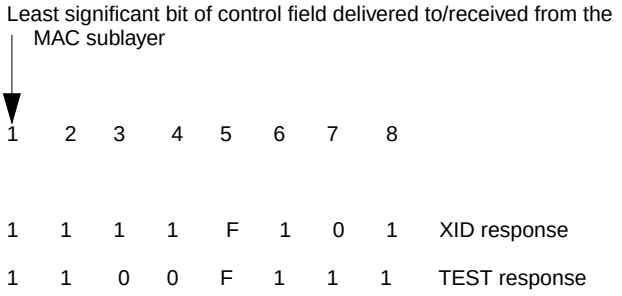


Figure 12—Type 1 operation response control field bit assignments

5.4.1.2.1 Exchange identification (XID) response

The XID response PDU shall be used to reply to an XID command PDU at the earliest opportunity. The XID response PDU shall identify the responding LLC and shall include an information field like that defined for the XID command PDU (see 5.4.1.1.2), regardless of what information is present in the information field of the received XID command PDU. The XID response PDU shall use an individual or null DSAP address and shall use an individual or null SSAP address. The XID response PDU shall have its F bit set to the state of the P bit in the XID command PDU.

5.4.1.2.2 Test (TEST) response

The TEST response PDU shall be used to reply to the TEST command PDU. The TEST response PDU shall have its F bit set to the value of the P bit in the TEST command PDU. An information field, if present in the TEST command PDU, shall be returned in the corresponding TEST response PDU. If the LLC cannot accept an information field (e.g., buffering limitation), a TEST response PDU without an information field may be returned. Refer to 6.7 for specific details on TEST response usage. The LLC sends a TEST response PDU, using the SSAP address of the TEST command PDU as the DSAP address of the TEST response PDU, and with an individual or null SSAP address selected by the DSAP address of the TEST command PDU.

5.4.2 Type 2 operation commands and responses

Type 2 commands and responses consist of I-format, S-format, and U-format PDUs.

5.4.2.1 Information transfer format command and response

The function of the information, I, command and response shall be to transfer sequentially numbered PDUs containing an octet-oriented information field across a data link connection. The encoding of the I PDU control field for Type 2 operation shall be as listed in figure 13.

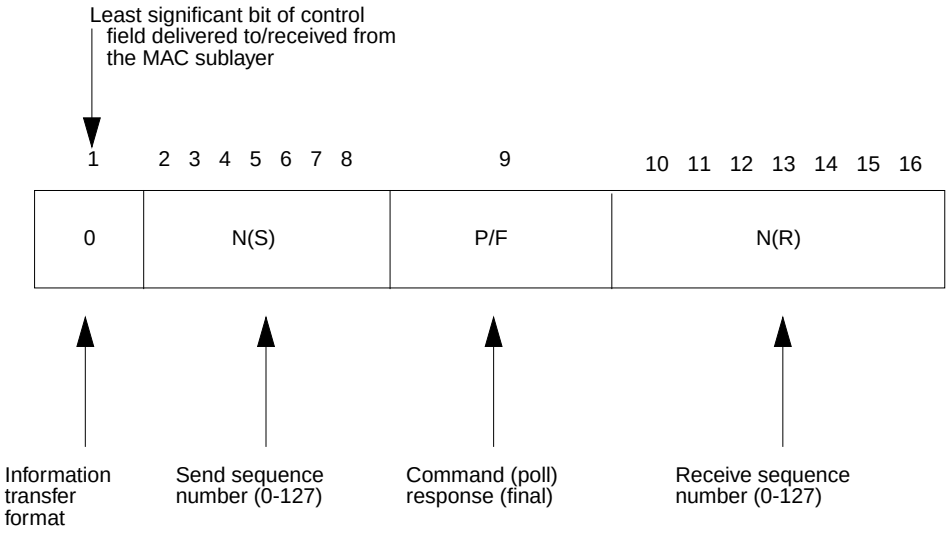


Figure 13—Information transfer format control field bits

The I PDU control field shall contain two sequence numbers: N(S), send sequence number, which shall indicate the sequence number associated with the I PDU; and N(R), receive sequence number, which shall indicate the sequence number (as of the time the PDU is sent) of the next expected I PDU to be received, and consequently shall indicate that the I PDUs numbered up through N(R)-1 have been received correctly.

See 5.3.2.3 for a description of P/F bit functions.

5.4.2.2 Supervisory format commands and responses

Supervisory, S, PDUs shall be used to perform numbered supervisory functions such as acknowledgments, temporary suspension of information transfer, or error recovery.

PDUs with the S format shall not contain an information field and, therefore, shall not increment the send state variable at the sender or the receive state variable at the receiver. The encoding of the S-format PDU control field for Type 2 operation shall be as shown in figure 14.

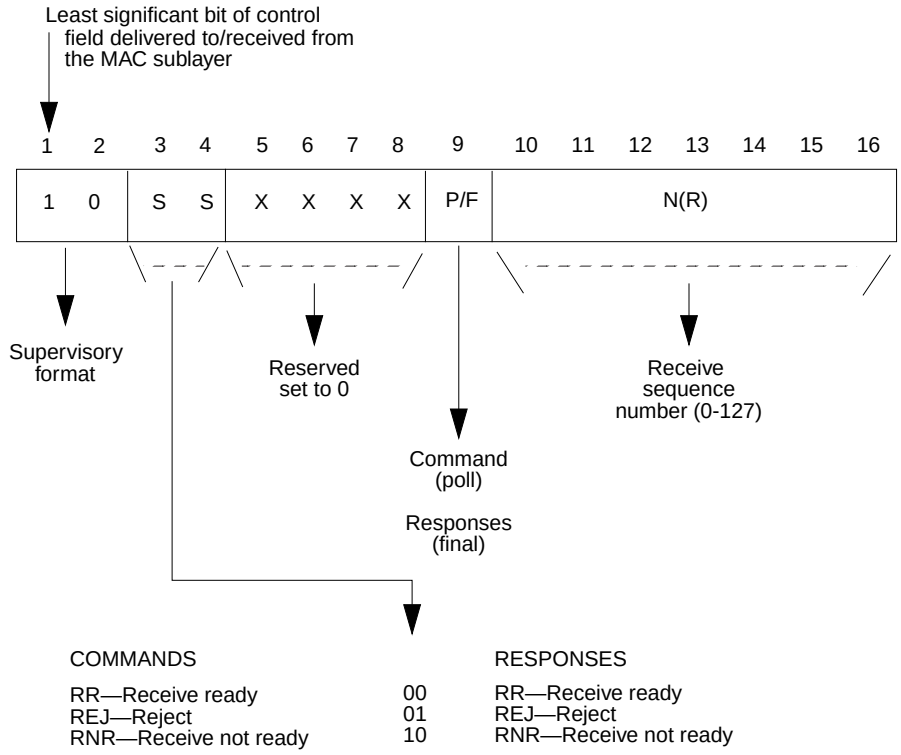


Figure 14—Supervisory format control field bits

An S-format PDU shall contain an N(R), receive sequence number, that shall indicate, at the time of sending, the sequence number of the next expected I PDU to be received, and consequently shall indicate that all received I PDUs numbered up through N(R)-1 have been received correctly.

When sent, an RR or REJ PDU shall indicate the clearance of any busy condition at the sending LLC that was indicated by the earlier sending of an RNR PDU.

See 5.3.2.3 for a description of the P/F bit functions.

5.4.2.2.1 Receive ready (RR) command and response

The RR PDU shall be used by an LLC to indicate it is ready to receive an I PDU(s). I PDUs numbered up through N(R)-1 shall be considered as acknowledged.

5.4.2.2.2 Reject (REJ) command and response

The REJ PDU shall be used by an LLC to request the resending of I PDUs starting within the PDU numbered N(R). I PDUs numbered up through N(R)-1 shall be considered as acknowledged. It shall be possible to send additional I PDUs awaiting initial sending after the resent I PDU(s).

With respect to each direction of sending on a data link connection, only one “sent REJ” condition shall be established at any given time. The “sent REJ” condition shall be cleared upon the receipt of an I PDU with an N(S) equal to the N(R) of the REJ PDU. The “sent REJ” condition may be reset in accordance with procedures described in 7.5.4.

5.4.2.2.3 Receive not ready (RNR) command and response

The RNR PDU shall be used by an LLC to indicate a busy condition (i.e., a temporary inability to accept subsequent I PDUs). I PDUs numbered up through N(R)-1 shall be considered as acknowledged. I PDUs numbered N(R) and any subsequent I PDUs received, if any, shall not be considered as acknowledged; the acceptance status of these PDUs shall be indicated in subsequent exchanges.

5.4.2.3 Unnumbered format commands and responses

Unnumbered, U, commands and responses shall be used in Type 2 operation to extend the number of data link connection control functions. PDUs sent with the U format shall not increment the state variables on the data link connection at either the sending or the receiving LLC. The encoding of the U-format command/response PDU control field shall be as shown in figure 15.

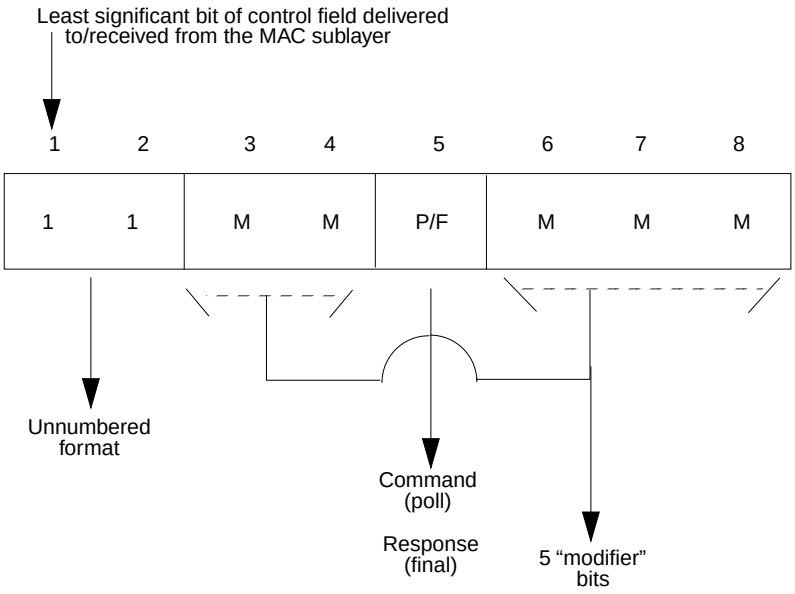


Figure 15—Unnumbered format control field bits

See 5.3.2.3 for a description of the P/F bit functions.

The U-format command and response encodings for Type 2 operation are listed in figure 16.

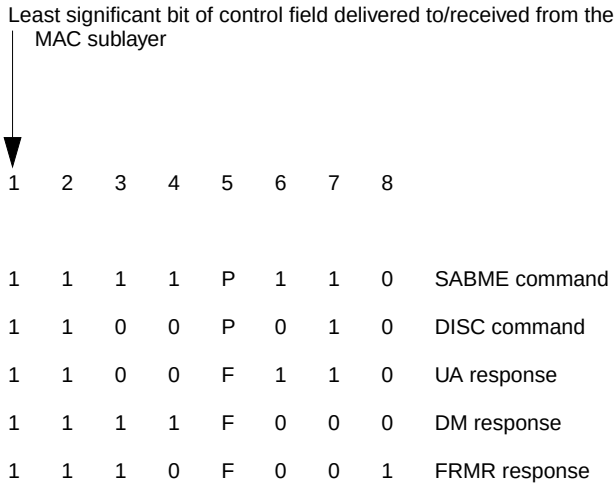


Figure 16—Unnumbered command and response control field bit assignments

5.4.2.3.1 Set asynchronous balanced mode extended (SABME) command

The SABME command PDU shall be used to establish a data link connection to the destination LLC in the asynchronous balanced mode. No information shall be permitted with the SABME command PDU. The destination LLC shall confirm receipt of the SABME command PDU by sending a UA response PDU or a DM response PDU on that data link connection at the earliest opportunity according to whether a DL-CONNECT response primitive or a DL-DISCONNECT request primitive is passed from the network layer to the LLC sublayer. Upon acceptance of the SABME command PDU, the destination LLCs send and receive state variables shall be set to zero. If the UA response PDU is received correctly, then the initiating LLC shall also assume the asynchronous balanced mode with its corresponding send and receive state variables set to zero.

Previously sent I PDUs that are unacknowledged when this command is actioned shall remain unacknowledged. Whether or not an LLC resends the contents of the information field of unacknowledged outstanding I PDUs shall be decided at a higher layer.

5.4.2.3.2 Disconnect (DISC) command

The DISC command PDU shall be used to terminate an asynchronous balanced mode previously set by a SABME command PDU. It shall be used to inform the destination LLC that the source LLC is suspending operation of the data link connection and the destination LLC should assume the logically disconnected mode. No information field shall be permitted with the DISC command PDU. Prior to actioning the command, the destination LLC shall confirm the acceptance of the DISC command PDU by sending a UA response PDU on that data link connection.

Previously sent I PDUs that are unacknowledged when this command is actioned shall remain unacknowledged. Whether or not an LLC resends the contents of the information field of unacknowledged outstanding I PDUs shall be decided at a higher layer.

5.4.2.3.3 Unnumbered acknowledgment (UA) response

The UA response PDU shall be used by an LLC on a data link connection to acknowledge the receipt and acceptance of the SABME and DISC command PDUs. These received command PDUs shall not be actioned until the UA response PDU is sent. No information field shall be permitted with the UA response PDU.

5.4.2.3.4 Disconnect mode (DM) response

The DM response PDU shall be used to report status indicating that the LLC is logically disconnected from the data link connection and is, by system definition, in ADM. No information field shall be permitted with the DM response PDU.

5.4.2.3.5 Frame reject (FRMR) response

The FRMR response PDU shall be used by the LLC in the asynchronous balanced mode to report the observance of a condition, that is not correctable by re-sending the identical PDU, that resulted from the receipt of a PDU from the remote LLC.

The FRMR response PDU shall be used in the case of the following conditions:

1) The receipt of a command PDU or a response PDU that is undefined or not implemented.
2) The receipt of a supervisory or unnumbered PDU with an information field that is not permitted.
3) The receipt of an I PDU, with an information field that exceeded the established maximum information field length that can be accommodated by the receiving LLC for that data link connection.
4) The receipt of an invalid N(R) from the remote LLC. [An invalid N(R) shall be defined as one that signifies an I PDU that has previously been sent and acknowledged, or signifies an I PDU that has not been sent and is not the next sequential I PDU awaiting to be sent.]
5) The receipt of an invalid N(S) from the remote LLC. [An invalid N(S) shall be defined as an N(S) that is greater than or equal to the last sent N(R)+RW, where RW is the receive window size of the local LLC, unless the N(S) is recognized as indicating a duplicate PDU as defined in 7.5.10.]

The FRMR response PDU may be used in the case of any or both of the following conditions:

— The receipt of an unsolicited F bit set to “1”.
— The receipt of an unexpected UA response PDU.

The responding LLC shall send the FRMR response PDU at the earliest opportunity.

The LLC receiving the FRMR response PDU shall pass a DL-RESET indication primitive to the network layer, which is responsible itself for initiating the appropriate mode setting or resetting corrective action by instructing the LLC to initialize both directions of information transfer on the data link connection, using the SABME and DISC command PDUs, as applicable.

An information field shall be returned with the FRMR response PDU to provide the reason for the PDU rejection. The information field shall contain the fields shown in figure 17.

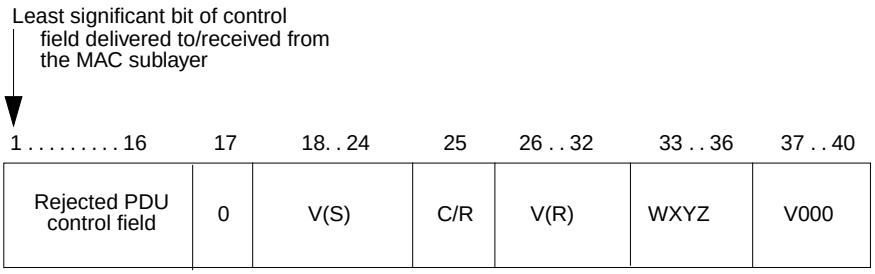
5.4.3 Type 3 operation commands and responses

The Type 3 commands and responses are U-format PDUs, respectively, whose encodings are listed in figure 18.

See 5.3.3.2 for a description of the P/F bit functions.

5.4.3.1 Acknowledged connectionless information (ACn) command

In Type 3 operation the ACn command PDU shall be used to send information or to request information, without the prior establishment of a data link connection. Use of the ACn command PDU is not dependent upon the existence of a data link connection between the destination and source LLCs, and its use will not affect the V(S) or V(R) variables associated with any existing data link connections. Reception of an ACn command PDU shall be acknowledged by an ACn response PDU (see 5.4.3.2) at the earliest opportunity. The



1) Rejected PDU control field shall be the control field of the received PDU that caused the FRMR exception condition on the data link connection. When the rejected PDU is a U-format PDU, the control field of the rejected PDU shall be positioned in bit positions 1-8, with 9-16 set to 0.
2) V(S) shall be the current send state variable value for this data link connection at the rejecting LLC (bit 18 = low-order bit).
3) C/R set to “1” shall indicate that the PDU which caused the FRMR was a response PDU, and C/R set to “0” shall indicate that the PDU which caused the FRMR was a command PDU.
4) V(R) shall be the current receive state variable value for this data link connection at the rejecting LLC (bit 26 = low-order bit).
5) W set to “1” shall indicate that the control field received and returned in bits 1 through 16 was invalid or not implemented. Examples of invalid PDU are defined as
   a) The receipt of a command PDU or a response PDU that is undefined or not implemented;
   b) The receipt of a supervisory or unnumbered PDU with an information field that is not permitted;
   c) The receipt of an unsolicited F bit set to “1”; and
   d) The receipt of an unexpected UA response PDU.
6) X set to “1” shall indicate that the control field received and returned in bits 1 through 16 was considered invalid because the PDU contained an information field that is not permitted with this command or response. Bit W shall be set to “1” in conjunction with this bit.
7) Y set to “1” shall indicate that the information field received exceeded the established maximum information field length that can be accommodated by the rejecting LLC on that data link connection.
8) Z set to “1” shall indicate that the control field received and returned in bits 1 through 16 contained an invalid N(R).
9) V set to “1” shall indicate that the control field received and returned in bits 1 through 16 contained an invalid N(S). Bit W shall be set to “1” in conjunction with this bit.

Figure 17—FRMR information field format

ACn command PDU shall have no effect on any Type 2 operation or sequence numbers maintained by the remote LLC. The ACn command shall have an individual, group or global address as the DSAP address and the originator’s individual address as the SSAP address. The information field in the ACn command PDU may be either null (having zero length) or non-null, and if non-null, shall contain a link service data unit.

NOTE—The use of the ACn command with a group or global remote address is a subject for further study and thus should be done carefully to avoid possible failure of the protocol.

5.4.3.2 Acknowledged connectionless acknowledgment (ACn) response

In Type 3 operation the ACn response PDU shall be used to reply to an ACn command PDU. Responses shall be made at the earliest opportunity. The ACn response PDU shall identify the responding LLC and be sent to the originating LLC. The ACn response PDU shall always contain a status subfield in its information field. The remainder of the information field may be either null or non-null, and if non-null shall contain a link service data unit as shown in figure 19.

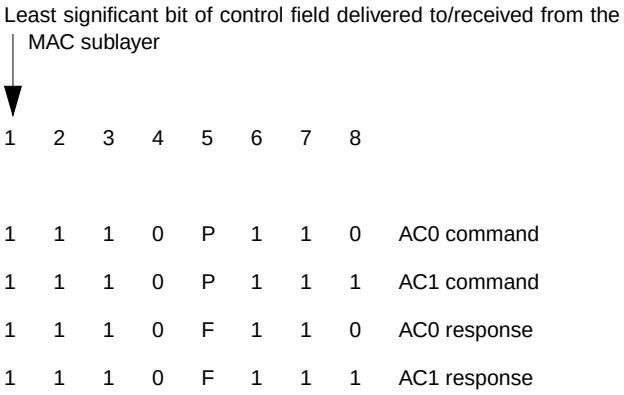


Figure 18—Type 3 operation command and response control field bit assignments

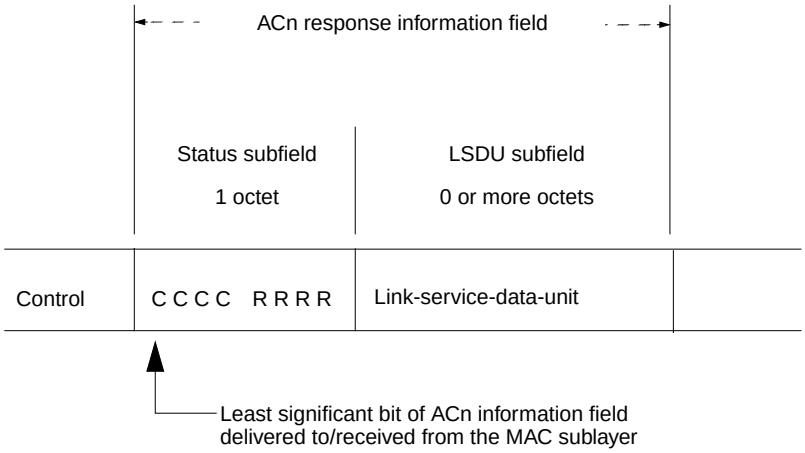


Figure 19—ACn response information field format

The code returned in the CCCC part of the status subfield indicates the success or failure of information transfer in the command PDU (from the initiating LLC to the responding LLC). The possible values of CCCC are given in figure 20.

The code returned in the RRRR part of the status subfield indicates the success or failure of information transfer in the response PDU (from the responding LLC to the initiating LLC). The possible values of RRRR are given in figure 21.

A returned status code categorized as “Success” indicates that the requested action was successfully performed. The category of “Temp Err” indicates that a temporary and self-correcting condition at the responding LLC prevented the performance of the requested action, and retrying the request is likely to result in success. The category of “Perm Err” indicates that a programming or hardware error prevented the performance of the requested action, and retrying the request is not likely to result in success.

The CCCC status codes indicating error conditions are to be used as follows. The “RS” code indicates that either the responding LLC or the specified LSAP does not support the reception of data in the command PDU. Note the support is optional for the reception of an LSDU in a command PDU with a P bit of one, and

Least significant bit of CCCC subfield delivered to/received from the MAC sublayer

CCCC	Mnemonic	Category	Description
0000	OK	Success	Command accepted
1000	RS	Perm Err	Unimplemented or inactivated service
1010	UE	Perm Err	LLC user interface error
0110	PE	Perm Err	Protocol error
1110	IP	Perm Err	Permanent implementation-dependent error
1001	UN	Temp Err	Resources temporarily unavailable
1111	IT	Temp Err	Temporary implementation-dependent error

All other CCCC codes are reserved.

Figure 20—ACn response status subfield CCCC values

Least significant bit of RRRR subfield delivered to/received from the MAC sublayer

RRRR	Mnemonic	Category	Description
0000	OK	Success	Response LSDU present
1000	RS	Perm Err	Unimplemented or inactivated service
1100	NE	Perm Err	Response LSDU never submitted
0010	NR	Success	Response LSDU not requested
1010	UE	Perm Err	LLC user interface error
1110	IP	Perm Err	Permanent implementation-dependent error
1001	UN	Temp Err	Resources temporarily unavailable
1111	IT	Temp Err	Temporary implementation-dependent error

All other RRRR codes are reserved.

Figure 21—ACn response status subfield RRRR values

the “RS” status code should be used to indicate that such data was sent to a non-supporting LLC. The “UE” status code indicates that, due to a failure at the interface between the responding LLC and its user, the responding LLC is not able to pass the received data to its user. The “PE” status code indicates that the received ACn command PDU violates the requirements of the standard; for example, the command PDU had a service class of “response.” The “UN” status code indicates that resources to receive the LSDU in the command PDU were temporarily unavailable. The “IT” and “IP” status codes may be used by the implementor to indicate a temporary or permanent error condition respectively, in the case that a specific standard code does not exist for the error.

The RRRR status codes indicating error conditions are to be used as follows. The “RS” code indicates that either the responding LLC or the specified DSAP does not support the inclusion of data in the response PDU. The “NE” status code indicates that the responding LLC has never (since powerup) received a DL-REPLY-UPDATE request primitive to associate a reply LSDU with the specified LSAP. The “UE” status code indicates that, due to a failure at the interface between the responding LLC and its user, the responding LLC is unable to obtain the required response data. The “UN” status code indicates that resources to include the requested LSDU in the response PDU were temporarily unavailable (e.g., a shared data area is being

updated). The “IT” and “IP” status codes may be used by the implementor to indicate a temporary or permanent error condition, respectively, in the case that a specific standard code does not exist for the error.

5.4.3.3 Summary of Type 3 PDU functions

Figure 22 summarizes the functions performed by ACn command and response PDUs according to the state of the P/F bit and the presence of a non-null LSDU.

Commands:

P	LSDU	Function
0	null	Resynchronization
0	non-null	Sending data
1	null	Requesting data
1	non-null	Exchanging data

Responses:

F	LSDU	Function
0	null	Acknowledgment of resynchronization or acknowledgment of received data
0	non-null	(Not allowed)
1	null	Acknowledgment, requested data unavailable
1	non-null	Acknowledgment with requested data

Figure 22—Summary of ACn command and response functionality

6. LLC description of the Type 1 procedures

6.1 Mode of operation

In Type 1 operation, no modes of operation are defined. An LLC sublayer using Type 1 procedures shall support the entire procedure set whenever it is operational on the local area network.

6.2 Procedure for addressing

The address fields shall be used to indicate the source (SSAP) and destination (DSAP) of the LLC PDU. The least significant bit in the source address field (SSAP) shall be used to identify whether a command or a response is contained in the PDU. Individual, group, global, and null addressing shall be supported for destination DSAP addresses. The source address field (SSAP) shall contain either an individual or null source address (see 3.3.1.2).

6.3 Procedure for the use of the P/F bit

A UI command PDU shall only be sent with the P bit set to “0”. If a UI command PDU is received with the P bit set to “1”, the LLC sublayer shall optionally discard it or pass it to the higher layer with a flag identifying that the P bit was set to “1”. Since a UI PDU shall not be sent as a response PDU, procedure regarding the use of the F bit do not apply.

An XID command PDU shall have the P bit set to either “0” or “1”. Upon receipt of an XID command PDU, the receiving LLC sublayer shall return an XID response PDU that has the F bit set equal to the value of the P bit contained in the incoming command PDU.

A TEST command PDU shall have the P bit set to either “0” or “1”. Upon receipt of a TEST command PDU, the receiving LLC sublayer shall return a TEST response PDU that has the F bit set equal to the value of the P bit contained in the incoming command PDU.

6.4 Procedures for logical data link setup and disconnection

Type 1 operation does not require any prior data link connection establishment (setup), and hence no data link disconnection. Once the service access point has been enabled within the LLC sublayer, presumably by layer management’s request, information may be sent to, or received from, a remote LLC sublayer SAP that is also participating in Type 1 operation.

6.5 Procedures for information transfer

6.5.1 Sending UI PDUs

Information transfer shall be accomplished by sending the UI command PDU with the P bit set to “0”. Sending UI PDUs with the P bit set to “1” or as response PDUs is prohibited. It shall be possible to send the UI command PDU at any time.

6.5.2 Receiving UI PDUs

Reception of the UI command PDU shall not be acknowledged or sequence number verified by the logical data link procedures; therefore, the UI PDU may be lost if a logical data link exception occurs during the sending of the command PDU. It shall be possible to receive a UI command PDU at any time. However, local conditions at the receiver may result in the discarding of valid UI command PDUs by the receiving

LLC. UI command PDUs that are received with the P bit set to “1” shall optionally be discarded or passed to the higher layer with a flag identifying that the P bit was set to “1”.

UI PDUs that are response PDUs are invalid transmissions and shall be discarded by the receiving LLC.

6.6 Uses of the XID command PDU and response PDU

While the response to an XID command PDU shall be mandatory, the origination of an XID command PDU shall be optional. It shall be possible for the XID capabilities to be used as a part of some network control functions. As such, an XID command PDU may be sent on direction from a higher layer function, an administration function having access to the data link layer, or an automatic start-up function. However, it shall also be possible for a more capable implementation of LLC to incorporate the use of the XID function directly to make more efficient use of the protocol.

Some possible uses of the XID capabilities include:

1) The XID command PDU with a null DSAP and a null SSAP is a way to solicit a response from any station (i.e., any DA). As such it represents a basic “Are You There?” test capability.
2) The XID command PDU with a group DA or group DSAP address can be used to determine the group membership. In particular, the XID command PDU with a global DA address can identify all active stations.
3) A duplicate address check can be made (see table 1).
4) For Class II LLCs in ABM, an XID exchange can be used to identify the receive window size at each LLC for that data link connection.

   NOTE—The use of an XID exchange for this purpose is not valid in the asynchronous disconnected mode (ADM) state.

5) An XID exchange with a null DSAP and a null SSAP can identify the class of each LLC.
6) An XID exchange with a specific DSAP and a specific SSAP can identify the service types supported by those service access points.
7) An LLC can announce its presence with a global DA address in an XID PDU.

6.7 Uses of the TEST command PDU and response PDU

The TEST function provides a facility to conduct loopback tests of the LLC to LLC transmission path. The initiation of the TEST function may be caused by an administration or management entity within the data link layer. Successful completion of the test consists of sending a TEST command PDU with a particular information field provided by this administration or management entity to the designated destination LLC address and receiving, in return, the identical information field in a TEST response PDU. Implementation of the TEST command PDU is optional but every LLC must be able to respond to a received TEST command PDU with a TEST response PDU. The length of the information field is variable from 0 to the largest size specified that each LLC on this local area network must support for normal data transfer.

It shall also be possible to send even larger information fields with the following interpretations. If the receiving LLC can successfully receive and return the larger information field, it will do so. If it cannot receive the entire information field but the MAC can detect a satisfactory FCS, the LLC shall discard the portion of the information field received, and may return a TEST response PDU with no information field. If the MAC cannot properly compute the FCS for the overlength information fields, the LLC shall discard the portion of the information field received, and shall give no response. Any TEST command PDU received in error shall be discarded and no response PDU sent. In the event of failure, it shall be the responsibility of the administration or management entity that initiated the TEST function to determine any future actions.

6.8 List of logical data link parameters

A number of logical data link parameters are defined, the range of values for which are determined on a system-by-system basis by the user at the time that the local area network is established.

The logical data link parameters for Type 1 operation shall be as follows:

6.8.1 Maximum number of octets in a UI PDU

Refer to the appropriate MAC protocol specification for any limitation on the maximum number of octets in a UI PDU. No restrictions are imposed by the LLC sublayer. However, in the interest of having a value that all users of Type 1 LLC may depend upon, all MACs must at least be capable of accommodating UI PDUs with information fields up to and including 128 octets in length.

6.8.2 Minimum number of octets in a PDU

The minimum length valid PDU shall contain exactly two service access point address fields and one control field in that order. Thus the minimum number of octets in a valid Type 1 PDU shall be 3.

6.9 Precise description of the Type 1 procedures

If discrepancies appear to exist with the text found in the balance of clause 6, this subclause (6.9) shall be viewed as being the definitive description.

6.9.1 LLC precise specification

The operation of the LLC is logically divided into several components. Each component characterizes a set of protocol operations performed by an LLC entity and is defined using a protocol state machine description. These state machines do not specify particular implementation techniques; rather, they are intended to describe the “external” characteristics of an LLC entity, as perceived by an LLC entity in a remote station or by a higher layer protocol in the local station.

6.9.1.1 LLC operation

The LLC operation is described using the following five types of components:

1) *Station Component.* This component is responsible for processing the events that affect the entire LLC entity. The station component handles PDUs addressed to the null DSAP address and processes the duplicate address check, if implemented. One station component shall exist for each MAC service access point present on the local area network.
2) *Link Service Access Point (SAP) Component.* This component is responsible for processing the events that affect a specific operating service access point. One SAP component shall exist for each SAP within the LLC entity.
3) *Connection Component.* This component is responsible for processing the events that affect a specific data link connection for Type 2 procedures only (see 7.9 below). One connection component shall exist for each data link connection supported in the LLC entity.
4) *Type 3 Receiver Component.* This component is responsible for processing Type 3 commands as they are received from the medium access control sublayer, and for returning Type 3 responses to the originators of the commands. At any given station there is potentially a separate Type 3 Receiver Component for each possible combination of remote address (SA and SSAP), and MAC priority. In most applications, however, only a subset of the possible Receiver Components will exist, since a component is needed only for those address and priority combinations for which Type 3 commands are actually received.

Table 1—Station component state transitions

Current state	Event	Action(s)	Next state
DOWN_STATE	[ENABLE_WITH_DUPLICATE_ ADDRESS_CHECK]	SEND_NULL_DSAP_XID_C START_ACK_TIMER RETRY_COUNT:=0 XID_R_COUNT:=0	DUPLICATE_ ADDRESS_ CHECK_STATE
	[ENABLE_WITHOUT_ DUPLICATE_ ADDRESS_CHECK]	REPORT_STATUS(STATION_UP)	UP_STATE
UP_STATE	DISABLE_REQUEST	REPORT_STATUS(STATION_DOWN)	DOWN_STATE
	RECEIVE_NULL_DSAP_XID_C	SEND_XID_R	UP_STATE
	RECEIVE_NULL_DSAP_TEST_C	SEND_TEST_R	UP_STATE
DUPLICATE_ ADDRESS_ CHECK_STATE (OPTIONAL)	[RECEIVE_NULL_DSAP_XID_ R_AND_XID_R_COUNT=0]	XID_R_COUNT:= XID_R_COUNT+1	DUPLICATE_ ADDRESS_ CHECK_STATE
	[RECEIVE_NULL_DSAP_XID_ R_AND_XID_R_COUNT=1]	REPORT_STATUS(DUPLICATE_ADDRESS_FOUND)	DOWN_STATE
	[RECEIVE_NULL_DSAP_XID_C]	SEND_XID_R	DUPLICATE_ ADDRESS_ CHECK_STATE
	[ACK_TIMER_EXPIRED_AND_ RETRY_COUNT<MAXIMUM_ RETRY]	SEND_NULL_DSAP_XID_C START_ACK_TIMER RETRY_COUNT:=RETRY_ COUNT+1 XID_R_COUNT:=0	DUPLICATE_ ADDRESS_ CHECK_STATE
	[ACK_TIMER_EXPIRED_AND_R ETRY_COUNT=MAXIMUM_ RETRY]	REPORT_STATUS(STATION_UP)	UP_STATE
	[DISABLE_REQUEST]	REPORT_STATUS(STATION_DOWN)	DOWN_STATE

5) *Type 3 Sender Component*. This component is responsible for sending Type 3 commands upon request of the network layer, and for the processing of Type 3 responses when they are received from the medium access control sublayer. At any given station there is potentially a separate Type 3 Sender Component for each possible combination of local LSAP, DA portion of the remote address, and MAC priority. In most applications, however, only a subset of the possible Sender Components will exist, since a component is needed only for those address and priority combinations for which Type 3 commands are actually sent.

6.9.1.2 Assumptions made

The operation of each component is described using a state machine description. These important points are assumed in these descriptions:

1) The components are hierarchically related; i.e., the station component is the “parent” of the SAP component, which in turn is the “parent” of the connection component. See figure 23.
2) Each “parent” component has a state that provides the enabling conditions for its “child” component(s) to operate. If the parent component leaves this state, then the “child” component(s) are disabled.
3) For each “parent” component, several “child” components are allowed to be concurrently operating once their “parent” enabling conditions have been satisfied.
4) There exists for each MAC service access point one and only one LLC entity, consisting of the various operating components.
5) In Class I LLC operation, each LLC can have zero or more SAPs being serviced (i.e., active) at any one time, independent of each other, which are differentiated by the DSAP address. The services of a SAP shall be provided by a separate service access point component.
6) In Class II LLC operation, the services of each SAP can also support zero or more concurrent data link connections (each designated by the logical concatenation of the MAC address (DA/SA) and the LLC address (DSAP/SSAP). Each data link connection is controlled by a separate connection component.

6.9.1.3 Component sections

Each component description shall consist of these sections:

1) *Component Overview.* This section shall discuss the overall purpose behind the component operation.
2) *Component State Transition Diagram.* This diagram shall graphically represent the component machine overview.
3) *Component State Transition Table.* This chart shall display a table of the state transitions, including columns for current state, event, action(s), and next state. This table shall define all valid events for each state as well as the resultant component action(s) and state change.
4) *Component Event Description.* Each of the events that are used in the state transition table is explained.
5) *Component Action Description.* Each of the actions that are used in the state transition table is explained.
6) *Component State Description.* Each of the states that are used in the state transition table are explained.

6.9.1.4 State machine operation rules

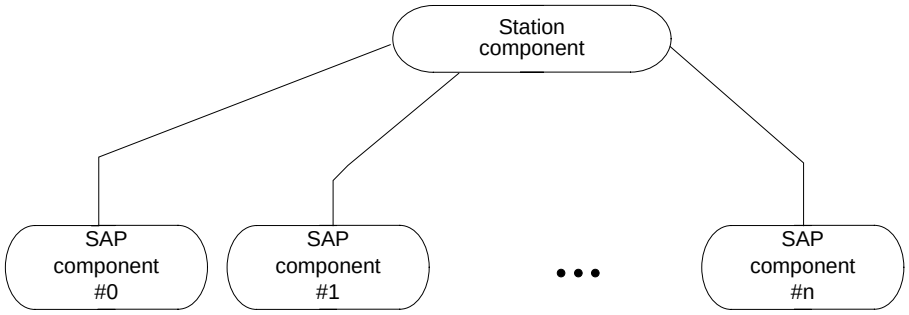
The following basic state machine operation rules apply:

1) Events shall cause a state transition in the machine, and shall result in execution of some action(s) along with a state change (which may return to the same state).
2) Events that are not listed as valid inputs to the current state of any of the operating components shall not cause
   a) actions or state changes; or
   b) PDU transmissions.

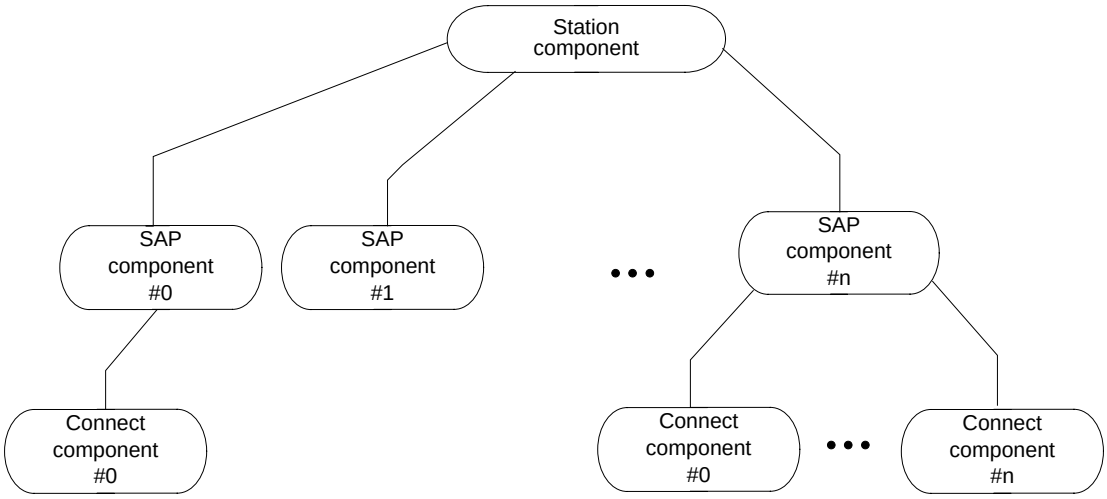
   The station should perform some error recovery that is appropriate for the particular implementation.
3) If an incoming PDU is destined for a DSAP that is not active (i.e., the appropriate component is not operating), it shall be considered to be an exception and dealt with in a manner appropriate for the receiving station.

6.9.2 Station component overview

The station component is responsible for handling all events that are directed to the LLC as a whole (i.e., events affecting all SAPs and connections serviced by that LLC). The station component shall begin in the



a) Class I LLC component relationship



b) Class II LLC component relationships

Figure 23—Component relationships

DOWN state, optionally check for a duplicate station address, and potentially enter the UP state; see figure 24 and table 1. The UP state of the LLC station component provides the enabling conditions for the operation of the service access point (SAP) components.

The station component shall be capable of receiving and responding to the XID and TEST command PDUs. It shall optionally be capable of initiating the XID command PDU, if duplicate address checking is performed by the LLC entity in a particular implementation; see table 2. These PDUs shall use the null DSAP address to denote that the station component is being referenced.

The performance of the duplicate address check requires that the station component be prepared to receive its own XID PDUs. The definition of the MAC operation provides for the ability to simultaneously transmit and receive. Since the DA=SA in the XID PDUs can be used for duplicate address check, the MAC will recognize its own address and pass the PDU to the station component. The station component will respond to an XID command PDU with an XID response PDU, regardless of whether it originated from itself or a remote LLC. The station component provides the duplicate address check by maintaining a count of received XID response PDUs. If more than one XID response PDU is received, then at least one other identical MAC DA exists on the LAN. See figure 24 and table 1 for details.

6.9.2.1 Station component state descriptions

1) **DOWN_STATE:** The station component is powered off, not initialized, and/or disabled from operating in the local area network.
2) **DUPLICATE_ADDRESS_CHECK_STATE:** The station component is in the process of checking for duplicate MAC addresses on the LAN. The main purpose of this state shall be to allow the LLC station component to verify that this station's MAC address is unique on the LAN. The station component shall send XID command PDUs with identical MAC DA and SA addresses, and shall wait for a possible XID Response PDU indicating the existence of other stations with identical MAC link addresses.
3) **UP_STATE:** The station component is enabled, powered on, initialized, and operating in the local area network. The LLC shall allow SAPs to exchange LLC PDUs on the medium.

6.9.2.2 Station component event descriptions

1) **ENABLE_WITH_DUPLICATE_ADDRESS_CHECK:** Station component user has initialized/enabled the station equipment, and has requested that the LLC check for MAC service access point address duplications before participating in data link communications.
2) **ENABLE_WITHOUT_DUPLICATE_ADDRESS_CHECK:** Station component user has initialized/enabled the station equipment, but duplicate MAC service access point address checking by the LLC is not supported/desired.
3) **ACK_TIMER_EXPIRED_AND_RETRY_COUNT<MAXIMUM_RETRY:** Acknowledgment timer has expired and retry count is less than maximum retry limit.
4) **ACK_TIMER_EXPIRED_AND_RETRY_COUNT=MAXIMUM_RETRY:** Acknowledgment timer has expired and retry count is equal to the maximum retry limit.
5) **RECEIVE_NULL_DSAP_XID_C:** An XID command PDU with the null DSAP address has been received.
6) **RECEIVE_NULL_DSAP_XID_R_AND_XID_R_COUNT=0:** A single XID response PDU with the null DSAP address has been received.
7) **RECEIVE_NULL_DSAP_XID_R_AND_XID_R_COUNT=1:** A second XID response PDU with the null DSAP address have been received.
8) **RECEIVE_NULL_DSAP_TEST_C:** A TEST command PDU with the null DSAP address has been received.
9) **DISABLE_REQUEST:** Station user has requested that the equipment be disabled from operating on the medium.

6.9.2.3 Station component action descriptions

1) **START_ACK_TIMER:** Start the acknowledgment timer. This allows the LLC to determine that it has not received an acknowledgment from the remote station within a specified response time.
2) **RETRY_COUNT:=0:** Initialize the retry counter.
3) **RETRY_COUNT:=RETRY_COUNT+1:** Increment the retry counter.
4) **XID_R_COUNT:=0:** Initialize the XID response PDU counter.
5) **XID_R_COUNT:=XID_R_COUNT+1:** Increment the XID response PDU counter.
6) **SEND_NULL_DSAP_XID_C:** The LLC shall send an XID command PDU with null SSAP and null DSAP addresses and with identical MAC DA and SA addresses.
7) **SEND_XID_R:** The LLC shall send an XID response PDU, using the SSAP address of the XID command PDU as the DSDAP address of the response PDU, and using a null SSAP address.
8) **SEND_TEST_R:** The LLC shall send a TEST response PDU, using the SSAP address of the TEST command PDU as the DSAP address of the response PDU, and using a null SSAP address.
9) **REPORT_STATUS:** The LLC shall be able to report data link status conditions, with the following valid reasons:
   a) **STATION_UP:** LLC entity is now operational
   b) **STATION_DOWN:** The LLC entity is now non-operational.

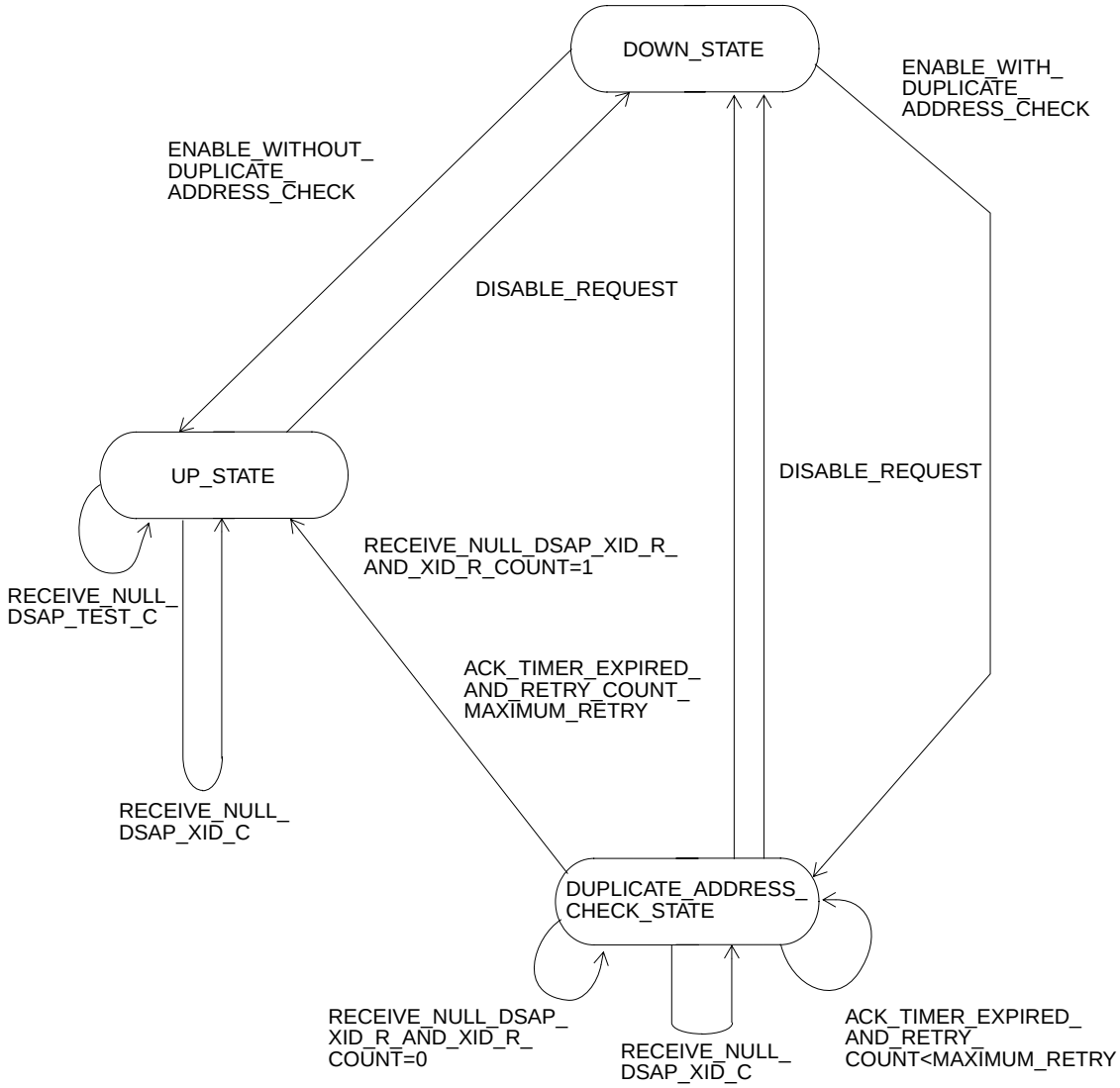


Figure 24—Station component state diagram

c) **DUPLICATE_ADDRESS_FOUND:** LLC entity has detected another LLC entity on the LAN with a MAC service access point address identical to its own.

6.9.3 Service access point (SAP) component overview

The SAP component handles all LLC Type 1 PDU traffic for a particular DSAP address in the local station component. The local service access point user is able to activate and deactivate the operation of each individual SAP component in the station component (see table 3 and figure 25). Once active, the SAP component shall process Type 1 LLC PDUs addressed to the DSAP and send Type 1 LLC PDUs either by service access point user request or as a result of some LLC protocol action.

For Class II stations, the ACTIVE state of the SAP component provides the activating conditions for Type 2 LLC connection component services (see figure 23). Any attempt to make a data link connection, either by the user or a remote LLC, while the SAP component is ACTIVE, shall be passed to the Type 2 LLC connection component and ignored by the SAP component (this includes the handling of the disconnect mode for a Type 2 LLC connection component).

Table 2—Station component options

Description	States omitted	Other requirements
No duplicate address check	DUPLICATE_ADDRESS_CHECK_STATE	Omit: ENABLE_WITH_DUPLICATE_ADDRESS_CHECK ACK_TIMER_EXPIRED_AND_RETRY_COUNT <MAXIMUM_RETRY ACK_TIMER_EXPIRED_AND_RETRY_COUNT=MAXIMUM_RETRY RECEIVE_NULL_DSAP_XID_R_AND_XID_R_COUNT=1 RECEIVE_NULL_DSAP_XID_R_AND_XID_R_COUNT = 0
Optional use of duplicate address check	none	Omit: none
Always perform duplicate address check	none	Omit: ENABLE_WITHOUT_DUPLICATE_ADDRESS_CHECK

Table 3—SAP component state transitions

Current state	Event	Action(s)	Next state
INACTIVE_STATE	SAP_ACTIVATION_REQUEST	REPORT_STATUS(SAP_ACTIVE)	ACTIVE_STATE
ACTIVE_STATE	RECEIVE_UI	UNITDATA_INDICATION	ACTIVE_STATE
	UNITDATA_REQUEST	SEND_UI	ACTIVE_STATE
	XID_REQUEST	SEND_XID_C	ACTIVE_STATE
	RECEIVE_XID_C	SEND_XID_R	ACTIVE_STATE
	RECEIVE_XID_R	XID_INDICATION	ACTIVE_STATE
	TEST_REQUEST	SEND_TEST_C	ACTIVE_STATE
	RECEIVE_TEST_C	SEND_TEST_R	ACTIVE_STATE
	RECEIVE_TEST_R	TEST_INDICATION	ACTIVE_STATE
	SAP_DEACTIVATION_REQUEST	REPORT_STATUS(SAP_INACTIVE)	INACTIVE_STATE

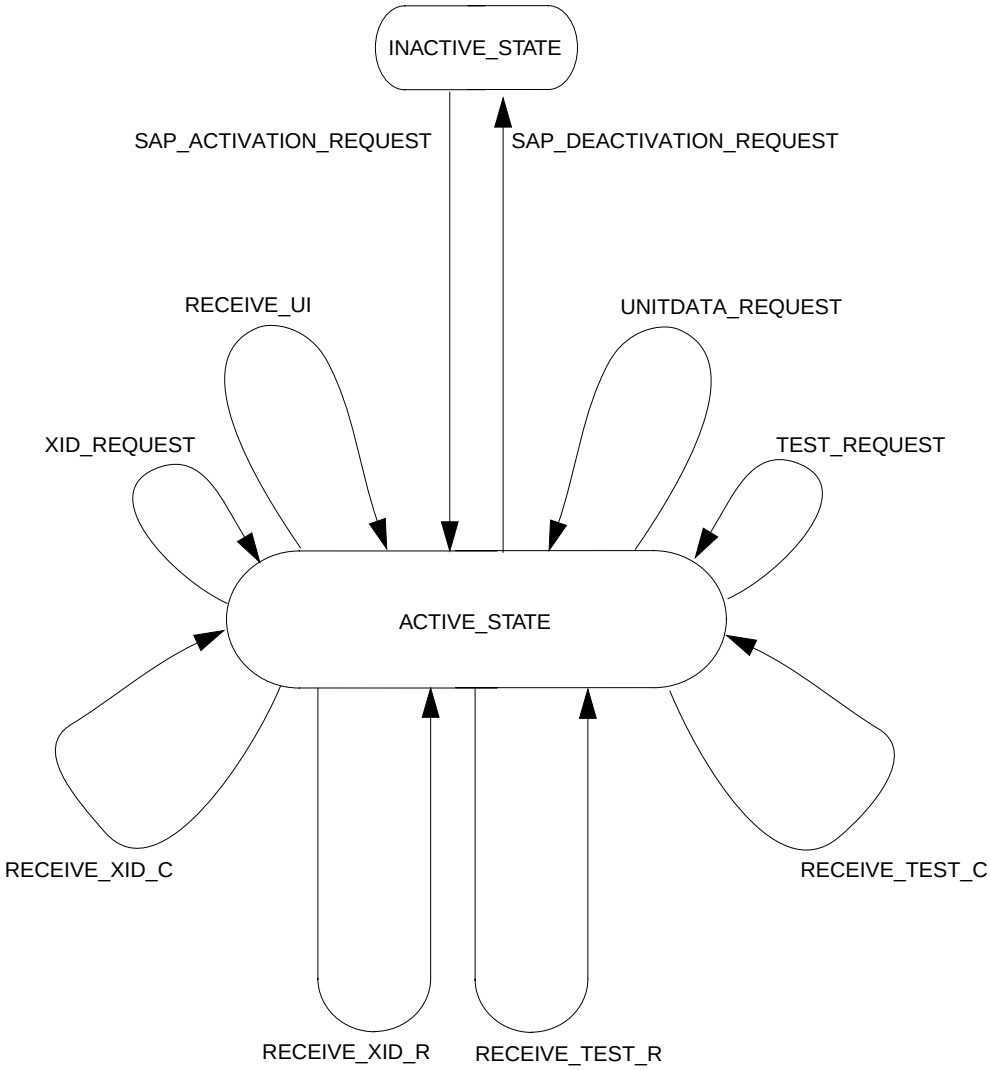


Figure 25—SAP component state diagram

6.9.3.1 SAP component state descriptions

1) **INACTIVE_STATE:** LLC SAP component is not active, functioning, or operational. No PDUs are accepted and/or sent.
2) **ACTIVE_STATE:** LLC SAP component is active, functioning, and operational. PDUs are received and sent.

6.9.3.2 SAP component event description

1) **SAP_ACTIVATION_REQUEST:** The SAP user has requested that the particular LLC SAP component be activated and begin logical data link operation of the Type 1 services.
2) **SAP_DEACTIVATION_REQUEST:** The SAP user has requested that the particular LLC SAP component be deactivated and no longer allowed to operate on the logical data link.

3) **XID_REQUEST:** The SAP user has requested that the LLC SAP component send an XID command PDU to one or more remote SAPs.
4) **TEST_REQUEST:** The SAP user has requested that the LLC SAP component send a TEST command PDU to one or more remote SAPs.
5) **RECEIVE_UI:** The local SAP component has received a UI PDU from a remote SAP.
6) **UNITDATA_REQUEST:** The SAP user has requested that a data unit be passed to a remote LLC SAP, via a UI PDU.
7) **RECEIVE_XID_C:** The local SAP component has received an XID command PDU from a remote SAP.
8) **RECEIVE_XID_R:** The local SAP component has received an XID response PDU from a remote SAP.
9) **RECEIVE_TEST_C:** The local SAP component has received a TEST command PDU from the remote SAP.
10) **RECEIVE_TEST_R:** The local SAP component has received a TEST response PDU from the remote SAP.

6.9.3.3 SAP component action descriptions

1) **UNITDATA_INDICATION:** LLC SAP component has received a UI PDU from a remote SAP. The service data unit is given to the SAP user.
2) **SEND_UI:** A UI PDU is sent to one or more remote SAPs in response to a user request to send a service data unit.
3) **SEND_XID_C:** LLC SAP component shall send an XID command PDU to remote SAPs in response to a SAP user request to identify other SAPs.
4) **SEND_XID_R:** LLC SAP component shall send a XID response PDU to remote SAPs in response to a received XID command PDU.
5) **SEND_TEST_C:** LLC SAP component shall send a TEST command PDU in response to SAP user request to test a remote SAP.
6) **SEND_TEST_R:** LLC SAP component shall send a TEST response PDU in response to a remote LLC TEST command PDU.
7) **REPORT_STATUS:** The LLC SAP component shall be able to report data link status conditions for the particular SAP component with the following valid reasons:
   a) **SAP_ACTIVE:** The SAP_ACTIVATION_REQUEST has been successfully processed and the component is now activated.
   b) **SAP_INACTIVE:** The SAP_DEACTIVATION_REQUEST has been successfully processed and the component is now deactivated.
8) **XID_INDICATION:** LLC SAP component has received an XID response PDU from a remote SAP. An indication of this event is passed to the SAP user, and may also return the XID information field.
9) **TEST_INDICATION:** LLC SAP component has received a TEST response PDU from a remote SAP. An indication of this event is passed to the SAP user, and may also return the TEST information field.

7. LLC description of the Type 2 procedures

7.1 Modes

In Type 2 operation, two modes of operation are defined—an operational mode and a non-operational mode.

7.1.1 Operational mode

The one operational mode shall be the asynchronous balanced mode (ABM).

ABM is a balanced operational mode where a data link connection has been established between two service access points. Either LLC shall be able to send commands at any time and initiate response transmissions without receiving explicit permission from the other LLC. Such an asynchronous transmission shall contain one or more LLC PDUs and shall be used for information field transfer and/or to indicate status changes in the LLC (for example, the number of the next expected information LLC PDU, transition from a ready to a busy condition or vice versa, occurrence of an exception condition).

ABM consists of a data link connection phase, an information transfer phase, a data link resetting phase, and a data link disconnection phase.

An LLC in ABM receiving a DISC command PDU shall respond with the unnumbered acknowledgment (UA) response PDU if it is capable of actioning the command.

7.1.2 Non-operational mode

The one non-operational mode shall be the asynchronous disconnected mode (ADM).

ADM differs from the operational mode (ABM) in that the data link connection is logically disconnected from the physical medium; i.e., no information (user data) shall be sent or accepted.

7.1.2.1 Purpose of ADM

ADM is defined to prevent a data link connection from appearing on the physical medium in a fully operational mode during unusual situations or exception conditions since such operation could cause the following:

1) Sequence number mismatch between the LLCs on the data link connection, or
2) Ambiguity in one LLC as to the status of another LLC.

A data link connection shall be system predefined as to the condition(s) that cause it to assume the asynchronous disconnected mode (ADM).

7.1.2.2 Examples of conditions leading to ADM

Examples of possible conditions (in addition to receiving a DISC command PDU) that shall cause a data link connection to enter ADM are

1) The power is turned on,
2) The data link layer logic is manually reset, or
3) The data link connection is manually switched from a local (home) condition to the connected-on-the-data-link (on-line) condition.

7.1.2.3 LLC requirements on a data link connection in ADM

An LLC on a data link connection in ADM shall be required to monitor input received from its MAC for the purpose of

1) Accepting and responding to one of the mode setting command PDUs (SABME, DISC), or
2) Sending a DM response PDU at a medium access opportunity, when required.

In addition, since the LLC has the ability to send command PDUs at any time, the LLC may send an appropriate mode setting command PDU.

An LLC in ADM receiving a DISC command PDU shall respond with the DM response PDU.

An LLC in ADM shall not establish a frame reject exception condition (see 5.4.2.3.5 and 7.6). ADM consists of a data link disconnected phase.

7.2 Procedure for addressing

The address fields shall be used to indicate the source (SSAP) and destination (DSAP) of the PDU. The first bit in the source address field (SSAP) shall be used to identify whether a command or response is contained in the PDU.

A single data link connection can be established between any two service access points on the local area network. This data link connection is identified by a pair of “complete” data link addresses, each of which consists of a logical concatenation of the implicit physical address (not contained in the frame structure), the MAC address (DA/SA), and the LLC address (DSAP/SSAP). In order for a receiving DSAP to correctly identify the data link connection associated with an incoming PDU, the receiving DSAP must have access to the “complete” data link address information for the remote SAP.

7.3 Procedures for the use of the P/F bit

The LLC receiving a command PDU (SABME, DISC, RR, RNR, REJ, or I) with the P bit set to “1”, shall send a response PDU with the F bit set to “1”.

The response PDU returned by an LLC to a SABME or DISC command PDU with the P bit set to “1” shall be a UA or DM response PDU with the F bit set to “1”. The response PDU returned by an LLC to an I, RR, or REJ command PDU with the P bit set to “1” shall be an I, RR, REJ, RNR, DM, or FRMR response PDU with the F bit set to “1”. The response PDU returned by an LLC to an RNR command PDU with the P bit set to “1” shall be an RR, REJ, RNR, DM, or FRMR response PDU with the F bit set to “1”.

NOTE—The P bit is usable by the LLC in conjunction with the timer recovery condition (see 7.5.9).

7.4 Procedures for data link setup and disconnection

7.4.1 Data link connection phase

Either LLC may be able to initialize the data link connection.

When the LLC wishes to initialize the link, it shall send the SABME command PDU and start the acknowledgment timer (see 7.8.1 below). Upon reception of the UA response PDU, the LLC shall have reset both its send and receive state variables V(S) and V(R) to 0 for the corresponding data link connection, shall stop its acknowledgment timer, and shall enter the information transfer phase.

When receiving the DM response PDU, the LLC that originated the SABME command PDU shall stop its acknowledgment timer, shall not enter the information transfer phase, and shall report to the higher layer for appropriate action.

For a description of the actions to be followed upon receipt of a SABME or DISC command PDU, see 7.4.5. Other Type 2 PDUs received (commands and responses) while attempting to connect shall be ignored by the LLC.

Should the acknowledgment timer run out before reception of the UA or DM response PDU, the LLC shall resend the SABME command PDU and restart the acknowledgment timer. After resending the SABME command PDU N2 times, the sending LLC shall stop sending the SABME command PDU and shall report to the higher layer for the appropriate error recovery action to initiate. The value of N2 is defined in 7.8.2 below.

When receiving an SABME command PDU, the LLC shall pass the indication to the network layer to indicate that an establishment of the data link connection is being requested from the remote LLC.

Thereafter, if the LLC receives a notification from the network layer to accept the connection, it shall return a UA response PDU to the remote LLC and set both its send and receive state variables V(S) and V(R) to 0 for the corresponding data link connection and enter the information transfer phase. The return of the UA response PDU shall take precedence over any other response PDU for the same data link connection that may be pending at the LLC. It shall be possible to follow the UA response PDU with additional LLC PDUs, if pending.

If the LLC receives a notification from the network layer not to enter the indicated phase, it shall return a DM response PDU to the remote LLC and remain in the link disconnected mode.

7.4.2 Information transfer phase

After having sent the UA response PDU to an SABME command PDU or having received the UA response PDU to a sent SABME command PDU, the LLC shall be able to accept or send, or both, I-format and S-format PDUs according to the procedures described in 7.5 below.

When receiving a SABME command PDU while in the information transfer phase, the LLC shall conform to the resetting procedure described in 7.6.

7.4.3 Data link disconnection phase

During the information transfer phase, either LLC shall be able to initiate disconnecting of the data link connection by sending a DISC command PDU.

When the LLC wishes to disconnect the data link connection, it shall send the DISC command PDU and start the acknowledgment timer (see 7.8.1). Upon reception of the UA or DM response PDU from the remote LLC, the LLC shall stop its acknowledgment timer and enter the link disconnected mode.

Should the acknowledgment timer run out before reception of the UA or DM response PDU, the LLC shall resend the DISC command PDU and restart the acknowledgment timer. After sending the DISC command PDU N2 times, the sending LLC shall stop sending the DISC command PDU, shall enter the data link disconnected phase, and shall report to the higher layer for the appropriate error recovery action to initiate. The value of N2 is defined in 7.8.2.

When receiving a DISC command PDU, the LLC shall return a UA response PDU and enter the data link disconnected phase. The return of the UA response PDU shall take precedence over any other response PDU for the same data link connection that may be pending at the LLC.

7.4.4 Data link disconnected phase

After having received a DISC command PDU from the remote LLC and returned a UA response PDU, or having received the UA response PDU to a sent DISC command PDU, the LLC shall enter the data link disconnected phase.

In the disconnected phase, the LLC shall be able to initiate data link connection. In the disconnected phase, the LLC shall react to the receipt of an SABME command PDU as described in 7.4.1 above and shall send a DM response PDU in answer to a received DISC command PDU.

When receiving any other Type 2 command PDU with the P bit set to “1” in the disconnected phase, the LLC shall send a DM response PDU with the F bit set to “1”. Other Type 2 PDUs received in the disconnected phase shall be ignored by the LLC.

7.4.5 Contention of unnumbered mode setting command PDUs

A contention situation in an LLC shall be resolved in the following way. If the sent and received mode setting command PDUs are the same, each LLC shall send the UA response PDU at the earliest opportunity. Each LLC shall enter the indicated phase either after receiving the UA response PDU, or after its acknowledgment timer expires.

If the sent and received mode setting command PDUs are different, each LLC shall enter the data link disconnected phase and shall issue a DM response PDU at the earliest opportunity.

7.5 Procedures for information transfer

The procedures that apply to the transfer of I PDUs in each direction on a data link connection during the information transfer phase are described below.

In the following, “number one higher” is in reference to a continuously repeated sequence series, i.e., 127 is one higher than 126 and 0 is one higher that 127 for modulo 128 series.

7.5.1 Sending I PDUs

When the LLC has an I PDU to send (i.e., and I PDU not already sent, or having to be resent as described in 7.5.5 below), it shall send the I PDU with an N(S) equal to its current send state variable V(S), and an N(R) equal to its current receive state variable V(R) for that data link connection. At the end of sending the I PDU, the LLC shall increment its send state variable V(S) by one.

If the acknowledgment timer is not running at the time that an I PDU is sent, the acknowledgment timer shall be started.

If the data link connection send state variable V(S) is equal to the last value of N(R) received plus k (where k is the maximum number of outstanding I PDUs, see 7.8.4) the LLC shall not send any new I PDUs on that data link connection, but shall be able to resend an I PDU as described in 7.5.6 or 7.5.9.

When a local LLC data link connection is in the busy condition, the LLC shall still be able to send I PDUs, provided that the remote LLC on this data link connection is not busy itself. When the LLC for a particular data link connection is in the FRMR exception condition, it shall stop transmitting I PDUs on that data link connection.

7.5.2 Receiving an I PDU

When the LLC data link connection is not in a busy condition and receives an I PDU whose send sequence number is equal to the receive state variable V(R), the LLC shall accept the information field of this PDU, increment by one it receives state variable V(R), and act as follows:

1) If an I PDU is available to be sent, the LLC shall be able to act as in 7.5.1 above and acknowledge the received I PDU by setting N(R) in the control field of the next sent I PDU to the value of the receive state variable V(R). The LLC shall also be able to acknowledge the received I PDU by sending an RR PDU with the N(R) equal to the value of the receive state variable V(R).
2) If no I PDU is available to be sent by the LLC, then the LLC shall either
   a) Send an RR PDU with the N(R) equal to the value of the receive state variable V(R) at the earliest opportunity; or
   b) If the received PDU was not a command PDU with the P bit set to “1”, wait for some period of time bounded by the probability of the remote acknowledgment timer expiry, for either an I PDU to become available to send, or to accumulate additional I PDUs to be acknowledged in a single RR PDU, subject to window size constraints.
3) If receipt of the I PDU caused the LLC to go into the busy condition with regard to any subsequent I PDUs, the LLC shall send an RNR PDU with the N(R) equal to the value of the receive state variable V(R). If an I PDU(s) is available to send, the LLC shall be able to send them as in 7.5.1 above prior to or following the sending of the RNR PDU.

When the LLC associated with a particular data link connection is in a busy condition, and receives an in-sequence I PDU, the LLC shall be able to ignore the information field contained in any received I PDU on that data link connection (see 7.5.8).

7.5.3 Reception of incorrect PDUs

When the LLC receives an invalid PDU (see 3.3.5) or a PDU with an incorrect DSAP or SSAP address, this PDU shall be discarded entirely.

7.5.4 Reception of out-of-sequence PDUs

When the LLC receives an I PDU whose send sequence number is not in sequence, i.e., not equal to the current receive state variable V(R) but is within the receive window, the LLC shall discard the information field of the I PDU and send an REJ PDU with the N(R) set to the value of V(R). The LLC shall then discard the information field of all I PDUs until the expected I PDU is correctly received. When receiving the expected I PDU, the LLC shall acknowledge the PDU as described in 7.5.2 above. The LLC shall use the N(R) and P bit indications in the discarded I PDUs.

On a given data link connection, only one “sent REJ” exception condition from a given LLC to another given LLC shall be established at a time. A “sent REJ” condition shall be cleared when the requested I PDU is received. The “sent REJ” condition shall be able to be reset when a reject timer time-out function runs out. When the LLC perceives by reject timer time-out that the requested I PDU will not be received, because either the requested I PDU or the REJ PDU was in error or lost, the LLC shall be able to repeat the REJ PDU in order to re-establish the “sent REJ” condition up to N2 times. The value of N2 is defined in 7.8.2.

7.5.5 Receiving acknowledgment

When correctly receiving an I-format or S-format PDU, even in the busy condition (see 7.5.8), the receiving LLC shall consider the N(R) contained in this PDU as an acknowledgment for all the I PDUs it has sent on this data link connection with an N(S) up to an including the received N(R) minus one. The LLC shall reset the acknowledgment timer when it correctly receives an I-format or S-format PDU with the N(R) higher than the last received N(R) (actually acknowledging some I PDUs).

If the timer has been reset and there are outstanding I PDUs still unacknowledged on this data link connection, the LLC shall restart the acknowledgment timer. If the timer then runs out, the LLC shall follow the procedures in 7.5.9 with respect to the unacknowledged I PDUs.

7.5.6 Receiving an REJ PDU

When receiving an REJ PDU, the LLC shall set its send state variable V(S) to the value of the N(R) received in the REJ PDU control field. The LLC shall (re)send the corresponding I PDU as soon as it is available. If other unacknowledged I PDUs had already been sent on that data link connection following the one indicated in the REJ PDU, then those I PDUs shall be resent by the LLC following the resending of the requested I PDU.

If retransmission beginning with a particular PDU occurs due to P/F bit = 1 exchange (see 7.5.9) and an REJ PDU is received that would also start retransmission with the same particular I PDU (as identified by the N(R) in the REJ PDU), the retransmission resulting from the REJ PDU shall be inhibited.

7.5.7 Receiving an RNR PDU

An LLC receiving an RNR PDU shall stop sending I PDUs on the indicated data link connection at the earliest possible time, and shall start the busy-state timer, if not already running. When the busy-state timer runs out, the LLC shall follow the procedure described in 7.5.9. In any case, the LLC shall not send any other I PDUs on that data link connection before receiving an RR or REJ PDU, or before receiving an I response PDU with the F bit set to “1”, or before the completion of a resetting procedure on that data link connection.

7.5.8 LLC busy condition

An LLC shall enter the busy condition on a data link connection when it is temporarily unable to receive or continue to receive I PDUs due to internal constraints; for example, receive buffering limitations. When the LLC enters the busy condition, it shall send an RNR PDU at the earliest opportunity. It shall be possible to send I PDUs awaiting to be sent on that data link connection prior to or following the sending of the RNR PDU. While in the busy condition, the LLC shall accept and process supervisory PDUs and return an RNR response PDU with the F bit set to “1” if it receives a supervisory or I command PDU with the P set to “1” on the affected data link connection.

To indicate the clearance of a busy condition on a data link connection, the LLC shall send either an I PDU with the F bit set to “1” if a P bit set to “1” is outstanding, an REJ PDU, or an RR PDU on the data link connection with N(R) set to the current receive state variable V(R), depending on whether or not the LLC discarded information fields of correctly received I PDUs. Additionally, the sending of a SABME command PDU or a UA response PDU shall indicate the clearance of a busy condition at the sending LLC on a data link connection.

7.5.9 Waiting acknowledgment

The LLC maintains an internal retransmission count variable for each data link connection that shall be set to “0” when the LLC receives or sends a UA response PDU to an SABME command PDU, or when a remote busy condition is cleared, or when the LLC correctly receives an I-format or S-format PDU with the N(R) higher than the last received N(R) (actually acknowledging some outstanding I PDUs).

If the acknowledgment timer, busy-state timer, or, optionally, P-bit timer runs out, the LLC on this data link connection shall enter the timer recovery condition and add one to its retransmission count variable.

The LLC shall then start the P-bit timer and send an S-format command PDU with the P bit set to “1”.

The timer recovery condition shall be cleared on the data link connection when the LLC receives a valid I-format or S-format PDU from the remote LLC, with the F bit set to “1”.

If, while in the timer recovery condition, the LLC correctly receives a valid I-format or S-format PDU with the F bit set to “1” and with the N(R) within the range from the last value of N(R) received to the current send state variable inclusive, the LLC shall clear the timer recovery condition, set its send state variable to the received N(R), stop the P-bit timer, and resend any unacknowledged PDUs.

If, while in the timer recovery condition, the LLC correctly receives a valid I-format or S-format PDU with the P/F bit set to “0” and with a N(R) within the range from the last value of N(R) received to the current send state variable inclusive, the LLC shall not clear the timer recovery condition but shall treat the N(R) value received as an acknowledgment for the indicated previously transmitted I PDUs (see 7.5.5).

If the P-bit timer runs out in the timer recovery condition, the LLC shall add one to its retransmission count variable. If the retransmission count variable is not equal to N2, the LLC shall resend an S-format PDU with the P bit set to “1” and restart its P-bit timer.

If the retransmission count variable is equal to N2, the LLC shall pass a DL-RESET indication primitive to the network layer to indicate the need for a link reset, as described in 7.6 below. N2 is a system parameter (see 7.8.2).

7.5.10 Reception of duplicate I PDUs

When an LLC receives an I PDU whose send sequence number, N(S), is not in the receive window, the PDU may, optionally, be recognized as a duplicate if the N(S) indicates that it is a copy of an I PDU previously received. In that case, the LLC shall discard the information field but use the N(R) and P bit indications in the duplicate PDU. If the LLC does not recognize a duplicate, the N(S) is invalid, and the PDU shall be handled according to 7.7.

Figures 26 and 27 provide graphic representations of the relationships between the valid receive window, the optional duplicate range, and the invalid range for instances where 1) the receive window (RW) is less than half the modulus value, and 2) the receive window (RW) is equal to or greater than half the modulus value.

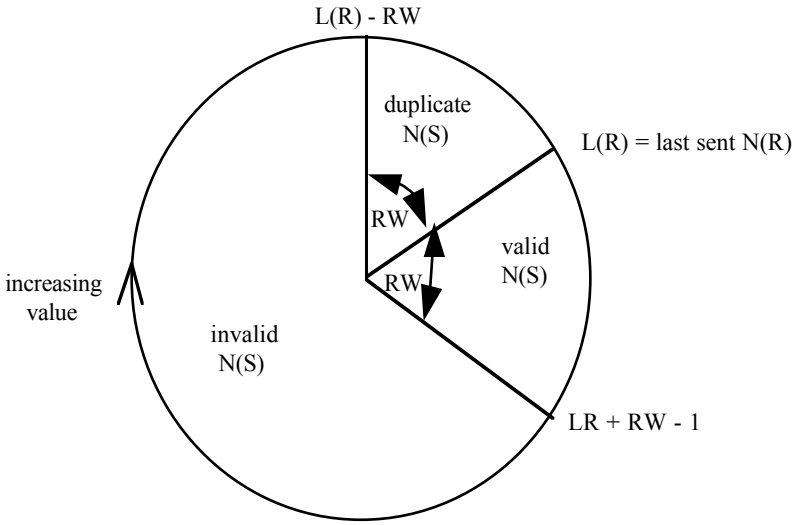


Figure 26—Relationships of receive window, invalid and duplicate ranges (RW < half the modulus value)

As depicted in figure 26, in those instances where the receive window is less than one-half the modulus value, the receive window, the invalid range, and the optional duplicate range relative to L(R) shall be defined as follows:

— The receive window RW contains the sequence numbers L(R) to L(R) + RW - 1 inclusive;
— The invalid range contains the sequence numbers L(R) + RW to L(R) – RW – 1 inclusive;
— The optional duplicate range contains the sequence numbers L(R) – RW to L(R) – 1 inclusive;

where L(R) is equal to the value of the last sent N(R). If the duplicate feature is not used, the invalid range shall contain the sequence numbers L(R) + RW to L(R) – 1 inclusive.

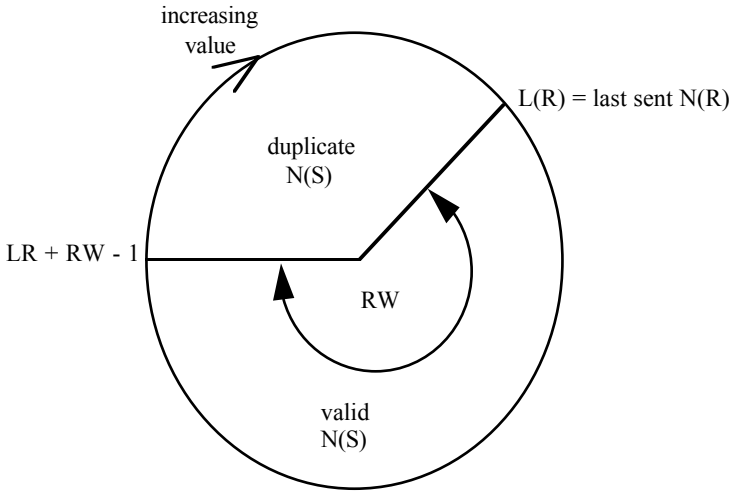


Figure 27—Relationships of receive window, invalid and duplicate ranges (RW ≥ half the modulus value)

As depicted in figure 27, in those instances where the receive window is equal to or greater than one-half the modulus value, the receive window and the optional duplicate range relative to L(R) shall be defined as follows:

— The receive window RW contains the sequence numbers L(R) to L(R) + RW – 1 inclusive;
— The optional duplicate range contains the sequence numbers L(R) + RW to L(R) – 1 inclusive;

where L(R) is equal to the value of the last sent N(R). If the duplicate feature is not used, the optional duplicate range becomes the invalid range.

7.6 Procedures for resetting

The resetting phase is used to initialize both directions of information transfer according to the procedure described below. The resetting phase shall only apply during the asynchronous balanced mode ABM.

Either LLC shall be able to initiate a resetting of both directions by sending an SABME command PDU and starting its acknowledgment timer.

After receiving an SABME command PDU, the LLC shall return, at the earliest opportunity,

1) A UA response PDU and reset its send and receive state variables V(S) and V(R) to 0 to reset the data link connection, or

2) A DM response PDU if the data link connection is to be terminated.

The return of the UA or DM response PDU shall take precedence over any other response PDU for the same data link connection that may be pending at the LLC. It shall be possible to follow the UA PDU with additional LLC PDUs, if pending. If the UA PDU is received correctly by the initiating LLC, it shall reset its send and receive state variables V(S) and V(R) to 0 and stop its acknowledgment timer. This shall also clear all exception conditions that might be present at either of the LLCs involved in the reset. This exchange shall also indicate clearance of any busy condition that may have been present at either LLC involved in the reset.

If a DM response PDU is received, the LLC shall enter the data link disconnected phase, shall stop its acknowledgment timer and shall report to the higher layer for appropriate action. If the acknowledgment timer runs out before a UA or DM response PDU is received, the SABME command PDU shall be resent and the acknowledgment timer shall be started. After the timer runs out N2 times, the sending LLC shall stop sending the SABME command PDU, shall report to the higher layer for the appropriate error recovery actions to initiate, and shall enter the asynchronous disconnected mode. The value of N2 is defined in 7.8.2.

Other Type 2 PDUs (with the exception of the SABME and DISC command PDUs) that are received by the LLC before completion of the reset procedure shall be discarded.

Under certain FRMR exception conditions listed in 7.7, it shall be possible for the LLC to ask the remote LLC to reset the data link connection by sending an FRMR response PDU.

Upon reception of an FRMR response PDU (even during an FRMR exception condition) the LLC shall initiate a resetting procedure by passing a DL-RESET indication primitive to the network layer.

After sending an FRMR response PDU, the LLC shall enter the FRMR exception condition. The FRMR exception conditions shall be cleared when the LLC receives or sends an SABME or DISC command PDU or DM response PDU. Any other Type 2 command PDU received while in the FRMR exception condition shall cause the LLC to resend the FRMR response PDU with the same information field as originally sent.

In the FRMR exception condition, additional I PDUs shall not be sent, and received I-format PDUs and S-format PDUs shall be discarded by the LLC.

It shall be possible for the LLC to start its acknowledgment timer on the sending of the FRMR response PDU. If the timer runs out before the reception of an SABME or DISC command PDU from the remote LLC, it shall be possible for the LLC to resend the FRMR response PDU and restart its acknowledgment timer. After the acknowledgment timer has run out N2 times, the LLC shall pass a DL-RESET indication primitive to the network layer. The value of N2 is defined in 7.8.2.

When an additional FRMR response PDU is sent while the acknowledgment timer is running, the timer shall not be reset or restarted.

7.7 FRMR exception conditions

The LLC shall request a resetting procedure (by sending an FRMR response PDU) as described in 7.6, when receiving, during the information transfer phase, a PDU with one of the conditions identified in 5.4.2.3.5. The coding of the information field of the FRMR response PDU that is sent is given in 5.4.2.3.5.

The LLC shall initiate a resetting procedure (by passing a DL-RESET indication primitive to the network layer) as described in 7.6 when receiving an FRMR response PDU during the information transfer phase.

7.8 List of data link connection parameters

A number of data link connection parameters are defined, the range of values for which are determined on a system-by-system basis by the user at the time that the local area network is established.

The data link connection parameters for Type 2 operation shall be as follows:

7.8.1 Timer functions

In Type 2 operation it is possible for a number of independent events to be taking place on a data link connection that could each employ a timing function. These timing functions are defined below, as identified in the text that describes Type 2 operation. It is understood that these timing functions can be realized by using a number of individual timers, or by using a single timer. If a single timing function is employed, it will be necessary for the designer to determine on an instance-by-instance basis when to reset and restart the timer and when to let it continue running based on the priority assigned to the individual actions that are in progress.

The periods of the timer functions shall take into account whether the timers are started at the beginning or the end of the event that initiated the timer (e.g., sending of a PDU by the LLC), and any delay introduced by the MAC sublayer.

The proper operation of the procedure shall require that the value of the timing functions be greater than the maximum time between the normal network operation of Type 2 PDUs and the reception of the corresponding Type 2 PDU returned as an answer to the initiating Type 2 PDU.

7.8.1.1 Acknowledgment timer

The acknowledgment timer is a data link connection parameter that shall define the time interval during which the LLC shall expect to receive an acknowledgment to one or more outstanding I PDUs or an expected response PDU to a sent unnumbered command PDU.

7.8.1.2 P-bit timer

The P-bit timer is a data link connection parameter that shall define the time interval during which the LLC shall expect to receive a PDU with the F bit set to “1” in response to a sent Type 2 command with the P bit set to “1”.

7.8.1.3 Reject timer

The reject timer is a data link connection parameter that shall define the time interval during which the LLC shall expect to receive a reply to a sent REJ PDU.

7.8.1.4 Busy-state timer

The busy-state timer is a data link connection parameter that shall define the timer interval during which the LLC shall wait for an indication of the clearance of a busy condition at the other LLC.

7.8.2 Maximum number of transmissions, N2

N2 is a data link connection parameter that indicates the maximum number of times that a PDU is sent following the running out of the acknowledgment timer, the P-bit timer, the reject timer, or the busy-state timer.

7.8.3 Maximum number of octets in an I PDU, N1

N1 is a data link connection parameter that denotes the maximum number of octets in an I PDU. Refer to the various MAC descriptions to determine the precise value of N1 for a given medium access method. LLC itself places no restrictions on the value of N1. However, in the interest of having a value of N1 that all users of Type 2 LLC may depend upon, all MACs must at least be capable of accommodating I PDUs with information fields up to an including 128 octets in length.

7.8.4 Transmit window size, k

The transmit window size (k) shall be a data link connection parameter that can never exceed 127. It shall denote the maximum number of sequentially numbered I PDUs that the sending LLC may have outstanding (i.e., unacknowledged). The value of k is the maximum number by which the sending LLC send state variable V(S) can exceed the N(R) of the last received I PDU.

7.8.5 Minimum number of octets in a PDU

A minimal valid data link connection PDU shall contain exactly two address fields and one control field in that order. Thus the minimum number of octets in a valid data link connection PDU shall be 3 or 4, depending on whether the PDU is a U-format PDU, or an I-format or S-format PDU, respectively.

7.8.6 Receive window size, RW

The receive window size (RW) shall be a data link connection parameter that can never exceed 127. It shall denote the maximum number of unacknowledged sequentially numbered I PDUs that the local LLC allows the remote LLC to have understanding. It is transmitted in the information field of XID (see 5.4.1.1.2) and applies to the XID sender. The XID receiver shall set its transmit window (k) to a value less than or equal to the receive window of the XID sender to avoid overrunning the XID sender.

7.9 Precise description the Type 2 procedures

7.9.1 General

If discrepancies appear to exist with the text found in the balance of clause 7, this subclause (7.9) shall be viewed as being the definitive description.

7.9.2 Connection service component overview

The connection service component handles all LLC Type 2 PDU traffic for a specific data link connection (designated by a DA, DSAP–SA, SSAP pair). Once activated, the connection service component shall process Type 2 LLC PDUs addressed to the local service access point from the remote service access point and shall send Type 2 PDUs to the remote service access access point as a result of either a service access point user request or as the result of some data link protocol action. (See figure 28 and table 4.)

When the service access point component (as described in 6.9) is placed in the ACTIVE state, all the connection service components associated with the service access point are placed in the ADM (asynchronous disconnected mode) state. When the service access point component leaves the ACTIVE state, all of the associated connection service components are deactivated, regardless of the current state of the connection service component.

The following points apply to the interpretation of the state tables:

1) Flag variables are used to limit the number of states by maintaining the state of particular conditions affecting the connection component.
The flags defined are
P_FLAG,
F_FLAG,
S_FLAG,
DATA_FLAG, and
REMOTE_BUSY.
2) In the list of events, events of the form RECEIVE_XXX_YYY are listed. The interpretation is that this event is the reception of any command PDU or response PDU not specifically listed for that state.
3) For some combinations of state and events(s), the table provides alternative actions. These are separated by horizontal dotted lines in the ACTIONS and NEXT STATE columns. Selection of an alternative is done on the basis of (a) local status, (b) the result of layer management action, or (c) implementation decision. There is no relationship between the order of alternatives between events, nor is it implied that the same alternative must be selected every time the event occurs.
4) In the list of actions there is no implied ordering, unless one or more of the actions is conditional upon the value(s) of flag(s) that are modified by other actions. In this case the test(s) must be completed before flag(s) are modified.
5) In the list of actions, actions of the form SEND_XXX_RSP (F=l) are indicated. It should be noted that if some other response PDU (with the F bit set to “0”) will be sent earlier, it is permissible to modify theat PDU from F bit set to “0” to F bit set to “1”, and to send the new PDU with the F bit to “0”. This could occur, for example, if an LLC implementation managed the queue of PDUs awaiting transmission.
6) For simplicity, the state table has four timers: the ACK_TIMER for timing acknowledgments, the P_TIMER for timing the P/F cycle, the REJ_TIMER for timing the “sent REJ” condition, and the BUSY_TIMER for timing the “remote busy” condition. It should be noted that by the addition of appropriate flags a functionally equivalent state table can be developed that requires only one timer.
7) Any START_TIMER action (re)starts the specified timer from zero, even if the timer is already running. When the timer reaches its limit, the appropriate TIMER_EXPIRED condition is set and the timer stopped. The TIMER_EXPIRED condition is cleared when it is recognized by the connection component state machine. The STOP_TIMER action stops the timer if it is running or clears the TIMER_EXPIRED condition if the timer has already reached its limit.
8) Events not recognized in a particular state are assumed to remain pending until any masking flag is modified or a transition is made to a state where they can be recognized.

NOTE—To ensure proper interpretation of the state table, the descriptions of the entries (see 7.9.2.1–7.9.2.3) should be read in concert with the state tables.

7.9.2.1 Connection component state descriptions

1) **ADM:** The connection component is in the asynchronous disconnected mode. It can accept an SABME PDU from a remote LLC SSAP or, at the request of the service access point user, can initiate an SABME PDU transmission to a remote LLC DSAP, to establish a data link connection. It also responds to a DISC command PDU, and to any command PDU with the P bit set to “1”.
2) **SETUP:** The connection component has transmitted an SABME command PDU to a remote LLC DSAP and is waiting for a reply.
3) **NORMAL:** A data link connection exists between the local LLC service access point and the remote LLC service access point. Sending and reception of information and supervisory PDUs can be performed.
4) **BUSY:** A data link connection exists between the local LLC service access point and the remote LLC service access point. I PDUs may be sent. Local conditions make it likely that the information field of received I PDUs will be ignored. Supervisory PDUs may be both sent and received.

5) **REJECT:** A data link connection exists between the local LLC service access point and the remote LLC service access point. The local connection component has requested that the remote connection component resend a specific I PDU that the local connection component has detected as being out of sequence. Both I PDUs and supervisory PDUs may be sent and received.
6) **AWAIT:** A data link connection exists between the local LLC service access point and the remote LLC service access point. The local LLC is performing a timer recovery operation and has sent a command PDU with the P bit set to “1”, and is awaiting an acknowledgment from the remote LLC. I PDUs may be received but not sent. Supervisory PDUs may be both sent and received.
7) **AWAIT_BUSY:** A data link connection exists between the local LLC service access point and the remote LLC service access point. The local LLC is performing a timer recovery operation and has sent a command PDU with the P bit set to “1”, and is awaiting an acknowledgment from the remote LLC. I PDUs may not be sent. Local conditions make it likely that the information field of received I PDUs will be ignored. Supervisory PDUs may be both sent and received.
8) **AWAIT_REJECT:** A data link connection exists between the local LLC service access point and the remote LLC service access point. The local connection component has requested that the remote connection component re-transmit a specific I PDU that the local connection component has detected as being out of sequence. Before the local LLC entered this state it was performing a timer recovery operation and had sent a command PDU with the P bit set to “1”, and is still awaiting an acknowledgment from the remote LLC. I PDUs may be received but not sent. Supervisory PDUs may be both sent and received.
9) **D_CONN:** At the request of the service access point user, the local LLC has sent a DISC command PDU to the remote LLC DSAP and is waiting for a reply.
10) **RESET:** As a result of a service access point user request or the receipt of an FRMR response PDU, the local connection component has sent an SABME command PDU to the remote LLC DSAP to reset the data link connection and is waiting for a reply.
11) **ERROR:** The local connection component has detected an error in a received PDU and has sent an FRMR response PDU. It is waiting for a reply from the remote connection component.
12) **CONN:** The local connection component has received an SABME PDU from a remote LLC SSAP, and it is waiting for the local user to accept or refuse the connection.
13) **RESET_CHECK:** The local connection component is waiting for the local user to accept or refuse a remote reset request.
14) **RESET_WAIT:** The local connection component is waiting for the local user to indicate a RESET_REQUEST or a DISCONNECT_REQUEST.

7.9.2.2 Connection service component event description

In the list of events below, the value of the P or F bits in received commands and responses is listed as X. In the state transition table, values of 0, 1, or X are used. The latter indicates that either 0 or 1 may occur in the event.

1) **CONNECT_REQUEST:** The user has requested that a data link connection be established with a remote LLC DSAP.
2) **CONNECT_RESPONSE:** The user has accepted the data link connection.
3) **DATA_REQUEST:** The user has requested that a data unit be sent to the remote LLC DSAP.
4) **DISCONNECT_REQUEST:** The user has requested that the data link connection with the remote LLC DSAP be terminated.
5) **RESET_REQUEST:** The user has requested that the data link connection with the remote LLC DSAP be reset.
6) **RESET_RESPONSE:** The user has accepted the reset of the data link connection.
7) **LOCAL_BUSY_DETECTED:** The local station has entered a busy condition and may not be able to accept I PDUs from the remote LLC SSAP.
8) **LOCAL_BUSY_CLEARED:** The local station busy condition has ended and the station can accept I PDUs from the remote LLC SSAP.

9) **RECEIVE_BAD_PDU:** The remote SSAP has sent to the local DSAP a command or response PDU that is not implemented, or has an information field when not permitted, or is an I PDU with an information field length greater than can be accommodated by the local LLC.
10) **RECEIVE_DISC_CMD(P=X):** The remote SSAP has sent a DISC command PDU with the P bit set to "X" addressed to the local DSAP.
11) **RECEIVE_DM_RSP(F=X):** The remote SSAP has sent a DM response PDU with the F bit set to "X" addressed to the local DSAP.
12) **RECEIVE_FRMR_RSP(F=X):** The remote SSAP has sent an FRMR response PDU with the F bit set to "X" addressed to the local DSAP.
13) **RECEIVE_I_CMD(P=X):** The remote SSAP has sent an I command PDU with the P bit set to "X" addressed to the local DSAP. Both the N(R) and N(S) fields are valid and the N(S) value is the expected sequence number.
14) **RECEIVE_I_CMD(P=X)_WITH_UNEXPECTED_N(S):** The remote SSAP has sent an I command PDU with the P bit set to "X" addressed to the local DSAP. The N(S) field of the command does not contain the expected sequence number but is within the window size. The N(R) field is valid.
15) **RECEIVE_I_CMD(P=X)_WITH_INVALID_N(S):** The remote SSAP has sent an I command PDU with the P bit set to "X" addressed to the local DSAP. The N(S) field of the command is invalid. The N(R) field is valid.
16) **RECEIVE_DUPLICATE_I_CMD(P=X).** The remote SSAP has sent an I command PDU with the P bit set to "X" addressed to the local DSAP. The value of the N(S) field of the command is not within the receive window size. The LLC recognizes the PDU as a duplicate. The N(R) field is valid.
17) **RECEIVE_I_RSP(F=X):** The remote SSAP has sent an I response PDU with the F bit set to "X" addressed to the local DSAP. Both the N(R) and N(S) fields are valid and the N(S) value is the expected sequence number.
18) **RECEIVE_I_RSP(F=X)_WITH_UNEXPECTED_N(S):** The remote SSAP has sent an I response PDU with the F bit set to "X" addressed to the local DSAP. The N(S) field of the command does not contain the expected sequence number but is within the window size. The N(R) field is valid.
19) **RECEIVE_I_RSP(F=X)_WITH_INVALID_N(S):** The remote SSAP has sent an I response PDU with the F bit set to "X" addressed to the local DSAP. The N(S) field of the response is invalid. The N(R) field is valid.
20) **RECEIVE_DUPLICATE_I_RSP(F=X).** The remote SSAP has sent an I response PDU with the F bit set to "X" addressed to the local DSAP. The value of the N(S) field of the response is not within the receive window size. The LLC recognizes the PDU as a duplicate. The N(R) field is valid.
21) **RECEIVE_REJ_CMD(P=X):** The remote SSAP has sent an REJ command PDU with the P bit set to "X" addressed to the local DSAP.
22) **RECEIVE_REJ_RSP(F=X):** The remote SSAP has sent an REJ response PDU with the F bit set to "X" addressed to the local DSAP.
23) **RECEIVE_RNR_CMD(P=X):** The remote SSAP has sent an RNR command PDU with the P bit set to "X" addressed to the local DSAP.
24) **RECEIVE_RNR_RSP(F=X):** The remote SSAP has sent an RNR response PDU with the F bit set to "X" addressed to the local DSAP.
25) **RECEIVE_RR_CMD (P=X):** The remote SSAP has sent an RR command PDU with the P bit set to "X" addressed to the local DSAP.
26) **RECEIVE_RR_RSP(F=X):** The remote SSAP has sent an RR response PDU with the F bit set to "X" addressed to the local DSAP.
27) **RECEIVE_SABME_CMD (P=X):** The remote SSAP has sent an SABME command PDU with the P bit set to "X" addressed to the local DSAP.
28) **RECEIVE_UA_RSP(F=X):** The remote SSAP has sent a UA response PDU with the F bit set to "X" addressed to the local DSAP.

29) **RECEIVE_XXX_CMD(P=X):** The remote SSAP has sent a Type 2 command PDU with the P bit set to “X” addressed to the local DSAP. The command is any command not specifically listed for that state.
30) **RECEIVE_XXX_RSP(F=X):** The remote SSAP has sent a Type 2 response PDU with the F bit set to “X” addressed to the local DSAP. The response is any response not specifically listed for that state.
31) **RECEIVE_XXX_YYY:** The remote SSAP has sent a Type 2 PDU addressed to the local DSAP. The PDU is any command or response not specifically listed for that state.
32) **RECEIVE_ZZZ_CMD(P=X)_WITH_INVALID_N(R):** The remote SSAP has sent an I, RR, RNR, or REJ command PDU with the P bit set to “X” addressed to the local DSAP. The N(R) field of the command is invalid.
33) **RECEIVE_ZZZ_RSP(F=X)_WITH_INVALID_N(R):** The remote SSAP has sent an I, RR, RNR, or REJ response PDU with the F bit set to “X” addressed to the local DSAP. The N(R) field of the response is invalid.
34) **P_TIMER_EXPIRED:** The P/F cycle timer has expired.
35) **ACK_TIMER_EXPIRED:** The acknowledgment timer has expired.
36) **REJ_TIMER_EXPIRED:** The “sent REJ” timer has expired.
37) **BUSY_TIMER_EXPIRED:** The remote-busy timer has expired.

In the state transition table some of the above events are qualified by the following conditions. The event is recognized only when the condition is true.

38) **DATA_FLAG=1:** When DATA_FLAG has a value of one, data unit(s) from I PDUs were discarded during a local busy period.
39) **DATA_FLAG=0:** When DATA_FLAG has a value of zero, data unit(s) from I PDUs were not discarded during a local busy period.
40) **DATA_FLAG=2:** When DATA_FLAG has a value of two, the BUSY state was entered from the REJECT state, and the requested I PDU has not yet been received.
41) **P_FLAG=1:** P_FLAG has a value of one when a command with the P bit set to “1” has been sent and a response with the F bit set to “1” is expected.
42) **P_FLAG=0:** P_FLAG has a value of zero when a response PDU with the F bit set to “1” is not expected.
43) **P_FLAG=F:** P_FLAG has a value equal to the F bit in the response PDU received.
44) **REMOTE_BUSY=0:** When REMOTE_BUSY has a value of zero, DATA_REQUEST events are acted upon and I PDUs can be sent. Note that when REMOTE_BUSY has a value of one, DATA_REQUEST events are not included in the state transition tables.
45) **RETRY_COUNT<N2:** The number of retries is less than the maximum number or retries.
46) **RETRY_COUNT>=N2:** The number of retries has reached the maximum number permissible.
47) **S_FLAG=1:** In the SETUP, RESET, and RESET_WAIT states, an S_FLAG value of one indicates that an SABME PDU has been received.
48) **S_FLAG=0:** In the SETUP, RESET, and RESET_WAIT states, an S_FLAG value of zero indicates that an SABME PDU has not been received.
49) **INITIATE_P/F_CYCLE:** The local LLC wants to initiate a P/F cycle. (This is only required if the local LLC is not generating other command PDUs for some reason.)

7.9.2.3 Connection component action description

In the list of actions described below the value of the P or F bits in the transmitted commands and responses is listed as X. In the state transition table, values of 0, 1, or X are used. The latter indicates that either 0 or 1 may be used.

1) **CLEAR_REMOTE_BUSY:** If REMOTE_BUSY has a value of one, then set REMOTE_BUSY to zero to indicate the remote LLC is now able to accept I PDUs, stop the BUSY_TIMER, inform the user by issuing REPORT_STATUS (REMOTE_NOT_BUSY) and, provided the local LLC is in the

NORMAL, REJECT, or BUSY state, start the (re)sending of any I PDUs that were waiting for the remote busy to be cleared.

2) **CONNECT_INDICATION:** Inform the user that a connection has been requested by a remote LLC SSAP.
3) **CONNECT_CONFIRM:** The connection service component indicates that the remote network entity has accepted the connection.
4) **DATA_INDICATION:** The connection service component passes the data unit from the received I PDU to the user.
5) **DISCONNECT_INDICATION:** Inform the user that the remote network entity has initiated disconnection of the data link connection.
6) **RESET_INDICATION:** Inform the user that either the remote network entity or the remote LLC component has initiated a reset of the data link connection, or that the local LLC has determined that the data link connection is in need of reinitialization. The valid results are
   a) **REMOTE:** The remote network entity or remote peer has initiated a reset of the data link connection.
   b) **LOCAL:** The local LLC has determined that the data link connection is in need of reinitialization.
7) **RESET_CONFIRM:** The connection service component indicates that the remote network entity has accepted the reset.
8) **REPORT_STATUS:** Report the status of the data link connection to the sublayer management function. Permissible status values are
   a) **FRMR_RECEIVED:** The local connection service component has received a FRMR response PDU.
   b) **FRMR_SENT:** The local connection service component has received an invalid PDU, and has sent a FRMR response PDU.
   c) **REMOTE_BUSY:** The remote LLC DSAP is busy. The local connection service component will not accept a DATA_REQUEST.
   d) **REMOTE_NOT_BUSY:** The remote LLC DSAP is no longer busy. The local connection service component will now accept a DATA_REQUEST.
9) **IF_F=1_CLEAR_REMOTE_BUSY:** If the I PDU is a response with the F bit set to “1” in response to a command PDU with the P bit set to “1”, then perform the CLEAR_REMOTE_BUSY action.
10) **IF_DATA_FLAG=2_STOP_REJ_TIMER:** If DATA_FLAG has a value of two, indicating that a REJ PDU has been sent, stop the “sent REJ” timer.
11) **SEND_DISC_CMD(P=X):** Transmit a DISC command PDU with the P bit set to “X” to the remote LLC DSAP.
12) **SEND_DM_RSP(F=X):** Send a DM response PDU with the F bit set to “X” to the remote LLC DSAP.
13) **SEND_FRMR_RSP(F=X):** Send a FRMR response PDU with the F bit set to “X” to the remote LLC DSAP.
14) **RE-SEND_FRMR_RSP(F=O):** Send the same FRMR response PDU with the same information field as sent earlier to the remote LLC DSAP. Set the F bit to “0”.
15) **RE-SEND_FRMR_RSP(F=P):** Send the same FRMR response PDU with the same information field as sent earlier to the remote LLC DSAP. Set the F bit equal to the P bit of the received command PDU.
16) **SEND_I_CMD(P=1):** Send an I command PDU with the P bit set to “1” to the remote LLC DSAP with the data unit supplied by the user with the DATA_REQUEST. Before sending, copy the current values of the send state variable V(S) and the receive state variable V(R) into the N(S) and N(R) fields, respectively, of the I PDU and increment (modulo 128) the send state variable V(S).
17) **RE-SEND_I_CMD(P=1):** Start resending all the unacknowledged I PDUs for this data link connection beginning with the N(R) given in the received PDU. Send the first as a command with the P bit set to “1”. If the queue contains more than one I PDU, the balance must be sent as commands with the P bit set to “0”, or as responses with the F bit set to “0”.

18) **RE-SEND_I_CMD(P=1)_OR_SEND_RR:** Start resending all the unacknowledged I PDUs for this data link connection beginning with the N(R) given in the received PDU. Send the first as a command with the P bit set to “1”. If the queue contains more than one I PDU the balance must be sent as commands with the P bit set to “0”, or as responses with the F bit set to “0”. It is permissible to send a RR command PDU with the P bit set to “1” to the remote LLC DSAP before starting the resending of the I PDUs. In this case, the first I PDU is sent as a command with the P bit set to “0”, or as a response with the F bit set to “0”. If no I PDU is ready to send, a RR command PDU with the P bit set to “1” must be sent to the remote LLC DSAP.
19) **SEND_I_XXX(X=0):** Send either an I response PDU with the F bit set to “0” or an I command PDU with the P bit set to “0” to the remote LLC DSAP with the data unit supplied by the user with the DATA_REQUEST. Before sending, copy the current values of the send state variable V(S) and the receive state variable V(R) into the N(S) and N(R) fields, respectively, of the I PDU and increment (modulo 128) the send state variable V(S).
20) **RE-SEND_I_XXX(X=0):** Start resending all the unacknowledged I PDUs for this data link connection beginning with the N(R) given in the received PDU. They must be sent as either commands with the P bit set to “0” or as responses with the F bit set to “0”.
21) **RE-SEND_I_XXX(X=0)_OR_SEND_RR:** Start resending all the unacknowledged I PDUs for this data link connection beginning with the N(R) given in the received PDU. They must be sent as either commands with the P bit set to “0” or as responses with the F bit set to “0”. It is permissible to send either a RR response PDU with the F bit set to “0” or an RR command PDU with the P bit set to “0” to the remote LLC DSAP before starting the resending of the I PDUs. If no I PDU is ready to send either an RR response PDU with the F bit set to “0” or an RR command PDU with the P bit set to “0” must be sent to the remote LLC DSAP.
22) **RE-SEND_I_RSP(F=1):** Start resending all the unacknowledged I PDUs for this data link connection beginning with the N(R) given in the received PDU. Send the first as a response with the F bit set to “1”. If the queue contains more than one I PDU, the balance must be transmitted as commands with the P bit set to “0” or as responses with the F bit set to “0”.
23) **SEND_REJ_CMD(P=1):** Send a REJ command PDU with the P bit set to “1” to the remote LLC DSAP.
24) **SEND_REJ_RSP(F=1):** Send a REJ response PDU with the F bit set to “1” to the remote LLC DSAP.
25) **SEND_REJ_XXX(X=0):** Send either a REJ response PDU with the F bit set to “0” or a REJ command PDU with the P bit set to “0” to the remote LLC DSAP.
26) **SEND_RNR_CMD(P=1):** Send a RNR command PDU with the P bit set to “1” to the remote LLC DSAP.
27) **SEND_RNR_RSP(F=1):** Send a RNR response PDU with the F bit set to “1” to the remote LLC DSAP.
28) **SEND_RNR_XXX(X=0):** Send either a RNR response PDU with the F bit set to “0” or a RNR command PDU with the P bit set to “0” to the remote LLC DSAP.
29) **SET_REMOTE_BUSY:** If REMOTE_BUSY is zero, then set REMOTE_BUSY to one to indicate the remote LLC is in the busy state and is not able to accept I PDUs, start the BUSY_TIMER, inform the sublayer management function by using REPORT-STATUS (REMOTE_BUSY), and stop any (re)sending of an I PDU that is in progress. If REMOTE_BUSY is equal to one, then start the BUSY_TIMER, if not running.
30) **OPTIONAL_SEND_RNR_XXX(X=0):** It is permissible to send a RNR command PDU with the P bit set to “0” or a RNR response PDU with the F bit set to “0” to the remote LLC DSAP in case the remote LLC did not receive the first RNR sent when the busy state was entered.
31) **SEND_RR_CMD(P=1):** Send a RR command PDU with the P bit set to “1” to the remote LLC DSAP.
32) **SEND_ACKNOWLEDGE_CMD(P=1):** Under all conditions it is permissible to send a RR command PDU with the P bit set to “1” to the remote LLC DSAP. If no I PDU is ready to send, the RR command PDU with the P bit set to “1” must be sent to the remote LLC DSAP. (This RR PDU may be delayed by a time bounded by the ACK_TIMER value, to wait for the generation of an I PDU.)

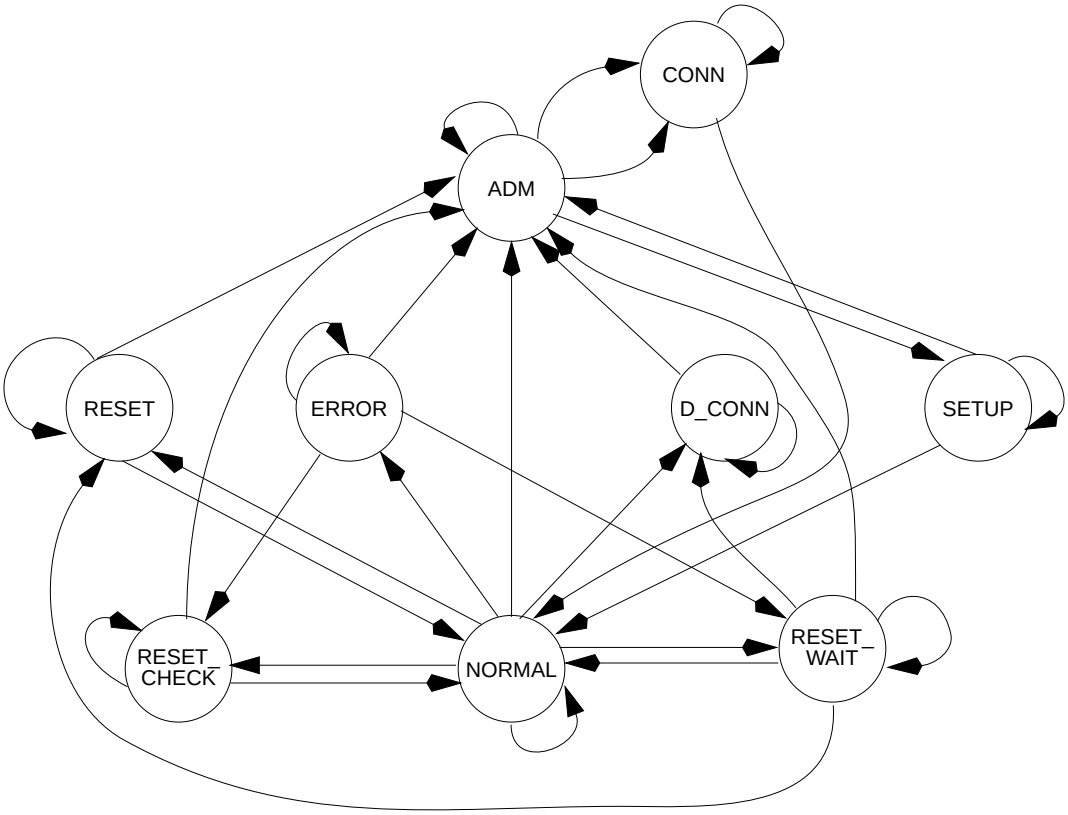
However, if an I PDU is ready to send, and can be modified to a command with the P bit set to “1”, then the RR command PDU does not need to be sent.

33) **SEND_RR_RSP(F=1):** Send a RR response PDU with the F bit set to “1” to the remote LLC DSAP.
34) **SEND_ACKNOWLEDGE_RSP(F=1):** Under all conditions it is permissible to send a RR response PDU with the F bit set to “1” to the remote LLC DSAP. If no I PDU is ready to send, the RR response PDU with the F bit set to “1” must be sent to the remote LLC DSAP. However, if an I PDU is ready to send, and can be modified to a response with the F bit set to “1”, then the RR response PDU does not need to be sent.
35) **SEND_RR_XXX(X=0):** Send either a RR response PDU with the F bit set to “0” or a RR command PDU with the P bit set to “0” to the remote LLC DSAP.
36) **SEND_ACKNOWLEDGE_XXX(X=0):** Under all conditions it is permissible to send either a RR response PDU with the F bit set to “0” or a RR command PDU with the P bit set to “0” to the remote LLC DSAP. If no I PDU is ready to send, either an RR response with the F bit set to “0” or an RR command PDU with the P bit set to “0” must be sent to the remote LLC DSAP. (This RR PDU may be delayed, by a time bounded by the ACK_TIMER value, to wait for the generation of an I PDU.) However, if an I PDU is ready to send, then the RR PDU does not need to be sent.
37) **SEND_SABME_CMD(P=X):** Send a SABME command PDU with the P bit set to “X” to the remote LLC DSAP.
38) **SEND_UA_RSP(F=X):** Send a UA response PDU with the F bit set to “X” to the remote LLC DSAP.
39) **S_FLAG:=0:** Set S_FLAG to zero to indicate that a SABME PDU has not been received from the remote LLC while the local connection service component is in the RESET, SETUP, or RESET_ WAIT state.
40) **S_FLAG:=1:** Set S_FLAG to one to indicate that a SABME PDU has been received from the remote LLC while the local connection service component is in the RESET, SETUP, or RESET_ WAIT state.
41) **START_P_TIMER:** Start the P/F cycle timer from zero; if the P_FLAG is zero, initialize RETRY_COUNT to zero, and set P_FLAG to one.
42) **START_ACK_TIMER:** Start the acknowledgment timer from zero.
43) **START_REJ_TIMER:** Start the “sent REJ” timer from zero.
44) **START_ACK_TIMER_IF_NOT_RUNNING:** If the acknowledgment timer is not currently running, then start the acknowledgment timer from zero.
45) **STOP_ACK_TIMER:** Stop the acknowledgment timer.
46) **STOP_P_TIMER:** Stop the P/F cycle timer and set P_FLAG to zero.
47) **STOP_REJ_TIMER:** Stop the “sent REJ” timer.
48) **STOP_ALL_TIMERS:** Stop the P/F cycle timer, the “sent REJ” timer, the remote-busy timer, and the acknowledgment timer.
49) **STOP_OTHER_TIMERS:** Stop the P/F cycle timer, the “sent REJ” timer, and the remote-busy timer.
50) **UPDATE_N(R)_RECEIVED:** If the N(R) of the received PDU acknowledges the receipt of one or more previously unacknowledged I PDUs, update the local record of N(R)_RECEIVED, set RETRY_COUNT to zero, and stop the acknowledgment timer. If unacknowledged I PDUs still exist, start the acknowledgment timer if it was stopped.

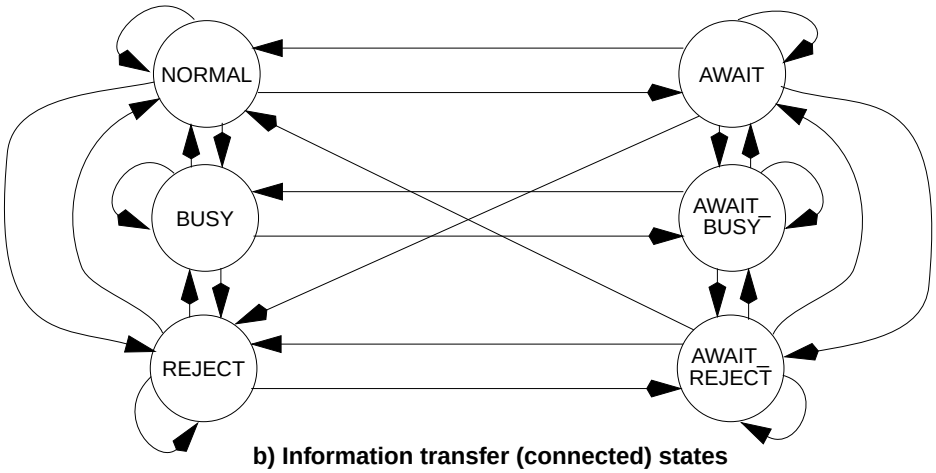
    NOTE—If some form of SEND_I_PDU is initiated at the same time as UPDATE_N(R)_RECEIVED, then the acknowledgment timer is always started if it was stopped.

51) **UPDATE_P_FLAG:** If the received PDU was a response with the F bit set to “1”, set the P_FLAG to zero and stop the P/F cycle timer.
52) **DATA_FLAG:=2:** Set the DATA_FLAG to two to record that the BUSY state was entered with a REJ PDU outstanding.
53) **DATA_FLAG:=0:** Set the DATA_FLAG to zero to indicate that the data units from received I PDUs were not discarded during a local busy period.

54) **DATA_FLAG:=1:** Set the DATA_FLAG to one to indicate that the data units from received I PDUs were discarded during a local busy period.
55) **IF_DATA_FLAG=0_THEN_DATA_FLAG:=1:** If the DATA_FLAG had been zero, indicating that no data units had been discarded, set it to one to indicate that data units have now been discarded.
56) **P_FLAG:=0:** Initialize the P_FLAG to zero. This indicates that the reception of a response PDU with the F bit set to "1" is not expected.
57) **P_FLAG:=P:** Set the P_FLAG to the value of the P bit in the command PDU being sent.
58) **REMOTE_BUSY:=0:** Set REMOTE_BUSY to zero to indicate that the remote LLC is able to accept I PDUs.
59) **RETRY_COUNT:=0:** Initialize RETRY_COUNT to zero.
60) **RETRY_COUNT:=RETRY_COUNT+1:** Increment RETRY_COUNT by one.
61) **V(R):=0:** Initialize the receive state variable to zero. This is the expected sequence number of the next I PDU received.
62) **V(R):=V(R)+1:** Increment (modulo 128) the receive state variable. This is the expected sequence number of the next I PDU received.
63) **V(S):=0:** Initialize the send state variable to zero. This is the sequence number of the next I PDU to be sent.
64) **V(S):=N(R):** Reset the send state variable to the value specified by the N(R) field of the PDU just received. This is the sequence number of the next I PDU to be sent.
65) **F_FLAG:=P:** Set the F_FLAG to the value of the P bit received. This is the value of the F bit to be sent in UA or DM PDUs.



a) Data link establishment, disconnection, and resetting states



b) Information transfer (connected) states

Figure 28—Connection component state diagram

Table 4—Connection component state transitions

Current state	Event	Action(s)	Next state
ADM	CONNECT_REQUEST	SEND_SABME_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=0 S_FLAG:=0 ..... DISCONNECT_INDICATION	SETUP ..... ADM
	RECEIVE_SABME_CMD(P=X)	CONNECT_INDICATION P_FLAG:=P	CONN
	RECEIVE_DISC_CMD(P=X)	SEND_DM_RSP(F=P)	ADM
	RECEIVE_XXX_CMD(P=1)	SEND_DM_RSP(F=1)	ADM
	RECEIVE_XXX_CMD(P=0) or RECEIVE_XXX_RSP(F=X)		ADM
CONN	CONNECT_RESPONSE	SEND_UA_RSP(F=F_FLAG) V(S):=0 V(R):=0 RETRY_COUNT:=0 P_FLAG:=0 REMOTE_BUSY:=0	NORMAL
	DISCONNECT_REQUEST	SEND_DM_RSP(F=F_FLAG)	ADM
	RECEIVE_SABME_CMD(P=X)	F_FLAG:=P	CONN
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION	ADM
	RECEIVE_XXX_YYY		CONN
RESET_ WAIT	RESET_REQUEST and S_FLAG=0	SEND_SABME_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=0	RESET
	RESET_REQUEST and S_FLAG=1	SEND_UA_RSP(F=F_FLAG) V(S):=0 V(R):=0 RETRY_COUNT:=0 P_FLAG:=0 REMOTE_BUSY:=0 RESET_CONFIRM	NORMAL
	DISCONNECT_REQUEST and S_FLAG=0	SEND_DISC_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=0	D_CONN
	DISCONNECT_REQUEST and S_FLAG=1	SEND_DM_RSP(F=F_FLAG)	ADM

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
RESET_ WAIT *(Con'd.)*	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION	ADM
	RECEIVE_SABME_CMD(P=X)	S_FLAG:=1 F_FLAG:=P	RESET_WAIT
	RECEIVE_DISC_CMD(P=X)	SEND_DM_RSP(F=P) DISCONNECT_INDICATION	ADM
	RECEIVE_XXX_YYY		RESET_WAIT
RESET_ CHECK	RESET_RESPONSE	SEND_UA_RSP(F=F_FLAG) V(S):=0 V(R):=0 RETRY_COUNT:=0 P_FLAG:=0 REMOTE_BUSY:=0	NORMAL
	DISCONNECT_REQUEST	SEND_DM_RSP(F=F_FLAG)	ADM
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION	ADM
	RECEIVE_SABME_CMD(P=X)	F_FLAG:=P	RESET_ CHECK
	RECEIVE_DISC_CMD(P=X)	SEND_DM_RSP(F=P) DISCONNECT_INDICATION	ADM
	RECEIVE_XXX_YYY		RESET_ CHECK
SETUP	RECEIVE_SABME_CMD(P=X)	SEND_UA_RSP(F=P) V(S):=0 V(R):=0 RETRY_COUNT:=0 S_FLAG:=1	SETUP
	RECEIVE_UA_RSP(F=X) and P_FLAG=F	STOP_ACK_TIMER V(S):=0 V(R):=0 RETRY_COUNT:=0 UPDATE_P_FLAG CONNECT_CONFIRM REMOTE_BUSY:=0	NORMAL
	ACK_TIMER_EXPIRED and S_FLAG=1	P_FLAG:=0 CONNECT_CONFIRM REMOTE_BUSY:=0	NORMAL
	RECEIVE_DISC_CMD(P=X)	SEND_DM_RSP(F=P) DISCONNECT_INDICATION STOP_ACK_TIMER	ADM
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION STOP_ACK_TIMER	ADM

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
SETUP *(Con'd.)*	RECEIVE_XXX_YYY		SETUP
	ACK_TIMER_EXPIRED and RETRY_COUNT<N2 and S_FLAG=0	SEND_SABME_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	SETUP
	ACK_TIMER_EXPIRED and RETRY_COUNT>=N2 and S_FLAG=0	DISCONNECT_INDICATION	ADM
RESET	RECEIVE_SABME_CMD(P=X)	SEND_UA_RSP(F=P) V(S):=0 V(R):=0 RETRY_COUNT:=0 S_FLAG:=1	RESET
	RECEIVE_UA_RSP(F=X) and P_FLAG=F	STOP_ACK_TIMER V(S):=0 V(R):=0 RETRY_COUNT:=0 UPDATE_P_FLAG RESET_CONFIRM REMOTE_BUSY:=0	NORMAL
	ACK_TIMER_EXPIRED and S_FLAG=1	P_FLAG:=0 RESET_CONFIRM REMOTE_BUSY:=0	NORMAL
	RECEIVE_DISC_CMD(P=X)	SEND_DM_RSP(F=P) DISCONNECT_INDICATION STOP_ACK_TIMER	ADM
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION STOP_ACK_TIMER	ADM
	RECEIVE_XXX_YYY		RESET
	ACK_TIMER_EXPIRED and RETRY_COUNT <N2 and S_FLAG=0	SEND_SABME_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	RESET
	ACK_TIMER_EXPIRED and RETRY_COUNT >=N2 and S_FLAG=0	DISCONNECT_INDICATION	ADM

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
D_CONN	RECEIVE_SABME_CMD(P=X)	SEND_DM_RSP(F=P) STOP_ACK_TIMER	ADM
	RECEIVE_UA_RSP(F=X) and P_FLAG=F	STOP_ACK_TIMER	ADM
	RECEIVE_DISC_CMD(P=X)	SEND_UA_RSP(F=P)	D_CONN
	RECEIVE_DM_RSP(F=X)	STOP_ACK_TIMER	ADM
	RECEIVE_XXX_YYY		D_CONN
	ACK_TIMER_EXPIRED and RETRY_COUNT <N2	SEND_DISC_CMD(P=X) P_FLAG:=P START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	D_CONN
	ACK_TIMER_EXPIRED and RETRY_COUNT>=N2		ADM
ERROR	RECEIVE_SABME_CMD(P=X)	RESET_INDICATION(REMOTE) STOP_ACK_TIMER F_FLAG:=P	RESET_ CHECK
	RECEIVE_DISC_CMD(P=X)	SEND_UA_RSP(F=P) DISCONNECT_INDICATION STOP_ACK_TIMER	ADM
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION STOP_ACK_TIMER	ADM
	RECEIVE_FRMR_RSP(F=X)	RESET_INDICATION(LOCAL) STOP_ACK_TIMER REPORT_STATUS(FRMR_RECEIVED) S_FLAG:=0	RESET_WAIT
	RECEIVE_XXX_CMD(P=X)	RE-SEND_FRMR_RSP(F=P) START_ACK_TIMER	ERROR
	RECEIVE_XXX_RSP(F=X)		ERROR
	ACK_TIMER_EXPIRED and RETRY_COUNT<N2	RE-SEND_FRMR_RSP(F=0) START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	ERROR
	ACK_TIMER_EXPIRED and RETRY_COUNT>=N2	S_FLAG:=0 RESET_INDICATION(LOCAL)	RESET_WAIT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
NORMAL or BUSY or REJECT or AWAIT or AWAIT_ BUSY or AWAIT_ REJECT	DISCONNECT_REQUEST	SEND_DISC_CMD(P=X) P_FLAG:=P START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0	D_CONN
	RESET_REQUEST	SEND_SABME_CMD(P=X) P_FLAG:=P START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0 S_FLAG:=0	RESET
	RECEIVE_SABME_CMD(P=X)	RESET_INDICATION(REMOTE) F_FLAG:=P STOP_ALL_TIMERS	RESET_ CHECK
	RECEIVE_DISC_CMD(P=X)	SEND_UA_RSP(F=P) DISCONNECT_INDICATION STOP_ALL_TIMERS	ADM
	RECEIVE_FRMR_RSP(F=X)	STOP_ALL_TIMERS RESET_INDICATION(LOCAL) REPORT_STATUS(FRMR_RECEIVED) S_FLAG:=0	RESET_WAIT
	RECEIVE_DM_RSP(F=X)	DISCONNECT_INDICATION STOP_ALL_TIMERS	ADM
	RECEIVE_ZZZ_CMD(P=X)_ WITH_INVALID_N(R) or RECEIVE_I_CMD(P=X)_ WITH_INVALID_N(S)	SEND_FRMR_RSP(F=P) REPORT_STATUS(FRMR_SENT) START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0	ERROR
	RECEIVE_ZZZ_RSP(F=X)_ WITH_INVALID_N(R) or RECEIVE_I_RSP(F=X)_ WITH_INVALID_N(S) or RECEIVE_BAD_PDU	SEND_FRMR_RSP(F=0) REPORT_STATUS(FRMR_SENT) START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0	ERROR
	RECEIVE_UA_RSP(F=X)	SEND_FRMR_RSP(F=0) REPORT_STATUS(FRMR_SENT) START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0	ERROR ..... Remain in current state

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
NORMAL or BUSY or REJECT or AWAIT or AWAIT_ BUSY or AWAIT_ REJECT *(con'd.)*	RECEIVE_XXX_RSP(F=1)_ AND_P_FLAG=0	SEND_FRMR_RSP(F=0) REPORT_STATUS(FRMR_SENT) START_ACK_TIMER STOP_OTHER_TIMERS RETRY_COUNT:=0 ..... Remain in current state	ERROR ..... Remain in current state
	P_TIMER_EXPIRED and RETRY_COUNT>=N2 or ACK_TIMER_EXPIRED and RETRY_COUNT>=N2 or REJ_TIMER_EXPIRED and RETRY_COUNT>=N2 or BUSY_TIMER_EXPIRED and RETRY_COUNT>=N2	STOP_ALL_TIMERS RESET_INDICATION(LOCAL) S_FLAG:=0	RESET_WAIT
NORMAL	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=0	SEND_I_CMD(P=1) START_P_TIMER START_ACK_TIMER_ IF_NOT_RUNNING ..... SEND_I_XXX(X=0) START_ACK_ TIMER_ IF_NOT_RUNNING	NORMAL ..... NORMAL
	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=1	SEND_I_XXX(X=0) START_ACK_TIMER_ IF_NOT_RUNNING	NORMAL
	LOCAL_BUSY_DETECTED and P_FLAG=0	SEND_RNR_CMD(P=1) START_P_TIMER DATA_FLAG:=0 ..... SEND_RNR_XXX(X=0) DATA_FLAG:=0	BUSY ..... BUSY
	LOCAL_BUSY_DETECTED and P_FLAG=1	SEND_RNR_XXX(X=0) DATA_FLAG:=0	BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
NORMAL *(con'd.)*	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=0 or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=0 or RECEIVE_I_RSP(F=1)_ WITH_UNEXPECTED_N(S) and P_FLAG=1	SEND_REJ_XXX(X=0) UPDATE_N(R)_RECEIVED UPDATE_P_FLAG START_REJ_TIMER IF_F=1_CLEAR_REMOTE_BUSY ..... SEND_REJ_CMD(P=1) UPDATE_N(R)_RECEIVED START_P_TIMER START_REJ_TIMER IF_F=1_CLEAR_REMOTE_BUSY	REJECT ..... REJECT
	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=1 or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=1	SEND_REJ_XXX(X=0) UPDATE_N(R)_RECEIVED START_REJ_TIMER	REJECT
	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_REJ_RSP(F=1) UPDATE_N(R)_RECEIVED START_REJ_TIMER	REJECT
	RECEIVE_I_RSP(F=X) and P_FLAG=F or RECEIVE_I_CMD(P=0) and P_FLAG=0	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_CMD(P=1) START_P_TIMER UPDATE_N(R)_RECEIVED IF_F=1_CLEAR_REMOTE_BUSY ..... V(R):=V(R)+1 DATA_INDICATION UPDATE_P_FLAG SEND_ACKNOWLEDGE_XXX(X=0) UPDATE_N(R)_RECEIVED IF_F=1_CLEAR_REMOTE_BUSY	NORMAL ..... NORMAL
	RECEIVE_I_RSP(F=0) and P_FLAG=1 or RECEIVE_I_CMD(P=0) and P_FLAG=1	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_XXX(X=0) UPDATE_N(R)_RECEIVED	NORMAL
	RECEIVE_I_CMD(P=1)	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED	NORMAL
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED	NORMAL

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
NORMAL *(con'd.)*	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	NORMAL
	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_RR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	NORMAL
	RECEIVE_RR_CMD(P=1)	SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	NORMAL
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0) or RECEIVE_RNR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	NORMAL
	RECEIVE_RNR_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	NORMAL
	RECEIVE_REJ_CMD(P=0) and P_FLAG=0 or RECEIVE_REJ_RSP(F=X) and P_FLAG=F	V(S):=N(R) UPDATE_N(R)_RECEIVED UPDATE_P_FLAG RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... V(S):=N(R) UPDATE_N(R)_RECEIVED START_P_TIMER RE-SEND_I_CMD(P=1) CLEAR_REMOTE_BUSY	NORMAL ..... NORMAL
	RECEIVE_REJ_CMD(P=0) and P_FLAG=1 or RECEIVE_REJ_RSP(F=0) and P_FLAG=1	V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY	NORMAL
	RECEIVE_REJ_CMD(P=1)	V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_|_RSP(F=1) CLEAR_REMOTE_BUSY	NORMAL

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
NORMAL *(con'd.)*	INITIATE_P/F_CYCLE and P_FLAG=0	SEND_RR_CMD(P=1) START_P_TIMER	NORMAL
	P_TIMER_EXPIRED and_RETRY_COUNT<N2	P_FLAG:=0 ..... SEND_RR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	NORMAL ..... AWAIT
	ACK_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2 or BUSY_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2	SEND_RR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	AWAIT
BUSY	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=0	SEND_I_CMD(P=1) START_P_TIMER START_ACK_TIMER_ IF_NOT_RUNNING ..... SEND_I_XXX(X=0) START_ACK_TIMER_ IF_NOT_RUNNING	BUSY ..... BUSY
	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=1	SEND_I_XXX(X=0) START_ACK_TIMER_ IF_NOT_RUNNING	BUSY
	LOCAL_BUSY_CLEARED and DATA_FLAG=1 and P_FLAG=0	SEND_REJ_CMD(P=1) START_REJ_TIMER START_P_TIMER ..... SEND_REJ_XXX(X=0) START_REJ_TIMER	REJECT ..... REJECT
	LOCAL_BUSY_CLEARED and DATA_FLAG=1 and P_FLAG=1	SEND_REJ_XXX(X=0) START_REJ_TIMER	REJECT
	LOCAL_BUSY_CLEARED and DATA_FLAG=0 and P_FLAG=0	SEND_RR_CMD(P=1) START_P_TIMER ..... SEND_RR_XXX(X=0)	NORMAL ..... NORMAL
	LOCAL_BUSY_CLEARED and DATA_FLAG=0 and P_FLAG=1	SEND_RR_XXX(X=0)	NORMAL

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
BUSY *(con'd.)*	LOCAL_BUSY_CLEARED and DATA_FLAG=2 and P_FLAG=0	SEND_RR_CMD(P=1) START_P_TIMER ..... SEND_RR_XXX(X=0)	REJECT ..... REJECT
	LOCAL_BUSY_CLEARED and DATA_FLAG=2 and P_FLAG=1	SEND_RR_XXX(X=0)	REJECT
	RECEIVE _I_RSP(F=X)_ WITH_UNEXPECTED_N(S) and P_FLAG=F or RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=0	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_P_FLAG UPDATE_N(R)_RECEIVED IF_DATA_FLAG=0_THEN_ DATA_FLAG:=1 IF_F=1_CLEAR_REMOTE_BUSY ..... SEND_RNR_CMD(P=1) START_P_TIMER UPDATE_N(R)_RECEIVED IF_DATA_FLAG=0_THEN_ DATA_FLAG:=1 IF_F=1_CLEAR_REMOTE_BUSY	BUSY ..... BUSY
	RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=1 or RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) and P_FLAG=1	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=0_THEN_ DATA_FLAG:=1	BUSY
	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=0_THEN_ DATA_FLAG:=1	BUSY
	RECEIVE_I_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=1 ..... V(R):=V(R)+1 DATA_INDICATION SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=0	BUSY ..... BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
BUSY *(con'd.)*	RECEIVE_I_RSP(F=X) and P_FLAG=F or RECEIVED_I_CMD(P=0) and P_FLAG=0	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_P_FLAG UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=1 IF_F=1_CLEAR_REMOTE_BUSY	BUSY
		.....	.....
		SEND_RNR_CMD(P=1) START_P_TIMER UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=1 IF_F=1_CLEAR_REMOTE_BUSY	BUSY
		.....	.....
		V(R):=V(R)+1 DATA_INDICATION SEND_RNR_CMD(P=1) START_P_TIMER UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=0 IF_F=1_CLEAR_REMOTE_BUSY	BUSY
		.....	.....
		V(R):=V(R)+1 DATA_INDICATION UPDATE_P_FLAG OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=0 IF_F=1_CLEAR_REMOTE_BUSY	BUSY
	RECEIVE_I_RSP(F=0) and P_FLAG=1 or RECEIVE_I_CMD(P=0) and P_FLAG=1	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=1	BUSY
		.....	.....
		V(R):=V(R)+1 DATA_INDICATION OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED IF_DATA_FLAG=2_ STOP_REJ_TIMER DATA_FLAG:=0	BUSY
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED	BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
BUSY *(con'd.)*	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	BUSY
	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_RR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	BUSY
	RECEIVE_RR_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	BUSY
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0) or RECEIVE_RNR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	BUSY
	RECEIVE_RNR_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	BUSY
	RECEIVE_REJ_CMD(P=0) and P_FLAG=0 or RECEIVE_REJ_RSP(F=X) and P_FLAG=F	V(S):=N(R) UPDATE_N(R)_RECEIVED UPDATE_P_FLAG RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_CMD(P=1) CLEAR_REMOTE_BUSY	BUSY ..... BUSY
	RECEIVE_REJ_CMD(P=0) and P_FLAG=1 or RECEIVE_REJ_RSP(F=0) and P_FLAG=1	V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY	BUSY
	RECEIVE_REJ_CMD(P=1)	V(S):=N(R) UPDATE_N(R)_RECEIVED SEND_RNR_RSP(F=1) RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY	BUSY
	INITIATE_P/F_CYCLE and P_FLAG=0	SEND_RNR_CMD(P=1) START_P_TIMER	BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
BUSY *(con'd.)*	P_TIMER_EXPIRED and RETRY_COUNT<N2	P_FLAG:=0 ..... SEND_RNR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	BUSY ..... AWAIT_ BUSY
	ACK_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2 or BUSY_TIMER_EXPIRED and P-FLAG=0 and RETRY_COUNT <N2	SEND_RNR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	AWAIT_ BUSY
	REJ_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2	DATA_FLAG:=1 ..... SEND_RNR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1 DATA_FLAG:=1	BUSY ..... AWAIT_ BUSY
	REJ_TIMER_EXPIRED and P_FLAG=1 and RETRY_COUNT <N2	DATA_FLAG:=1	BUSY
REJECT	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=0	SEND_I_CMD(P=1) START_P_TIMER START_ACK_TIMER_IF_NOT_ RUNNING ..... SEND_I_XXX(X=0) START_ACK_TIMER_IF_NOT_ RUNNING	REJECT ..... REJECT
	DATA_REQUEST and REMOTE_BUSY=0 and P_FLAG=1	SEND_I_XXX(X=0) START_ACK_TIMER_IF_NOT_ RUNNING	REJECT
	LOCAL_BUSY_DETECTED and P_FLAG=0	SEND_RNR_CMD(P=1) START_P_TIMER DATA_FLAG:=2 ..... SEND_RNR_XXX(X=0) DATA_FLAG:=2	BUSY ..... BUSY
	LOCAL_BUSY_DETECTED and P_FLAG=1	SEND_RNR_XXX(X=0) DATA_FLAG:=2	BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
REJECT *(con'd.)*	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S) or RECEIVE_I_RSP(F=1)_ WITH_UNEXPECTED_N(S) and P_FLAG=1	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	REJECT
	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED	REJECT
	RECEIVE_I_RSP(F=X) and P_FLAG=F or RECEIVE_I_CMD(P=0) and P_FLAG=0	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_CMD(P=1) START_P_TIMER UPDATE_N(R)_RECEIVED IF_F=1_CLEAR_REMOTE_BUSY STOP_REJ_TIMER ..... V(R):=V(R)+1 DATA_INDICATION UPDATE_P_FLAG SEND_ACKNOWLEDGE_XXX(X=0) UPDATE_N(R)_RECEIVED IF_F=1_CLEAR_REMOTE_BUSY STOP_REJ_TIMER	NORMAL ..... NORMAL
	RECEIVE_I_RSP(F=0) and P_FLAG=1 or RECEIVE_I_CMD(P=0) and P_FLAG=1	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_XXX(X-0) UPDATE_N(R)_RECEIVED STOP_REJ_TIMER	NORMAL
	RECEIVE_I_CMD(P=1)	V(R):=V(R)+1 DATA_INDICATION SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED STOP_REJ_TIMER	NORMAL
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED	REJECT
	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	REJECT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
REJECT *(con'd.)*	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_RR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	REJECT
	RECEIVE_RR_CMD(P=1)	SEND_ACKNOWLEDGE_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	REJECT
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0) or RECEIVE_RNR_RSP(F=1) and P_FLAG=1	UPDATE_P_FLAG UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	REJECT
	RECEIVE_RNR_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	REJECT
	RECEIVE_REJ_CMD(P=0) and P_FLAG=0 or RECEIVE_REJ_RSP(F=X) and P_FLAG=F	V(S):=N(R) UPDATE_N(R)_RECEIVED UPDATE_P_FLAG RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_CMD(P=1) START_P_TIMER CLEAR_REMOTE_BUSY	REJECT ..... REJECT
	RECEIVE_REJ_CMD(P=0) and P_FLAG=1 or RECEIVE_REJ_RSP(F=0) and P_FLAG=1	V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY	REJECT
	RECEIVE_REJ_CMD(P=1)	V(S):=N(R) UPDATE_N(R)_RECEIVED RE-SEND_I_RSP(F=1) CLEAR_REMOTE_BUSY	REJECT
	INITIATE_P/F_CYCLE and P_FLAG=0	SEND_RR_CMD(P=1) START_P_TIMER	REJECT
	REJ_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2	SEND_REJ_CMD(P=1) START_P_TIMER START_REJ_TIMER RETRY_COUNT:=RETRY_COUNT+1	REJECT ..... NORMAL

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
REJECT *(con'd.)*	P_TIMER_EXPIRED and RETRY_COUNT <N2	P_FLAG:=0 ..... SEND_RR_CMD(P=1) START_P_TIMER START_REJ_TIMER RETRY_COUNT:=RETRY_COUNT+1	REJECT ..... AWAIT_ REJECT
	ACK_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2 or BUSY_TIMER_EXPIRED and P_FLAG=0 and RETRY_COUNT <N2	SEND_RR_CMD(P=1) START_P_TIMER START_REJ_TIMER RETRY_COUNT:=RETRY_COUNT+1	AWAIT_ REJECT
AWAIT	LOCAL_BUSY_DETECTED	SEND_RNR_XXX(X=0) DATA_FLAG:=0	AWAIT_ BUSY
	RECEIVE_I_RSP(F=1)_ WITH_UNEXPECTED_N(S)	SEND_REJ_XXX_(X=0) UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER RE-SEND_I_XXX(X=0) START_REJ_TIMER CLEAR_REMOTE_BUSY ..... SEND_REJ_CMD(P=1) UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_XXX(X=0) START_P_TIMER START_REJ_TIMER CLEAR_REMOTE_BUSY	REJECT ..... REJECT
	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S)	SEND_REJ_XXX(X=0) UPDATE_N(R)_RECEIVED START_REJ_TIMER	AWAIT_ REJECT
	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_REJ_RSP(F=1) UPDATE_N(R)_RECEIVED START_REJ_TIMER	AWAIT_ REJECT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT *(con'd.)*	RECEIVE_I_RSP(F=1)	V(R):=V(R)+1 DATA_INDICATION UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_CMD(P=1)_OR_SEND_RR START_P_TIMER CLEAR_REMOTE_BUSY ..... V(R):=V(R)+1 DATA_INDICATION STOP_P_TIMER UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_XXX(X=0)_OR_SEND_RR CLEAR_REMOTE_BUSY	NORMAL ..... NORMAL
	RECEIVE_I_RSP(F=0) or RECEIVE_I_CMD(P=0)	V(R):=V(R)+1 DATA_INDICATION SEND_RR_XXX(X=0) UPDATE_N(R)_RECEIVED	AWAIT
	RECEIVE_I_CMD(P=1)	V(R):=V(R)+1 DATA_INDICATION SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED	AWAIT
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED	AWAIT
	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	AWAIT
	RECEIVE_RR_RSP(F=1) or RECEIVE_REJ_RSP(F=1)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_CMD(P=1) START_P_TIMER CLEAR_REMOTE_BUSY	NORMAL ..... NORMAL
	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_REJ_CMD(P=0) or RECEIVE_REJ_RSP(F=0)	UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT *(con'd.)*	RECEIVE_RR_CMD(P=1) or RECEIVE_REJ_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT
	RECEIVE_RNR_RSP(F=1)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER SET_REMOTE_BUSY	NORMAL
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0)	UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT
	RECEIVE_RNR_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT
	P_TIMER_EXPIRED and RETRY_COUNT <N2	SEND_RR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	AWAIT
AWAIT_ BUSY	LOCAL_BUSY_CLEARED and DATA_FLAG=1	SEND_REJ_XXX(X=0) START_REJ_TIMER	AWAIT_ REJECT
	LOCAL_BUSY_CLEARED and DATA_FLAG=0	SEND_RR_XXX(X=0)	AWAIT
	LOCAL_BUSY_CLEARED and DATA_FLAG=2	SEND_RR_XXX(X=0)	AWAIT_ REJECT
	RECEIVE_I_RSP(F=1)_ WITH_UNEXPECTED_N(S)	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER DATA_FLAG:=1 CLEAR_REMOTE_BUSY RE-SEND_I_XXX(X=0) ..... SEND_RNR_CMD(P=1) UPDATE_N(R)_RECEIVED V(S):=N(R) START_P_TIMER DATA_FLAG:=1 CLEAR_REMOTE_BUSY RE-SEND_I_XXX(X=0)	BUSY ..... BUSY
	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S)	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED DATA_FLAG:=1	AWAIT_ BUSY
	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED DATA_FLAG:=1	AWAIT_ BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT_ BUSY *(con'd.)*	RECEIVE_I_RSP(F=1)	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED V(S):=N(R) DATA_FLAG:=1 STOP_P_TIMER CLEAR_REMOTE_BUSY RE-SEND_I_XXX(X=0) ..... SEND_RNR_CMD(P=1) V(R):=V(R)+1 DATA_INDICATION START_P_TIMER UPDATE_N(R)_RECEIVED V(S):=N(R) DATA_FLAG:=0 CLEAR_REMOTE_BUSY RE-SEND_I_XXX(X=0) ..... OPTIONAL_SEND_RNR_XXX(X=0) V(R):=V(R)+1 DATA_INDICATION STOP_P_TIMER UPDATE_N(R)_RECEIVED V(S):=N(R) DATA_FLAG:=0 CLEAR_REMOTE_BUSY RE-SEND_I_XXX(X=0)	BUSY ..... BUSY ..... BUSY
	RECEIVE_I_RSP(F=0) or RECEIVE_I_CMD(P=0)	OPTIONAL_SEND_RNR_XXX(X=0) UPDATE_N(R)_RECEIVED DATA_FLAG:=1 ..... OPTIONAL_SEND_RNR_XXX(X=0) V(R):=V(R)+1 DATA_INDICATION UPDATE_N(R)_RECEIVED DATA_FLAG:=0	AWAIT_ BUSY ..... AWAIT_ BUSY
	RECEIVE_I_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED DATA_FLAG:=1 ..... SEND_RNR_RSP(F=1) V(R):=V(R)+1 DATA_INDICATION UPDATE_N(R)_RECEIVED DATA_FLAG:=0	AWAIT_ BUSY ..... AWAIT_ BUSY
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED	AWAIT_ BUSY

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT_ BUSY *(con'd.)*	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	AWAIT_ BUSY
	RECEIVE_RR_RSP(F=1) or RECEIVE_REJ_RSP(F=1)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_CMD(P=1) START_P_TIMER CLEAR_REMOTE_BUSY	BUSY ..... BUSY
	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_REJ_CMD(P=0) or RECEIVE_REJ_RSP(F=0)	UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT_ BUSY
	RECEIVE_RR_CMD(P=1) or RECEIVE_REJ_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT_ BUSY
	RECEIVE_RNR_RSP(F=1)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER SET_REMOTE_BUSY	BUSY
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0)	UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT_ BUSY
	RECEIVE_RNR_CMD(P=1)	SEND_RNR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT_ BUSY
	P_TIMER_EXPIRED and RETRY_COUNT <N2	SEND_RNR_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_ COUNT+1	AWAIT_ BUSY
AWAIT_ REJECT	LOCAL_BUSY_DETECTED	SEND_RNR_XXX(X=0) DATA_FLAG:=2	AWAIT_ BUSY
	RECEIVE_I_CMD(P=0)_ WITH_UNEXPECTED_N(S) or RECEIVE_I_RSP(F=0)_ WITH_UNEXPECTED_N(S)	UPDATE_N(R)_RECEIVED	AWAIT_ REJECT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT_ REJECT *(con'd.)*	RECEIVE_I_CMD(P=1)_ WITH_UNEXPECTED_N(S)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED	AWAIT_ REJECT
	RECEIVE_I_RSP(F=1)	V(R):=V(R)+1 DATA_INDICATION UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_CMD(P=1)_OR_SEND_RR START_P_TIMER STOP_REJ_TIMER CLEAR_REMOTE_BUSY ..... V(R):=V(R)+1 DATA_INDICATION STOP_P_TIMER STOP_REJ_TIMER UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_XXX(X=0)_OR_SEND_RR CLEAR_REMOTE_BUSY	NORMAL ..... NORMAL
	RECEIVE_I_RSP(F=0) or RECEIVE_I_CMD(P=0)	V(R):=V(R)+1 DATA_INDICATION SEND_RR_XXX(X=0) STOP_REJ_TIMER UPDATE_N(R)_RECEIVED	AWAIT
	RECEIVE_I_CMD(P=1)	V(R):=V(R)+1 DATA_INDICATION SEND_RR_RSP(F=1) STOP_REJ_TIMER UPDATE_N(R)_RECEIVED	AWAIT
	RECEIVE_DUPLICATE_ I_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED	AWAIT_ REJECT
	RECEIVE_DUPLICATE_ I_CMD(P=0) or RECEIVE_DUPLICATE_ I_RSP(F=X)	UPDATE_N(R)_RECEIVED UPDATE_P_FLAG IF_F=1_CLEAR_REMOTE_BUSY	AWAIT_ REJECT
	RECEIVE_RR_RSP(F=1) or RECEIVE_REJ_RSP(F=1) or RECEIVE_I_RSP(F=1)_ WITH_UNEXPECTED_N(S)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER RE-SEND_I_XXX(X=0) CLEAR_REMOTE_BUSY ..... UPDATE_N(R)_RECEIVED V(S):=N(R) RE-SEND_I_CMD(P=1) START_P_TIMER CLEAR_REMOTE_BUSY	REJECT ..... REJECT

Table 4—Connection component state transitions (*Continued*)

Current state	Event	Action(s)	Next state
AWAIT_ REJECT *(con'd.)*	RECEIVE_RR_CMD(P=0) or RECEIVE_RR_RSP(F=0) or RECEIVE_REJ_CMD(P=0) or RECEIVE_REJ_RSP(F=0)	UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT_ REJECT
	RECEIVE_RR_CMD(P=1) or RECEIVE_REJ_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED CLEAR_REMOTE_BUSY	AWAIT_ REJECT
	RECEIVE_RNR_RSP(F=1)	UPDATE_N(R)_RECEIVED V(S):=N(R) STOP_P_TIMER SET_REMOTE_BUSY	REJECT
	RECEIVE_RNR_CMD(P=0) or RECEIVE_RNR_RSP(F=0)	UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT_ REJECT
	RECEIVE_RNR_CMD(P=1)	SEND_RR_RSP(F=1) UPDATE_N(R)_RECEIVED SET_REMOTE_BUSY	AWAIT_ REJECT
	P_TIMER_EXPIRED and RETRY_COUNT <N2	SEND_REJ_CMD(P=1) START_P_TIMER RETRY_COUNT:=RETRY_COUNT+1	AWAIT_ REJECT

8. LLC description of the Type 3 procedures

8.1 Modes of operation

In Type 3 operation, no modes of operation are defined. An LLC using Type 3 procedures shall support the entire procedure set for all LSAPs that are configured for Type 3 operation.

8.2 Procedure for addressing

The address fields shall be used to indicate the source (SSAP) and destination (DSAP) of the LLC PDU. The first bit in the source address field (SSAP) shall be used to identify whether a command or response is contained in the PDU. There is an implied pairing of the sending and receiving service access points, in so much as the sender must maintain temporary state information about the receiving service access point. This pairing consists of a logical concatenation of the implicit physical address (not included in the frame structure), the MAC address (DA/SA) and the LLC address (DSAP/SSAP).

8.3 Procedure for the use of the P/F bit

LLC shall set the P bit in an ACn command PDU to “0” if the command PDU is not a request for the remote LLC to return an LSDU in its acknowledgment. Thus the P bit is set to “0” when data is to be passed only from the sending station to the receiving station, or when the command PDU is to be passed only for resynchronization. LLC shall set the P bit in an ACn command PDU to “1” if the command PDU is a request for the remote LLC to return an LSDU in its acknowledgment. Setting the P bit to “1” allows data to be passed in both directions; however, if it is desired that data pass only from the responding LLC to the sending LLC, a null information field may be placed in the command PDU. When transmitting an ACn response PDU, LLC sets the F bit equal to the P bit in the received ACn command PDU and includes a non-null LSDU subfield only if the F bit is a “1”.

8.4 Procedures for link setup and disconnection

Type 3 operation does not contain a formal procedures for data link connection establishment (setup) and hence no data link disconnection. Once the service access point has been activated within the LLC, presumably by layer management's request, information may be sent to or received from a remote LLC service access point that is also participating in Type 3 operation.

There is, however, an optional mechanism whereby the user can ensure that the one-bit receive sequence state variable, V(RI), in the remote LLC is synchronized with the transmit sequence state variable, V(SI), at the local LLC. The resynchronization procedure is analogous to logical link setup. There is also an optional procedure whereby the LLC can dispose of state information used for transmitting and receiving. This disposal procedure is analogous to link disconnection, and it overcomes the problem of having to retain state information related to all address pairs that have communicated at any time in the past.

8.4.1 Sequence number resynchronization

At any time, the network layer can pass to LLC an DL-DATA-ACK request primitive with a null LSDU. Using the normal acknowledged connectionless-mode data unit exchange service procedures, LLC will send an ACn command PDU, but since the LSDU is null, no indication will be passed to the remote network layer regardless of whether the remote LLC determines the command PDU to be a duplicate or non-duplicate. In any case, at the completion of this procedure, the V(RI) state variable at the remote LLC will be properly synchronized with the V(SI) state variable at the local LLC. It is recommended that the network layer per-

form this resynchronization procedure if local state information may have been destroyed due to the local station being powered off/on or being reset.

8.4.2 Destruction of a transmit sequence state variable, V(SI)

The V(SI) state variables may be destroyed to avoid maintaining state information indefinitely. The minimum time period before any given transmit sequence state variable can be destroyed after the variable is created or last updated is given by logical link parameter T3 (see 8.6.6).

NOTE—The period of T3 can be made infinite so that the destruction of the V(SI) state variables never occurs at any given station (except upon reset).

8.4.3 Destruction of a receive sequence state variable V(RI) and reception status state variable V(RB)

Upon expiration of the receive variable lifetime timer associated with a given variable, the V(RI) and V(RB) receive state variables shall be destroyed (declared not to exist). The receive variable lifetime timer runs for a period given by logical link parameter T2 (see 8.6.5). Further, if an event occurs that causes the integrity of the state variables to be in question (such as the station being reset or powered off/on), the V(RI) and V(RB) state variables shall be destroyed. (It is assumed that if LLC is reset, the higher communication layers are also reset or are at least aware of the reset of LLC.)

NOTE—The period of T2 can be made infinite so that the destruction of the V(RI) state variables never occurs at any given station (except upon reset); however, if T2 is infinite, T3 must also be made infinite, since T3 must be greater than T2 (see 8.6.6). Thus if the V(RI) state variables are retained indefinitely, the V(SI) state variables must also be retained indefinitely.

8.5 Procedures for information transfer

8.5.1 Sending ACn command PDUs

Information transfer from an initiating station to a responding station shall be accomplished by sending the AC0 or AC1 command. It shall be possible to send such a command PDU at a given priority at any time, to any receiving LLC provided the sending LLC is not currently awaiting an AC0 or AC1 response PDU from that LLC for the same local service access point, the same remote MAC address, and at that same MAC priority.

Upon being passed a DL-DATA-ACK request primitive from the network layer, the LLC shall send an ACn command PDU containing the specified LSDU with the P bit in the ACn command PDU set to “0”. Upon being passed a DL-REPLY request primitive from the network layer, the LLC shall send an ACn command PDU containing the specified LSDU with the P bit in the ACn command PDU set to “1”.

Whenever a new (not re-transmitted) ACn command PDU is to be sent, and there exists a transmit sequence state variable V(SI) associated with the given DA portion of the remote address, local address (SSAP), and priority, the value of V(SI) shall be used to select the control field code-point of the PDU. If V(SI) is zero, code-point AC0 is used, and if V(SI) is one, code-point AC1 is used. If the required V(SI) state variable does not already exist, it shall be created with a value of zero, and the AC0 code-point is used. The service class requested by the network layer shall be passed to the medium access control sublayer. The service class may request an acknowledged service from the medium access control sublayer if supported.

When a sending LLC sends a command PDU, it shall start an acknowledgment timer for that transmission and increment an internal transmission count variable. If no ACn response PDU is received before the timer runs out, the sending LLC shall resend the ACn command PDU, increment the internal transmission count variable, and reset and restart the acknowledgment timer. If an ACn response PDU is still not received, this

resending procedure shall be repeated until the value of the internal transmission count variable is found to equal the value of the logical link parameter N4, as described in 8.6.1, at which time an unsuccessful status shall be reported to the network layer. An internal transmission count shall be maintained for each Type 3 information exchange between a pair of sending and receiving LLCs. Both the acknowledgment timer and the internal transmission count shall not affect the Type 3 information exchange to other receiving LLCs. Type 3 information exchange shall not interfere with any Type 1 or Type 2 operation.

8.5.2 Receiving ACn command PDUs

8.5.2.1 General case

This subclause covers operations required for general medium access control sublayer (MAC) capabilities. (See 8.5.2.2. for permissible simplifications applying to a certain special case.)

Upon receipt of an AC0 or AC1 command PDU, the LLC shall check for the existence of a V(RI) receive state variable associated with the source address (SA and SSAP) and the MAC priority of the PDU. If the V(RI) exists, it is compared with bit eight of the code-point of the received PDU (“0” for AC0, or “1” for AC1). If the comparison shows equality or if the V(RI) variable did not exist, the received PDU is recognized to be a non-duplicate. If, however the comparison shows inequality, the received PDU is recognized to be a duplication of the most recently received ACn command PDU.

8.5.2.1.1 Non-duplicate ACn command PDU

For the case of a non-duplicate PDU, if the P bit is a “0” and the received LSDU in the PDU was non-null and successfully received, that LSDU shall be passed to the network layer in a DL-DATA-ACK indication primitive. If the P bit is a “1” and the appropriate reply LSDU can be accessed, a DL-REPLY indication primitive shall be passed to the network layer whether or not the LSDU is null; however, if the LSDU is not null and successfully received, it shall be passed in the indication primitive. If the P bit is a “1” and the received LSDU in the PDU was non-null and successfully received, but the appropriate reply LSDU cannot be accessed, the received LSDU shall be passed to the network layer in a DL-DATA-ACK indication primitive.

The state variable V(RI) associated with the addresses and priority of the received command PDU shall be set equal to the complement of bit eight of the code-point in the received PDU. The state variable V(RB) associated with the addresses and priority of the received command PDU shall be set to reflect the success or failure of the reception of the LSDU (if non-null) in the received PDU. (The V(RI) and V(RB) variables shall be created if they did not already exist.) The receive variable lifetime timer associated with that V(RI) and V(RB) shall be started (or reset and restarted if already running).

LLC shall acknowledge the receipt of a non-duplicate ACn command PDU by sending to the originator of the command PDU an ACn response PDU having bit eight of its control field code point set to the (new) value of the V(RI) state variable associated with the specified address pair and priority. If the P bit in the received command PDU is a “0”, the response PDU shall be sent with the F bit set to “0” and with only a status subfield in the information field (LSDU subfield is null). If the P bit in the command PDU is a “1”, the response PDU shall be sent with the F bit set to “1”, and if there is available an LSDU associated with the DSAP specified by the command PDU, the information field in the response PDU shall contain that LSDU in the LSDU subfield along with the status subfield.

8.5.2.1.2 Duplicate ACn command PDU

The LLC procedures upon the reception of a duplicate ACn command PDU are the same as those for the non-duplicate PDU with the following exceptions. The V(RI) and V(RB) state variables and the associated timer are not affected by the reception of a duplicate command PDU. The DL-DATA-ACK indication primitive is not issued, regardless of the P bit in the command PDU. If an LSDU is received in the command PDU,

it is discarded. The ACn response PDU is sent in the same manner as in the non-duplicate case, except that the status code in the CCCC portion of the status subfield (see figure 19) is set according to the reception status stored previously in the appropriate V(RB) state variable, instead of being set to show the success or failure of the reception of the current LSDU.

8.5.2.2 Simplification for a specific case

A certain medium access control (MAC) sublayer capability allows the simplification of LLC state information at the receiving end of the ACn command, while maintaining the compatibility of the peer-to-peer interaction of the protocol entities. This capability is that the MAC insures that during the period in which a first transmission and all retries of a single ACn command occur, the destination station will not receive some other intervening ACn PDU, and the MAC at the originating station services its queues for each priority supported in FIFO (first in, first out) order within that priority.

The associated simplification is that only one V(RI) state variable and one V(RB) state variables are needed (as opposed to one for each combination of remote address, and priority). The V(R) must be set equal to the complement of bit eight of the code-point of the most recent previously received ACn command PDU. The V(RB) must be set to reflect the success or failure of the reception of the LSDU in the same PDU. It is required, however, that the remote address (SSAP and SA) and priority of that most recent previously received ACn command PDU also be retained as state information. Whenever an update of V(RI) is specified, the remote address and priority state information must also be updated.

Whenever a comparison of bit eight of the code-point of the PDU with V(RI) is specified, that comparison is performed along with a comparison of the remote address and priority from the PDU with the corresponding state information. The ACn command PDU is determined to be a duplicate if and only if V(RI) is the complement of bit eight of the code-point, and the remote address and priority of the PDU agree with the corresponding state information.

8.5.3 Sending ACn response PDUs

An AC0 or AC1 response PDU shall be sent only upon the reception of an AC1 or AC0 command PDU, respectively, and shall be sent to the originator (SSAP and SA) of the associated command PDU. (Note that bit eight of the code-point in the response PDU is opposite from that in the associated command PDU.) The status subfield in the response PDU shall indicate whether or not resources were available to successfully receive the information field in the associated command PDU and, in the case of the F bit equal to “1”, whether or not an LSDU was available for return in the response PDU. If the underlying medium access control sublayer supports the “request with response” service class, the service class requested of the medium access control sublayer by LLC for the response PDU shall be determined by the service class of the associated ACn command PDU in the following manner:

Service class received with ACn command	Service class sent with ACn response
Request-with-no-Response	Request-with-no-Response
Request-with-Response	Response
Response	(Protocol Error)

NOTE—On resources unavailable. This International Standard does not specify at what sublayers buffering is done. For clarification, however, if the buffering of incoming PDUs is done by the medium access control sublayer, the reception_status parameter in the MA-UNITDATA indication primitive can be used to inform the LLC of a buffering problem in the received PDU. Regardless of where buffering is done, if a normal buffer is unavailable, one small emergency buffer could be used to hold header information from an incoming PDU long enough to allow LLC to build its response PDU.

8.5.4 Receiving acknowledgment

8.5.4.1 General case

This subclause covers operations required for general medium access control (MAC) sublayer capabilities. (See 8.5.4.2 for permissible simplifications applying to a certain special case.)

After sending an AC0 or AC1 command PDU to some remote LLC, the sending LLC shall expect to receive an acknowledgment in the form of an AC1 or AC0 response PDU, respectively, from the LLC to which the command PDU was sent. Upon receiving such a response PDU, the LLC shall compare bit eight of the code-point in that PDU with the current value of the transmit sequence state variable V(SI) associated with the PDU's SA portion of the remote address, DSAP portion of the local address, and MAC priority.

If the comparison shows inequality, the response is considered valid and the LLC shall stop the acknowledgment timer associated with the transmission for which the acknowledgment was received, and reset the internal transmission count to zero. The V(SI) state variable is then complemented. Further for a valid response, LLC shall pass a DL-DATA-ACK-STATUS indication primitive or a DL-REPLY-STATUS indication primitive to the network layer, depending on which request primitive is being confirmed. In the case that response data was requested and successfully returned in the ACn response PDU, that LSDU shall be passed to the network layer. LLC shall pass status to the network layer based on the status subfield in the response PDU.

If the bit comparison shows equality, the ACn response PDU shall be considered invalid. The LLC shall take no further action on an invalid response PDU, and shall continue to expect to receive a valid ACn response PDU. The acknowledgment timer shall not be affected by the reception of an invalid response PDU.

8.5.4.2 Simplification for a specific case

The specific case is that the MAC is able to correlate the received ACn response PDU with the appropriate previously transmitted ACn command PDU. This specific case allows the simplification of LLC procedures for receiving ACn responses, by eliminating the need for LLC to check the validity of the received response PDU. Thus LLC need not compare bit eight of the code-point of the PDU with the current value of the transmit sequence state variable V(SI). Instead, the acknowledgment is always considered valid, and the procedures for a valid acknowledgment are always performed.

8.6 List of logical link parameters

A number of logical link parameters are defined, the range of values for which are determined on a system-by-system basis by the user at the time that the local area network is established.

The logical link parameters for Type 3 operation shall be as follows:

8.6.1 Maximum number of transmissions, N4

N4 is a logical link parameter that indicates the maximum number of times that an ACn command PDU is sent by LLC trying to accomplish a successful information exchange. Normally, N4 is set large enough to overcome the loss of a PDU due to link error conditions. If the medium access control sublayer has its own retransmission capability, the value of N4 may be set to one so that LLC does not itself requeue a PDU to the medium access control sublayer.

8.6.2 Maximum number of octets in an ACn command PDU, N3

N3 is a logical link parameter that denotes the maximum number of octets in an ACn command PDU. Refer to the various MAC descriptions to determine the precise value of N3 for a given medium access method. LLC places no restrictions on the value of N3.

8.6.3 Minimum number of octets in a PDU

A minimum length valid ACn command PDU shall contain exactly two address fields and one control field in that order. Thus the minimum number of octets in a valid ACn command PDU shall be 3. A minimum length valid ACn response PDU shall contain exactly two address fields, one control field, and the status subfield in that order. Thus the minimum number of octets in a valid ACn response PDU shall be 4.

8.6.4 Acknowledgment time, T1

The acknowledgment time is a logical link parameter that determines the period of the acknowledgment timers, and as such shall define the time interval during which the LLC shall expect to receive an ACn response PDU from a specific LLC from which the LLC is awaiting a response PDU. The acknowledgment time shall take into account any delay introduced by the MAC sublayer and whether the timer is started at the beginning or at the end of the sending of the ACn command PDU by the LLC. The proper operation of the procedure shall require that the acknowledgment time be greater than the normal time between the sending of an ACn command PDU and the reception of the corresponding ACn response PDU.

If the medium access control sublayer performs its own retransmissions and if the logical link parameter N4 is set to one to prevent LLC from requeuing a PDU, then the acknowledgment time T1 may be set to infinity, making the acknowledgment timers unnecessary.

8.6.5 Receive lifetime variable, T2

This time value is a logical link parameter that determines the period of all of the receive variable lifetime timers. T2 shall be longer by a margin of safety than the longest possible period during which the first transmission and all retries of a single PDU may occur. The margin of safety shall take into account anything affecting LLCs perception of the arrival time of PDUs, such as LLC response time, timer resolution, and variations in the time required for the medium access control sublayer to pass received PDUs to LLC.

If the destruction of the received state variables is not desired, the value of time T2 may be set to infinity. In this case the receive variable lifetime timer need not be implemented.

8.6.6 Transmit lifetime variables, T3

This time value is a logical link parameter that determines the minimum lifetime of the transmit sequence state variables. T3 must be longer by a margin of safety than 1) the logical link variable T2 at stations to which ACn commands are sent, and 2) the longest possible lifetime of an ACn command-response pair. The lifetime of an ACn command-response pair must take into account the sum of processing time, queuing delays, and transmission time for the command and response PDUs at the local and remote stations.

If the destruction of the transmit state variables is not desired, the value of time T3 may be set to infinity. Note, if the receive variable lifetime parameter, T2 is set to infinity at remote stations to which ACn commands are sent, then the T3 parameter must be set to infinity at the local station.

8.7 Precise description of Type 3 procedures

If discrepancies appear to exist with the text found in the balance of clause 8, this subclause (8.7) shall be viewed as being the definitive description.

8.7.1 Type 3 receiver component

8.7.1.1 Type 3 receiver component overview

The Type 3 receiver component is responsible for receiving ACn commands from remote stations and returning the appropriate ACn response. There is one Type 3 receiver component for each unique combination of remote address (SA and SSAP), and MAC priority associated with received Type 3 command PDUs, and this component has only one state. All state information is contained in state variables. (This model is more suitable than a multi-state model, because the Type 3 operations are transaction oriented; that is, each command-response pair is essentially independent.) All operations at the responding LLC necessary for the handling of a single transaction are completed at one time interval.

Each receiver component uses its own V(RI) state variable and V(RB) state variable when checking for a duplicate command PDU and when checking the status of a previous reception. Further, there is one receive lifetime timer for each receiver component.

8.7.1.2 Type 3 receiver component state descriptions

1) **Ready:** This is the only state. LLC is capable of receiving and acknowledging Type 3 PDUs.

8.7.1.3 Type 3 receiver component function descriptions

The following function return values are used both for qualifying events and for supplying values used in actions.

1) **RECEIVE STATUS():** Returns an indication of the success or failure of the processing of the information field of the received command PDU. (It is assumed, however, that the LLC header was successfully received any time an MA-UNITDATA indication primitive is passed to LLC.)
   The possible returned values are:

   — **OK:** Information field successfully processed.

   — **UN:** Resources temporarily unavailable for information field.

   — **RS:** Reception of information is unimplemented or inactivated.

   — **UE:** Hardware failure prevents information transfer to user.

   — **IT:**Temporary implementation dependent error.

   — **IP:** Permanent implementation dependent error.

2) **ACCESS():** Returns an indication of whether or not an LSDU associated with the DSAP specified in the received command PDU is available for inclusion in a response PDU. The possible returned values are:

   — **OK:** The LSDU exists and it can be accessed quickly enough to include it in the response PDU.

   — **UN:** Resources temporarily unavailable to access the LSDU.

— **RS:** The return of an LSDU is unimplemented or inactivated.

— **NE:** Response LSDU was never submitted by user (since powerup).

— **UE:** Hardware failure prevents information transfer from user.

— **IT:** Temporary implementation dependent error.

— **IP:** Permanent implementation dependent error.

8.7.1.4 Type 3 receiver component event descriptions

1) **REPLY_UPDATE_REQUEST:** The network layer has passed a DL_REPLY_UPDATE request primitive to LLC.
2) **RECEIVE_ACn_CMD(SQC, P, INFO):** The medium access control sublayer has passed to LLC an MA-UNITDATA indication primitive containing an AC0 or AC1 command PDU, where the command sequence bit SQC (bit eight of the code-point) is “0” for an AC0 command or “1” for an AC1 command. The following parameter values exist for this event:

   — **SQC=V(RI):** Either the command sequence bit is equal to the V(RI) state variable for this receiver component, or that state variable does not exist.

   — **SQC<>V(RI):** There exists a V(RI) state variable for this receiver component and the command sequence bit is not equal to that state variable.

   — **P=0:** The P bit in the command is a “0”.

   — **P=1:** The P bit in the command is a “1”.

   — **INFO=NULL:** The information field in the command is null (of zero length).

   — **INFO<>NULL:** The information field in the command is not null.

3) **RCV_LIFE_TIMER_EXPIRED:** The receive variable lifetime timer for this receiver component has expired.

In the state transition table, some of the events are qualified by the following conditions. The event is recognized only when the condition is true.

1) **RECEIVE_STATUS()=OK:** The information field in the received command PDU was successfully received and can be passed to the network layer.
2) **RECEIVE_STATUS()<>OK:** The information field in the received command PDU was not successfully received or cannot be passed to the network layer.
3) **ACCESS()=OK:** A response LSDU associated with the LSAP does exist and it can be accessed quickly enough to include it in the response PDU.
4) **ACCESS()<>OK:** Either a response LSDU associated with the LSAP does not exist or the LSDU does exist but it cannot be accessed quickly enough to include it in the response PDU.

8.7.1.5 Type 3 receiver component action descriptions

1) **SAVE:=GIVEN_LSDU:** The LSDU given in the associated DL-REPLY-UPDATE request primitive is held in readiness for transmission by being placed in the abstract location, SAVE. The SAVE location used is specifically associated with the LSAP given in the primitive and the new LSDU replaces any previously held for that LSAP.

Table 5—Type 3 receiver component state transition table

Current state	Event	Action(s)	Next state
READY	REPLY_UPDATE_REQUEST	SAVE:=GIVEN_LSDU REPLY_UPDATE_STATUS_ INDICATION	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=0, INFO<>NULL) and RECEIVE_STATUS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=0, C=OK, R=NR, LSDU=NULL) DATA_ACK_INDICATION V(RI):=1-SQC V(RB):=OK START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=O, INFO=NULL) and RECEIVE_STATUS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=0, C=OK, R=NR, LSDU=NULL) V(RI):=1-SQC V(RB:=OK START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=0) and RECEIVE_STATUS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=0, C=RECEIVE_STATUS(), R=NR, LSDU=NULL) V(RI):=1-SQC V(RB):=RECEIVE_STATUS() START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=1) and RECEIVE_STATUS()=OK and ACCESS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=OK, R=OK, LSDU=SAVE) REPLY_INDICATION(LSDU=INF0) V(RI):=1-SQC V(RB):=OK START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=1) and RECEIVE_STATUS()<>OK and ACCESS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=RECEIVE STATUS(), R=OK, LSDU=SAVE) REPLY_INDICATION(LSDU=NULL) V(RI):=1-SQC V(RB):==RECEIVE_STATUS() START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=1, INFO<>NULL) and RECEIVE_STATUS()=OK and ACCESS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=OK, R=ACCESS(), LSDU=NULL) DATA_ACK_INDICATION V(RI):=1-SQC V(RB):=OK START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=1, INFO=NULL) and RECEIVE_STATUS()=OK and ACCESS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=OK, R=ACCESS(), LSDU=NULL) V(RI):=1-SQC V(RB):=OK START_RCV_LIFE_TIMER	READY

Table 5—Type 3 receiver component state transition table (*Continued*)

Current state	Event	Action(s)	Next state
READY *(Con'd.)*	RECEIVE_ACn_CMD(SQC=V(RI), P=1) and RECEIVE_STATUS()<>OK and ACCESS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=RECEIVE_STATUS(), R=ACCESS(), LSDU=NULL) V(RI):=1-SQC V(RB):=RECEIVE_STATUS() START_RCV_LIFE_TIMER	READY
	RECEIVE_ACn_CMD(SQC<>V(RI), P=0)	SEND_ACn_RSP(SQR=1-SQC, F=0, C=V(RB), R=NR, LSDU=NULL)	READY
	RECEIVE_ACn_CMD(SQC<>V(RI), P=1 and ACCESS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=V(RB), R=OK, LSDU=SAVE) REPLY_INDICATION(LSDU=NULL)	READY
	RECEIVE_ACn_CMD(SQC<>V(RI), P=1) and ACCESS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=V(RB), R=ACCESS(), LSDU=NULL)	READY
	RCV_LIFE_TIMER_EXPIRED	DESTROY_V(RI)	READY

2) **SEND_ACn_RSP(SQR, F, C, R, LSDU):** Pass an MA-UNITDATA request primitive to the medium access control sublayer containing an AC0 or AC1 response PDU. The following parameter values exist for this action.

— **SQR=1-SQC:** The response sequence bit (bit eight of the code-point) is set to the complement of the sequence bit from the received command.

— **F=0:** The F bit of the response is set to "0".

— **F=1:** The F bit of the response is set to "1".

— **C=OK:** The CCCC portion of the status subfield (see figure 19) is set to the "OK" code (successful reception).

— **C=RECEIVE_STATUS():** The CCCC portion of the status subfield is set to the value returned by the RECEIVE_STATUS function.

— **C=V(RB):** The CCCC portion of the status subfield is set equal to the V(RB) state variable associated with the source address and priority of the received command PDU.

— **R=NR:** The RRRR portion of the status subfield is set to the "NR" code (response data not requested).

— **R=OK:** The RRRR portion of the status subfield is set to the "OK" code (response data included).

— **R=ACCESS():** The RRRR portion of the status subfield is set to the value returned from the ACCESS function.

— **LSDU=NULL:** The LSDU subfield of the response is null (of zero length).

— **LSDU=SAVE:** The LSDU subfield of the response contains the LSDU held in readiness in the SAVE location for this LSAP.

3) **DATA_ACK_INDICATION:** Pass to the network layer a DL-DATA-ACK indication primitive containing a LSDU equal to the information field from the associated received command PDU.
4) **REPLY_INDICATION(LSDU):** Pass to the network layer a DL-REPLY indication primitive. The following parameter values exist for this action:

— **LSDU=INFO:** The network layer is passed an LSDU equal to the information field from the associated received command PDU. (This field may be null.)

— **LSDU=NULL:** The network layer is passed a null LSDU.

5) **REPLY_UPDATE_STATUS_INDICATION:** Pass to the network layer a DL-REPLY-UPDATE-STATUS indication primitive.
6) **START_RCV_LIFE_TIMER:** Start the receive variable lifetime timer for this receiver component. (If the timer is already running, reset and restart it.).
7) **V(RI):=1-SQC:** The V(RI) state variable for this receiver component is set to the complement of the sequence bit (bit eight of the code-point) in the received command PDU.
8) **V(RB):=OK:** The V(RB) state variable for this receiver component is set to the "OK" code (successful reception).
9) **V(RB):=RECEIVE_STATUS():** The V(RB) state variable for this receiver component is set to the value returned by the RECEIVE_STATUS function.
10) **DESTROY_V(RI):** Destroy (declare not to exist) the V(RI) and V(RB) state variables for this receiver component.

8.7.2 Type 3 sender component

8.7.2.1 Type 3 sender component overview

The Type 3 sender component is responsible for sending ACn command PDUs to a remote LLC. The sender component also receives response PDUs, and resends the command PDUs if no response is received. The Type 3 protocol allows one outstanding (not yet acknowledged) command PDU for each combination of SSAP, DA portion of the remote address, and MAC priority. In order to model that restriction, a separate Type 3 sender component exists for each of these cases. Each sender component uses its own V(SI) state variable when selecting the code-point for a new transmission and when checking for a valid response code-point.

Each sender component has three states. In the IDLE state, it is capable of processing a request primitive from the network layer to send a new command PDU. In the WAIT_A and WAIT_R states, the sender component is only capable of receiving a response from the remote LLC, or of timing out and performing a retransmission. The WAIT_A state is used when the expected response is a data-less acknowledgment, and the WAIT_R state is used when the expected response is a data-bearing reply.

It is assumed that implicit context information passed across the interface between LLC and the MAC sublayer allows LLC to relate an MA-UNITDATA-STATUS indication primitive to the appropriate previous MA-UNITDATA request primitive. Only an MA-UNITDATA-STATUS indication primitive associated with the most recent MA-UNITDATA request primitive issued by a given sender component is acted upon. Any others received from the MAC are ignored.

8.7.2.2 Type 3 sender component state descriptions

1) **IDLE:** In this state, LLC is capable of executing a request from the network layer to send a Type 3 command PDU.

2) **WAIT_A:** In this state, LLC is waiting for an acknowledgment of a previously sent Type 3 command PDU that was invoked by a DL-DATA-ACK request primitive.
3) **WAIT_R:** In this state, LLC is waiting for an acknowledgment of a previously sent Type 3 command PDU that was invoked by a DL-REPLY request primitive.

8.7.2.3 Type 3 sender component event descriptions

1) **DATA_ACK_REQUEST:** The network layer has passed a DL-DATA-ACK request primitive to the LLC.
2) **REPLY_REQUEST:** The network layer has passed a DL-REPLY request primitive to the LLC.
3) **RECEIVE_ACn_RSP(SQR, R, LSDU):** The MAC sublayer has passed to LLC an MA-UNITDATA indication primitive containing an AC0 or AC1 response PDU, where the response sequence bit SQR (bit eight of the code-point) is "0" for an AC0 response or "1" for an AC1 response. The following parameter values exist for this event:
   - **SQR=V(SI):** The response sequence bit is equal to the V(SI) state variables for this sender component.
   - **SQR<>V(SI):** The response sequence bit is not equal to the V(SI) state variable for this sender component.
   - **R=OK:** The RRRR portion of the status subfield (see figure 19) of the received response PDU shows the "OK" status (indicating that an LSDU is included).
   - **R<>OK:** The RRRR portion of the status subfield of the received response PDU shows a status other than "OK" (indicating that an LSDU is not included).
   - **LSDU=NULL:** The LSDU subfield in the response is null (of zero length).
   - **LSDU<>NULL:** The LSDU subfield in the response is not null.
4) **MA_UNITDATA_STATUS:** The MAC sublayer has passed to LLC an MA-UNITDATA-STATUS indication primitive that confirms the status of the most recent MA-UNITDATA request primitive passed by this sender component to the MAC.
5) **ACK_TIMER_EXPIRED:** The acknowledgment timer associated with this sender component (i.e., the timer for a specific address pair and priority) has expired.
6) **TX_LIFE_TIMER_EXPIRED:** The transmit variable lifetime for this sender component has expired.

In the state transmission table, some of the events are qualified by the following conditions. The event is recognized only when the condition is true.

7) **RETRY_COUNT<N4:** The retry count value for this sender component is less than the logical link parameter N4.
8) **RETRY_COUNT>=N4:** The retry count value for this sender component is greater than or equal to the logical link parameter N4.
9) **TRANSMISSION_STATUS=GOOD:** The transmission_status parameter passed in the associated MA-UNITDATA-STATUS indication primitive shows the transmission to be successful.
10) **TRANSMISSION_STATUS=BAD:** The transmission_status parameter passed in the associated MA-UNITDATA-STATUS indication primitive shows the transmission to be unsuccessful.

8.7.2.4 Type 3 sender component descriptions

1) **SEND_ACn_CMD(SQC, P):** Pass an MA-UNITDATA request primitive containing an AC0 or AC1 command PDU to the MAC sublayer. The following parameter values exist for this action:
   - **SQC=V(SI):** Set the command sequence bit (bit eight of the code-point) equal to the V(SI) state variable for this sender component. If that V(SI) state variable not does exist, create it with a value of zero; otherwise use the current value.
   - **P=0:** The P bit of the response is set to "0".
   - **P=1:** The P bit of the response is set to "1".

2) **RE-SEND_OLD_CMD:** Pass an MA-UNITDATA request primitive containing the ACn command PDU most recently sent by this sender component to the MAC sublayer.
3) **START_ACK_TIMER:** Start the acknowledgment timer for this sender component (i.e., the timer for a specific address pair and priority).
4) **CANCEL_ACK_TIMER:** Cancel the acknowledgment timer for this sender component.
5) **START_TX_LIFE_TIMER:** Start the transmit variable lifetime timer for this sender component. (If the timer is already running, reset and restart it.)
6) **CANCEL_TX_LIFE_TIMER:** Cancel the transmit variable lifetime timer that is responsible for the transmit variable lifetime.
7) **DATA_ACK_STATUS_INDICATION(STATUS):** Pass to the network layer a DL-DATA-ACK-STATUS indication primitive. The following parameter values exist for this action:
   - **STATUS=TRANSMISSION_STATUS:** The status parameter passed to the network layer is set according to the transmission_status parameter passed to LLC in the associated MA-UNIT-DATA-STATUS indication primitive.
   - **STATUS=UNSUCCESSFUL:** The status parameter is set to indicate failure to receive an acknowledgment.
   - **STATUS=STATUS_SUBFIELD:** The status parameter is set according to the status returned in the received response PDU.
8) **REPLY_STATUS_INDICATION(STATUS, LSDU):** Pass to the network layer a DL-REPLY-STATUS indication primitive. The following parameter values exist for this action:
   - **STATUS=TRANSMISSION_STATUS:** The status parameter passed to the network layer is set according to the transmission_status parameter passed to LLC in the associated MA-UNIT-DATA-STATUS indication primitive.
   - **STATUS=UNSUCCESSFUL:** The status parameter is set to indicate failure to receive an acknowledgment.
   - **STATUS=STATUS_SUBFIELD:** The status parameter is set according to the status returned in the received response PDU.
   - **STATUS=PE:** The status parameter is set to the PE status (protocol error).
   - **LSDU=NULL:** The data parameter is null.
   - **LSDU=GIVEN_LSDU:** The data parameter contains the LSDU given in the associated MA-UNITDATA indication primitive.
9) **V(SI):=1-V(SI):** Complement the V(SI) state variable for this sender component.
10) **RETRY_COUNT:=0:** Set the retry counter for this sender component to zero.
11) **RETRY_COUNT:=RETRY_COUNT+1:** Increment the retry counter for this sender component.
12) **REPORT_STATUS(ILLEGAL_LSDU):** Report to layer management that an LSDU was received in violation of the Type 3 LLC protocol.
13) **DESTROY_V(SI):** Destroy (declare not to exist) the V(SI) state variable for this sender component.

Table 6—Type 3 sender component state transition table

Current state	Event	Action(s)	Next state
IDLE	MA_UNITDATA_STATUS or RECEIVE_ACn_RSP		IDLE
	DATA_ACK_REQUEST	CANCEL_TX_LIFE_TIMER SEND_ACn_CMD(SQC=V(SI), P=0) START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	WAIT_A
	REPLY_REQUEST	CANCEL_TX_LIFE_TIMER SEND_ACn_CMD(SQC=V(SI), P=1) START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	WAIT_R
	TX_LIFE_TIMER_EXPIRED	DESTROY_V(SI)	(IDLE) (See Note)
WAIT_A	RECEIVE_ACn_RSP(SQR<>V(SI), LSDU=NULL)	DATA_ACK_STATUS_INDICATION(STATUS=STATUS_SUBFIELD) CANCEL_ACK_TIMER RETRY_COUNT:=0 V(SI):=1-V(SI) START_TX_LIFE_TIMER	IDLE
	RECEIVE_ACn_RSP(SQR<>V(SI), LSDU<>NULL)	DATA_ACK_STATUS_INDICATION(STATUS=PE) CANCEL_ACK_TIMER RETRY_COUNT:=0 V(SI):=1-V(SI) REPORT_STATUS(ILLEGAL_LSDU) START_TX_LIFE_TIMER	IDLE
	RECEIVE_ACn_RSP(SQR=V(SI)) or MA_UNITDATA_STATUS and TRANSMISSION_STATUS=GOOD		WAIT_A
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=BAD	DATA_ACK_STATUS_INDICATION(STATUS=TRANSMISSION_STATUS) CANCEL_ACK_TIMER RETRY_COUNT:=0 START_TX_LIFE_TIMER	IDLE
	ACK_TIMER_EXPIRED and RETRY_COUNT<N4	RE-SEND_OLD_CMD START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	WAIT_A
	ACK_TIMER_EXPIRED and RETRY_COUNT>=N4	DATA_ACK_STATUS_INDICATION(STATUS=UNSUCCESSFUL) RETRY_COUNT:=0 START_TX_LIFE_TIMER	IDLE

Table 6—Type 3 sender component state transition table (*Continued*)

Current state	Event	Action(s)	Next state
WAIT_R	RECEIVE_ACn_RSP(SQR<>V(SI), R=OK)	REPLY_STATUS_INDICATION(STATUS=STATUS_SUBFIELD, LSDU=GIVEN_LSDU) CANCEL_ACK_TIMER RETRY_COUNT:=0 V(SI):=1-V(SI) START_TX_LIFE_TIMER	IDLE
	RECEIVE_ACn_RSP(SQR<>V(SI), R<>OK)	REPLY_STATUS_INDICATION(STATUS=STATUS_SUBFIELD, LSDU=NULL) CANCEL_ACK_TIMER RETRY_COUNT:=0 V(SI):=1-V(SI) START_TX_LIFE_TIMER	IDLE
	RECEIVE_ACn_RSP(SQR=V(SI)) or MA_UNITDATA_STATUS and TRANSMISSION_STATUS=GOOD		WAIT_R
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=BAD	REPLY_STATUS_INDICATION(STATUS=TRANSMISSION_STATUS, LSDU=NULL) CANCEL_ACK_TIMER RETRY_COUNT:=0 START_TX_LIFE_TIMER	IDLE
	ACK_TIMER_EXPIRED and RETRY_COUNT<N4	RE-SEND_OLD_CMD START_ACK_TIMER RETRY_COUNT:=RETRY_COUNT+1	WAIT_R
	ACK_TIMER_EXPIRED and RETRY_COUNT>=N4	REPLY_STATUS_INDICATION(STATUS=UNSUCCESSFUL, LSDU=NULL) RETRY_COUNT:=0 START_TX_LIFE_TIMER	IDLE
NOTE—When the transmit variable lifetime timer expires (TX_LIFE_TIMER_EXPIRED), the next state is not required as the transmit component does not exist.			

9. LLC RDE procedures

9.1 Overview of RDE

9.1.1 Overview of source routing and route determination

Source Routing Transparent (SRT) bridging connects multiple network segments into a single bridged network. SRT bridging allows the source to specify the path (bridged route) that a frame will take through the bridged network by specifying the particular segments in the routing information field of the frame. The Route Determination Entity (RDE) uses this MAC service to discover the route and uses the discovered route for transferring data. The SRT bridge allows four types of routing through the bridged network for each frame. The classifications are Specifically Routed Frames (SRF), Spanning Tree Explorer (STE) frames, Non-Source Routed (NSR) frames, and All Routes Explorer (ARE) frames.

The SRF takes the route specified in the routing field and only appears on those specified segments. The STE frame is forwarded through the network on the spanning tree path, but is forwarded by the rules for SRT bridges (note that a bridge that does not support source routing will not forward STE frames). The NSR frame traverses the network on the spanning tree path according to the rules for transparent bridging. The ARE frame is forwarded through the bridged network such that a copy of the frame is sent through each unique path. Both the STE and the ARE build the route in the frame's routing field such that the receiving station knows the specific path that was taken.

9.1.2 RDE system structure

The LLC uses the MAC services to transfer data between stations. Part of the MAC service includes the capability to specify the routing field that SRT MAC bridges use to route traffic through the bridged LAN. The RDE uses this MAC service to discover the route and uses the discovered route for transferring data. The system structure is shown in figure 29.

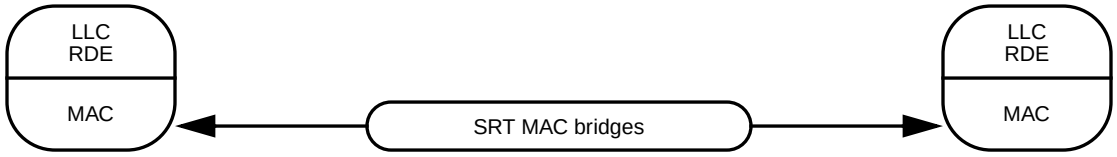


Figure 29—System structure

9.2 Support of the LLC service

9.2.1 Service to the LLC

RDE functions (in conjunction with the other LLC functions) determine a path selection through the bridged LAN. Each PDU sent from the LLC is provided with a routing field that selects the specific path the PDU will take through the bridged network.

9.2.2 Preservation of the LLC service

The RDE does not impact the operation of the other LLC functions. When a PDU is to be sent by the LLC entity, the RDE will supply the routing information. The RDE is not dependent on the state of the other functions, and the other functions are not knowledgeable of the presence of RDE.

9.2.3 Quality of service

9.2.3.1 Service availability

RDE uses the spanning tree path as the default when a specific path is not known. The spanning tree path is the same path that would be used if the RDE were not present.

9.2.3.2 Frame loss

RDE reduces frame loss. When the spanning tree reconfigures, the transparent bridging is interrupted while the bridges learn the new traffic patterns. Source-routing bridges continue to forward SRFs during these periods.

9.2.3.3 Frame misordering

RDE provides a HOLD policy to prevent the RDE from sending a PDU on a faster route while a previous PDU is using a slower route.

9.2.3.4 Frame duplication

The ARE routing is only used for route discovery. PDUs are sent using either a specific route or the spanning tree.

9.2.3.5 Path latency

The RDE requires all routes not to exceed a maximum latency time specified by system management.

9.2.3.6 Maximum service data unit size

Maximum PDU size is decreased by a maximum of 30 octets when source routing is utilized. The use of source route discovery provides the value of maximum PDU size. Routes may be selected for their capacity, and routes will be rejected that do not meet the minimum capacity specified by system management.

9.2.3.7 Throughput

The use of source routing allows resource sharing and thus increases network utilization.

9.2.4 Internal sublayer service within the LLC

9.2.4.1 Service for other LLC functions

The RDE provides a service interface between the LLC operation and the MAC sublayer. For each SEND PDU action, the RDE will provide the routing information field and generate the MA_UNITDATA request primitive to the MAC sublayer. For each MA_UNITDATA indication primitive from the MAC sublayer, the RDE will extract the routing information and forward the PDU for further processing to the LLC sublayer. (See figure 30.)

9.2.4.2 Service between RDE components

There are two components defined for the RDE—the route control component (RCC) and the route determination component (RDC). The RCC passes protocol data units (PDUs) between the LLC and the MAC, stripping routing information from incoming PDUs and adding routing information (when available) to outgoing PDUs. The RDC maintains the routing information for a particular data link through the exchange of route discovery PDUs with a partner RDC.

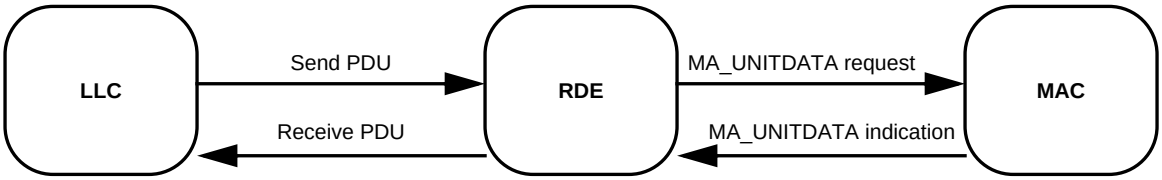


Figure 30—RDE service interfaces

There are four services defined between the RDC and RCC—routing information service, route learning service, RDE PDU transfer service, and the route reset service. These services and their corresponding service primitives and parameters are defined in the following subclauses.

9.2.4.2.1 Routing information service

The routing information service provides the following primitives to allow for the transfer of routing information between RDE components:

— GET_ROUTE request
— GET_ROUTE confirm

The GET_ROUTE request primitive is passed from the RCC to the RDC to request routing information for the specified local and remote LSAP.

The GET_ROUTE confirm primitive is passed from the RDC to the RCC with the requested routing information. The detailed specifications of these service primitives are given below.

1) GET_ROUTE request

   a) *Function:* This primitive is the service request primitive for routing information.

   b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

      GET_ROUTE request (
      source_address,
      destination_address
)

      The source_address and destination_address parameters specify the local and remote LSAPs for which routing information is being requested.

   c) *When generated:* RCC receives a request to send a frame.

   d) *Effect on receipt:* RDC returns the routing information. If the routing information is not available, the RDC returns a value of null indicating that the PDU should be sent using the spanning tree path.

2) GET_ROUTE confirm

   a) *Function:* This primitive is the service confirm primitive for the route information service.

b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

GET_ROUTE confirm (
source_address,
destination_address,
routing_information
)

The source_address and destination_address parameters specify the local and remote LSAPs for which routing information is being requested. The routing_information parameter contains the information to create the routing_information parameter of the MA_UNITDATA_request primitive.

c) *When generated:* RDC receives a request for routing information.

d) *Effect on receipt:* RCC adds the routing_information parameter to the MA_UNITDATA_request primitive.

9.2.4.2.2 Route learning service

The route learning service provides the following primitive to allow the RDE components to learn new routes:

— RIF indication

The RIF indication primitive is passed from the RCC to the RDC to indicate the routing information of a received MSDU. The detailed specifications of this service primitive are given below.

1) RIF indication

a) *Function:* This is the service indication primitive for the route learning service.

b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

RIF indication (
source_address,
destination_address,
routing_information
)

The source_address and destination_address parameters specify the local and remote LSAPs for which routing information is being requested. The routing_information parameter contains the information from the routing_information parameter of the received MA_UNITDATA indication. A null value indicates NSR.

c) *When generated:* RCC receives a PDU with an individual address as the destination MAC address.

d) *Effect on receipt:* RDC obtains routing information provided by the received PDU.

9.2.4.2.3 RDE PDU transfer service

The RDE PDU transfer service provides the following primitives to allow for the transfer of RDE PDUs between the RDE components:

— RDE_PDU request
— RDE_PDU indication

The RDE_PDU request primitive is passed from the RDC to the RCC to request that an RDE PDU be generated. The RDE_PDU indication primitive is passed from the RCC to the RDC to indicate that an RDE PDU has been received. The detailed specifications of these service primitives are given below.

1) RDE_PDU request

   a) *Function:* This is the service request primitive for the RDE PDU transfer service.

   b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

```
RDE_PDU request     (
                    source_address,
                    destination_address,
                    rde_pdu_type,
                    routing_information
                    )
```

   The source_address and destination_address parameters specify the local and remote LSAPs for which routing information is being indicated. The rde_pdu_type will indicate a route query command (RQC) or a route selected (RS) PDU. The routing_information parameter contains the information used to create the routing_information parameter of the MA_UNITDATA_request primitive.

   c) *When generated:* The RDC initiates or completes the route discovery process.

   d) *Effect on receipt:* The RCC will create and send the requested PDU.

2) RDE_PDU indication

   a) *Function:* This is the service indication primitive for the RDE PDU transfer service.

   b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

```
RDE_PDU indication  (
                    source_address,
                    destination_address,
                    rde_pdu_type,
                    routing_information
                    )
```

   The source_address and destination_address parameters specify the local and remote LSAPs for which routing information is being indicated. The rde_pdu_type will indicate an RQC, a route query response (RQR), or an RS PDU. The routing_information parameter contains the received routing information.

c) *When generated:* The RCC receives an MA_UNITDATA indication primitive addressed to the RDE LLC address.

d) *Effect on receipt:* The RDC uses the information for its route selection.

9.2.4.2.4 Route reset service

The route reset service provides the following primitive to allow for the resetting of routing information:

— RESET_ROUTE request

The RESET_ROUTE request primitive is passed from the RCC to the RDC to request that routing information be flushed and the route selection process be initiated.

The detailed specifications of this service primitive are given below.

1) RESET_ROUTE request

a) *Function:* This is the service request primitive for the route reset service.

b) *Semantics of the service primitive:* The primitive shall provide parameters as follows:

```
RESET_ROUTE request     (
                        source_address,
                        destination_address
                        )
```

The source_address and destination_address parameters specify the local and remote LSAPs for which route reset is being requested.

c) *When generated:* The RCC observes selected conditions/events that indicate that the route is no longer valid.

d) *Effect on receipt:* RDC will flush the routing information and initiate route discovery.

9.3 Principles of operation

9.3.1 Route control operation

The RCC provides routing information by intercepting the LLC send request and generating a GET_ ROUTE request primitive to the specific RDC. The RDC responds with a GET_ROUTE confirm primitive providing the routing_information parameter to be used in the MA_UNITDATA request primitive. When a PDU is received, the RCC extracts the route information, forwards the PDU for LLC processing, and indicates the route to the appropriate RDC.

Upon receipt of an RDE_PDU primitive, the RCC will generate an RDE PDU, which is sent to the RCC of the remote station. When an RDE PDU is received from a remote station, it is indicated to the appropriate RDC, via an RDE_PDU indication primitive. PDUs addressed to the RDE LLC address are not forwarded for LLC processing.

9.3.2 Station architecture

The RDE is a service entity within the LLC sublayer. Its function is to determine the routing information for each required data link, and to supply this information for each PDU sent from the LLC. The RDE is required to add this routing information to any sent PDUs and to remove any routing information from received PDUs. Furthermore, the RDE will reestablish new routing information for a given data link when required. (See figure 31.)

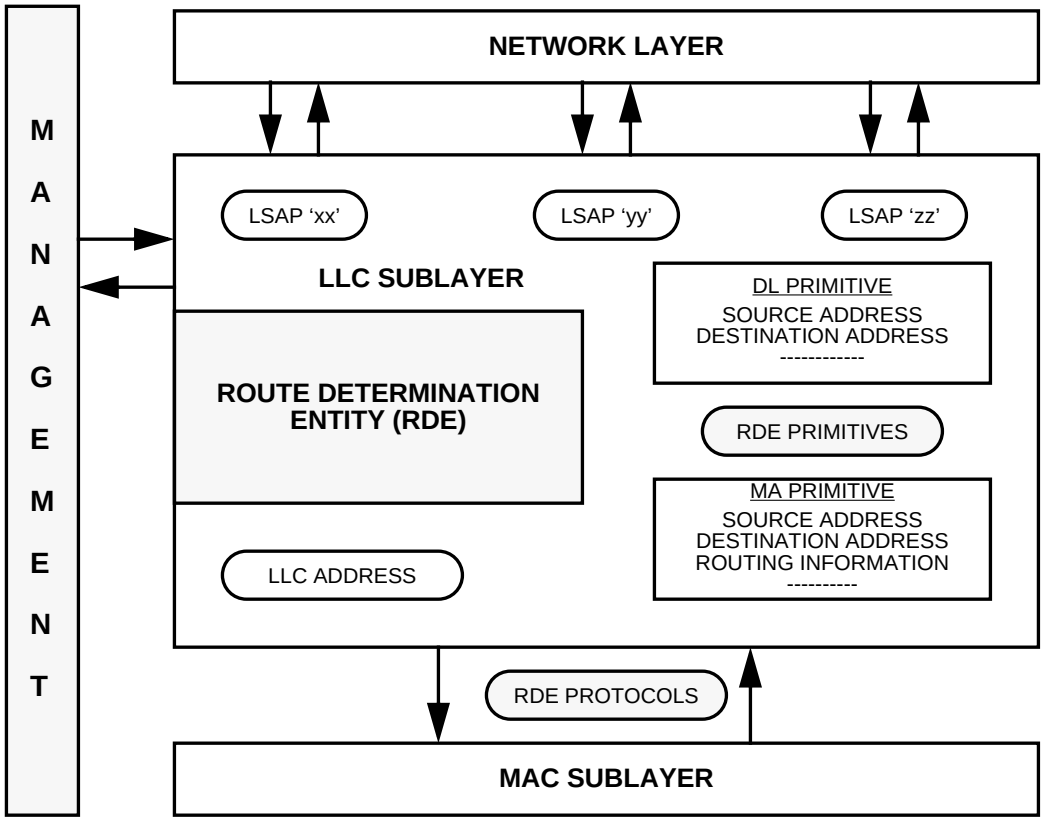


Figure 31—Station architecture

9.3.3 Model of operation

The RDE is composed of a single RCC, and for each data link that supports RDE, an RDC. The RCC will process the PDUs being sent and received by the LLC and invoke the services of the appropriate RDC to determine and apply a route for each data link. (See figure 32.)

9.3.4 PDU transmission

When an LLC SEND action is executed, the RCC will intercept the send request and determine if the PDU is to be sent using the spanning tree or a specific route. If a specific route is needed, the RCC will invoke the GET_ROUTE request primitive to the RDC. The RDC will respond with a GET_ROUTE confirm primitive specifying the route. The RCC will then invoke the MA_ UNITDATA request primitive to send the PDU.

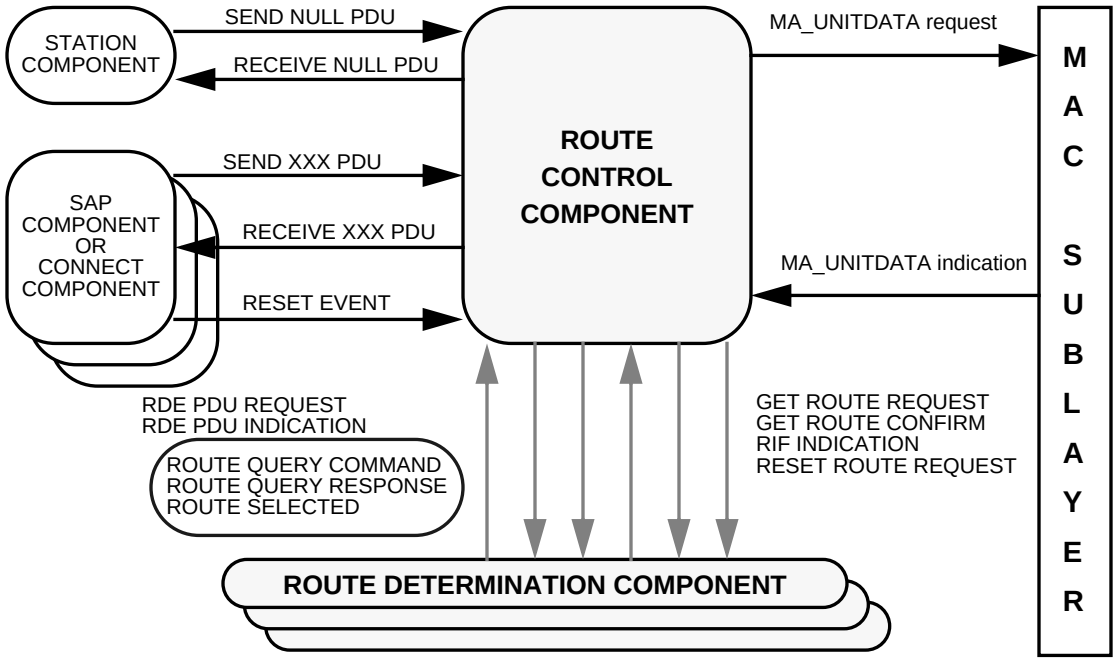


Figure 32—Model of operation

9.3.5 PDU reception

When an MA_UNITDATA indication primitive occurs, the RCC will remove the route information and forward the received PDU to the LLC for normal processing. If the PDU was received with a specific route, that route is indicated to the appropriate RDC.

9.3.6 The learning process

When a route is not known for a particular data link, the RDC will initiate the route discovery process by requesting the RCC to send an RQC PDU to the remote station. For each RQR PDU received, the RCC will notify the appropriate RDC. The RDC will select the path based on criteria supplied by system management. The RDC will request the RCC to notify the remote station of the selected route by sending an RS PDU. The selected route will be used for transmission of all subsequent PDUs to that data link.

Each station is required to respond to the RQC PDU with an RQR PDU. A station may also learn and use the route selected by its peer (remote station) either from the RS PDU or from the actual use of the route on received PDUs.

9.3.7 The protocol

9.3.7.1 LSAP value for RDE

RDE has been assigned its own LLC address, which may be used for the purpose of performing route determination. An RDE PDU uses the RDE LLC address for both source and destination. Thus RDE PDUs are sent only from the RDE LLC address to the RDE LLC address of the remote station.

9.3.7.2 RDE PDUs

The RDE makes use of three PDUs, encoded as unnumbered information (UI) frames.

1) **ROUTE_QUERY_COMMAND:** This PDU is sent from the station originating (originating station) the route discovery process to the remote station (target station). This PDU notifies the target that route discovery is being performed and solicits the target to send a ROUTE_QUERY_RESPONSE PDU. The ROUTE_QUERY_COMMAND PDU is sent on the spanning tree route (NSR or STE).
2) **ROUTE_QUERY_RESPONSE:** This PDU is sent from the target to the station originating the route discovery. The response is returned using the ARE route type. One response will be received by the originating station for each unique path through the bridged LAN.
3) **ROUTE_SELECTED:** This PDU is sent from the station that originated the route discovery to the target station using the selected path to notify the target that a route has been selected.

9.3.7.3 Sample protocol flow

Figure 33 shows a typical flow during route discovery. (1) The LLC in station A sends a data frame; since no route has been determined, it is sent on the spanning tree and station A sends a route query PDU (3) to discover the best route. Station B responds to the RQC (3) with an RQR (4) sent all routes. When station A selects the route, it sends an RS PDU (6) to station B and uses that route for data frames (7). Station B uses the route selected by station A for all of its data frames (8) sent to station A.

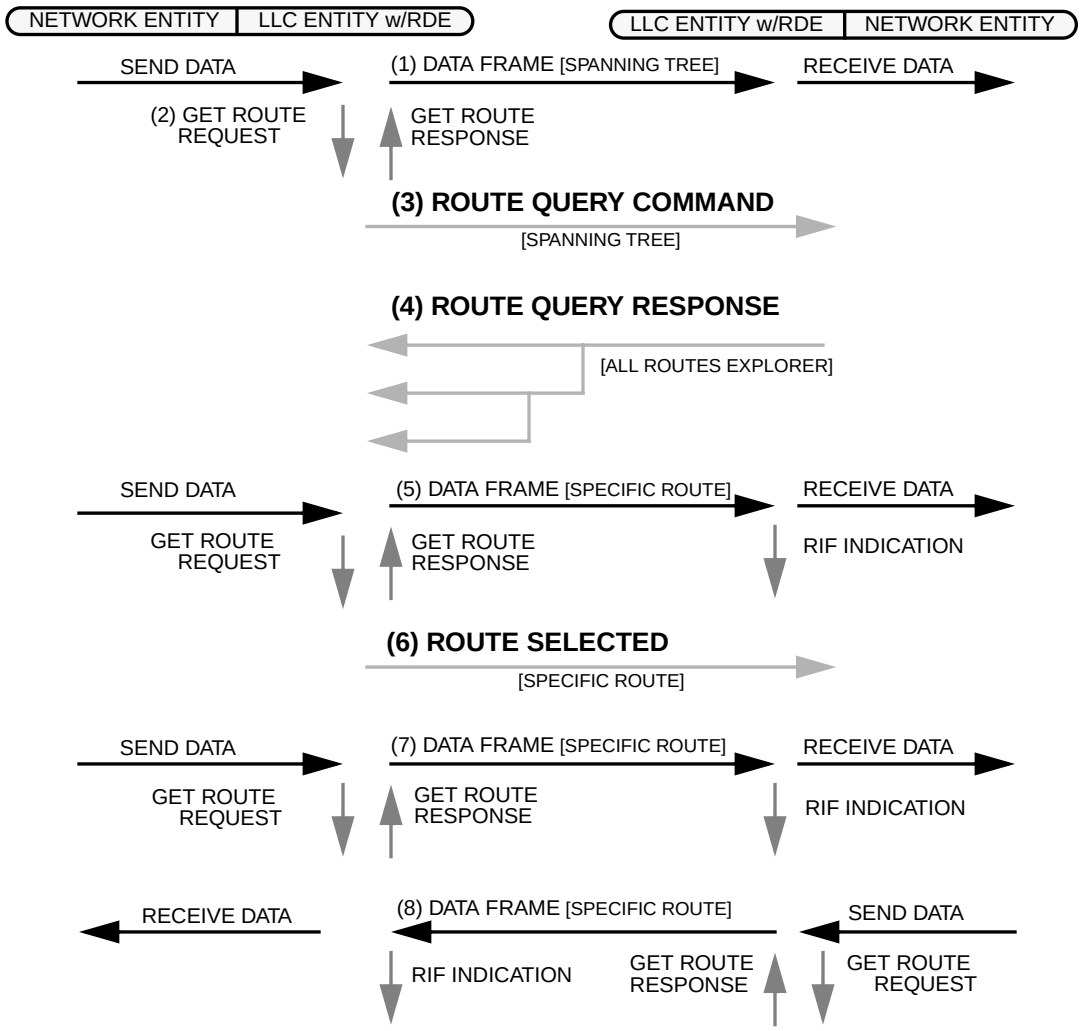


Figure 33—Sample data flow

9.4 Encoding of RDE PDUs

The PDUs used for route discovery shall be LLC Type 1 UI data frames (DSAP=RDE LLC address, I/G=0, SSAP=RDE LLC address, C/R=0, CONTROL=11000000) with the information field encoded as shown in figure 34 and described below.

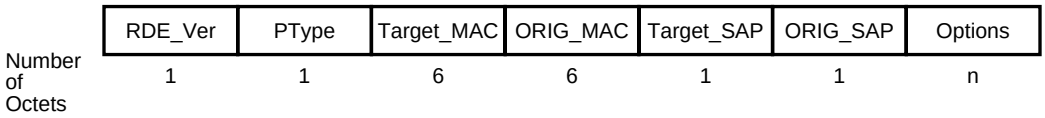


Figure 34—RDE PDU information field format

RDE_Ver indicates the version of the RDE protocol. For this version, the value of RDE_Ver shall be 1 (encoded as 10 000 000) for all RDE PDU types sent. Accommodations for subsequent RDE protocol versions is a subject for further study.

PType indicates the RDE PDU type. PType shall have the following encoding:

Value	PDU type
10000000	Route query command (RQC)
01000000	Route query response (RQR)
11000000 *lsb* |_____| *msb*	Route selected command (RSC)

Reception of an RDE PDU with any other PType value shall be ignored.

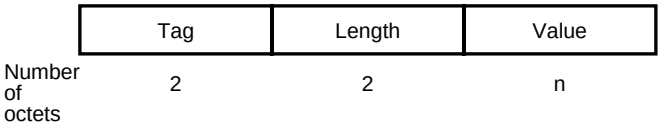
Target_MAC is the MAC address of the target station (lsb of the first byte is the MAC I/G bit).

ORIG_MAC is the MAC address associated with the station that originated the request (lsb of the first byte is the MAC I/G bit). The PDU shall be ignored if the MAC I/G bit is not 0.

Target_SAP is the LLC address of the target SAP. The LSB is the I/G bit.

ORIG_SAP is the LLC address associated with the RDC that originated the request. The LSB is the I/G bit.

Options is reserved for option fields, and employs the Tag-Length-Value encoding as pictured below.



NOTE—Any PDU addressed to the RDE LLC address shall be discarded if
1) It is not a UI PDU.
2) The ORIG_MAC is not an individual address.
3) The PType is not recognized.

Furthermore, any PDU addressed to the RDE LLC address may be discarded if the SSAP is not the RDE LLC address.

9.4.1 Route query command (RQC)

The RQC is sent as either an NSR PDU or an STE PDU. The choice of NSR or STE depends upon the STR_INDICATOR setting (see 9.6.1.1). The use of an SRF to verify the routing path is allowed but not specified by this International Standard. The values for the RDE PDU information fields are specified in later subclauses.

9.4.2 Route query response (RQR)

The RQR is usually sent as ARE PDU. The ARE response will be used for an RQC sent NSR or STE PDU. The values for the RDE fields are specified in the following subclauses.

When an RQC PDU is received, an RQR PDU is generated by swapping the destination_address and source_ address of the MAC header (MA_UNITDATA request primitive), the content of the information field is modified as follows:

1) The RDE_Ver field is set to 1.
2) The PTYPE field is set to RQR.
3) The OPTIONS field returned reflects only the options supported by the local RDE.

9.4.3 Route selected command (RSC)

The RSC is sent as an SRF PDU. The values for the RDE fields are specified in the following subclauses.

9.5 Encoding of the routing information field (RIF)

The RIF varies in length up to 30 octets. When source routing is not used, the RIF is said to be NULL and contains no information. The I/G bit of the MAC source address is defined as the routing information indicator (RII). When the RIF is null, this bit is set to 0. When the RIF is not null, the RII is set to 1. For further details on the format and encoding of the RIF, see ISO/IEC 10038 : 1993, annex C.

9.6 RDE route control process

9.6.1 Route control operating parameters

9.6.1.1 STR_INDICATOR

STR_INDICATOR is a station policy parameter that defines how the spanning tree shall be used. STR_INDICATOR shall assume a value of either NSR or STE. When STR_INDICATOR is set to the value NSR, the route indicated by STR shall be interpreted as RIF is null. When STR_INDICATOR is set to the value STE, the route indicated by STR shall have the following parameters (RTYPE=STE, D=0, LF=111111, LTH=2).

9.6.1.2 AGING_ENABLED(LOCAL_LSAP)

The AGING_ENABLED parameter is a flag that, for each local LSAP, indicates whether an aging function is enabled (true) or disabled (false).

9.6.2 Route determination parameters

The route determination parameters are used by the RDC. There is a set of route determination parameters for each individual LLC address (LOCAL_LSAP) supported by the LLC and the parameter will be designated by the LOCAL_LSAP value.

9.6.2.1 RDE_ENABLED(LOCAL_LSAP)

An RDE policy flag that, when set true, allows route determination operation for the specific SAP.

9.6.2.2 REPLACE(LOCAL_LSAP)

An RDE policy flag that, when set true, enables the RDC to continually update the route to the one being used by its peer (if the route meets the route criteria). When false, it disables the automatic updating of the route while an acceptable route is selected.

9.6.2.3 HOLD(LOCAL_LSAP)

An RDE policy flag that, when set true, requires that a new route not be used (delayed) until all PDUs using the old route have had time to exit the system. This flag is used by the RDCs.

9.6.2.4 MAX_RD(LOCAL_LSAP)

An RDE parameter that limits the number of route descriptors that may be contained in a valid route and thus limits the number of bridges that may be used. This parameter is used by the RDC.

9.6.2.5 MIN_PDU_SIZE(LOCAL_LSAP)

An RDE parameter that establishes the lower limit on the number of octets that must be supported by a valid route. This parameter is used by the RDC.

9.6.2.6 MAX_RESP_TIME(LOCAL_LSAP)

An RDE parameter that establishes the maximum round-trip response time for the route query process. This parameter is used by the RDC.

9.6.2.7 RESET_ON_TEST(LOCAL_LSAP)

An RDE parameter that, when enabled, indicates that a SEND_TEST_C event causes a RESET_ ROUTE request primitive action for the specified SAP.

9.6.3 Precise description of route control procedures

9.6.3.1 Precise specification

The operation of RDE is logically divided into two components. Each component characterizes a set of protocol operations performed by an RDE and is defined using a protocol state machine description. These state machines do not specify particular implementation techniques; rather, they are intended to describe the "external" characteristics of an LLC entity as perceived by an LLC entity in a remote station.

The route determination operation is described using the following components:

1) **Route Control Component (RCC):** The RCC performs two basic functions. The first function is to supply routing information for the MA_UNITDATA request primitive generated by the LLC operation. For this function, it is a service user of the RDC. The second function is to provide the frame based protocol used for route discovery. For this function, it is the service provider to the RDC. One RCC shall exist per LLC.
2) **RDE Route Determination Component (RDC):** The RDC is responsible for processing the events that affect the route of a specific data link. One RDC shall exist for each data link supported by the LLC.

9.6.3.2 RCC overview

The RCC provides routing for PDUs sent from the LLC entity and derives routing information from PDUs addressed to the LLC entity. PDUs are classified into the following three groups:

1) Those PDUs addressed to or sent from the NULL LLC address
2) Those PDUs addressed to or sent from the RDE LLC address
3) All others

The RCC handles each of these groups differently.

— PDUs originated by the null SAP shall be sent on the spanning tree path. Any response PDU sent from the null SAP shall use the same path used by the command.

— All PDUs from/to the RDE LLC address originate/terminate in the RCC. The RCC is responsible for processing the RQ and RS PDUs. The RDE SAP is an internal SAP to the RDE LLC entity and handles all PDUs addressed to the RDE LLC address. When requested by an RDC, the RCC shall generate the RQC PDU and RS PDU addressed to the RDE SAP of the designated remote station. When the RCC receives an RQR PDU or an RS PDU, it indicates its reception to the appropriate RDC. When the RCC receives an RQC, it not only indicates the reception to the appropriate RDC, but it also responds with an RQR PDU to the requesting station.

— The RCC provides routes for all other PDUs as follows. A PDU sent to the broadcast MAC address or a group MAC address shall use the spanning tree path; otherwise, the RCC requests the route from the appropriate RDC. When a PDU is received, the RCC indicates the received route to the appropriate RDC.

  Each RDC is designated by both the local LLC address it serves and remote LSAP address to which it provides the route. The local LLC parameter shall be indicated by LOCAL_LSAP. The remote LSAP address shall be designated by the combination of the remote MAC address (RMAC) and the remote LLC address (RSAP). A particular RDC is therefore designated by specifying source_address and destination_address.

9.6.3.3 Route control event/action specifications

When the station is operational, the RCC is enabled. There is only one state (operational) for the RCC and the transitions indicate the action to be taken when the event occurs. The state transitions are specified as receive actions, send actions, and reset actions. The receive actions are shown in table 7, the send actions are shown in table 8, and the reset actions are shown in table 9.

9.6.3.4 Route control RECEIVE event/action specifications

The state machine for the receive actions is shown in table 7.

9.6.3.4.1 Route control RECEIVE event specifications

1) **CTL=TEST-C:** The control field specifies a TEST command PDU.
2) **CTL=XID-C:** The control field specifies an XID command PDU.
3) **CTL<>(UI,XID,TEST):** The control field is neither UI, nor XID, nor TEST.
4) **DSAP=NULL:** The DSAP address field of the received PDU = ‘00000000’.
5) **DSAP=RDE:** The DSAP address field of the received PDU = ‘01100101’.
6) **DSAP=YYY:** The DSAP address field is not the null address and is not the RDE address.
7) **MA_UNITDATA_INDICATION:** The MAC indicates that an LLC PDU has been received.
8) **PTYPE=RQC:** The RDE PDU is an RQC.

Table 7—Route control receive actions

	Event	Action
1	MA_UNIDATA_INDICATION & DSAP=NULL & RIF=NULL	RECEIVE_XXX_PDU SAVE_ROUTE(SA,SSAP)=NULL
2	MA_UNITDATA_INDICATION & DSAP=NULL & RIF <>NULL	RECEIVE_XXX_PDU SAVE_ROUTE(SA,SSAP)=REVERSE
3	MA_UNITDATA_INDICATION & DSAP=YYY & DA_I/G=0 & RIF=NULL	RECEIVE_XXX_PDU
4	MA_UNITDATA_INDICATION & DSAP=YYY & DA_I/G=0 & RIF<>NULL & RTYPE=SRF	RECEIVE_XXX_PDU RIF_INDICATION
5	MA_UNIDATA_INDICATION & DSAP=YYY & DA_I/G=0 & RIF<>NULL & RTYPEE<>SRF	RECEIVE_XXX_PDU
6	MA_UNITDATA_INDICATION & DSAP=YYY & DA_I/G=1	RECEIVE_XXX_PDU
7	MA_UNITDATA_INDICATION & DSAP=RDE & PTYPE=RS	RDE_PDU_INDICATION
8	MA_UNITDATA_INDICATION & DSAP=RDE & PTYPE=RQC & RIF=NULL	SEND_RQR; RIF=ARE RDE_PDU_INDICATION
9	MA_UNITDATA_INDICATION & DSAP=RDE & RTYPE=RQC & RIF<>NULL & RTYPE=STE	SEND_RQR; RIF=ARE RDE_PDU_INDICATION
10	MA_UNITDATA_INDICATION & DSAP=RDE & PTYPE=RQC & RIF<>NULL & RTYPE=ARE	SEND_RQR; RIF=REVERSE RDE_PDU_INDICATION
11	MA_UNITDATA_INDICATION & DSAP=RDE & PTYPE=RQC & RIF<>NULL & RYPE=SRF	SEND_RQR; RIF=REVERSE RDE_PDU_INDICATION
12	MA_UNITDATA_INDICATION & DSAP=RDE & PTYPE=RQR	RDE_PDU_INDICATION
13	MA_UNITDATA_INDICATION & DSAP=RDE & (SA_I/G=1 or CTL<>(UI,XID,TEST)	INC_RDE_PROT_ERROR
14	MA_UNITDATA_INDICATION & DSAP=RDE & CTL=XID-C & RIF=NULL	SEND_XID_R; RIF=NULL
15	MA_UNITDATA_INDICATION & DSAP=RDE & CTL=XID-C & RIF<>NULL	SEND_XID_R; RIF=REVERSE
16	MA_UNITDATA_INDICATION & DSAP=RDE & CTL=TEST-C & RIF=NULL	SEND_TEST_R; RIF=NULL
17	MA_UNITDATA_INDICATION & DSAP=RDE & CTL=TEST-C & RIF<>NULL	SEND_TEST_R; RIF=REVERSE

9) **PTYPE=RQR:** The RDE PDU is an RQR.
10) **PTYPE=RS:** The RDE PDU is an RSC.
11) **RIF=NULL:** The PDU was sent without a routing_information parameter.
12) **RIF<>NULL:** The PDU was sent with a routing_information parameter.
13) **RTYPE<>SRF:** The RTYPE of the received RIF is not SRF.
14) **RTYPE=ARE:** The RTYPE of the received RIF is ARE.
15) **RTYPE=SRF:** The RTYPE of the received RIF is SRF.
16) **RTYPE=STE:** The RTYPE of the received RIF is STE.
17) **SA_I/G=1:** The MAC source address is a group address as specified by the I/G bit being set to 1.

9.6.3.4.2 Route control RECEIVE action specifications

1) **INC_RDE_PROT_ERROR:** If implemented, this shall increment the invalid RDE PDU counter.
2) **RDE_PDU_INDICATION:** The RCC shall indicate the reception of the RDE PDU to the RDC designated by the source_address and destination_address parameters. If the TARGET_SAP is the global address, then the indication will be made to all RDCs designated by RMAC/RSAP. If TARGET_SAP is a group address, then the indication shall be made to all members of the group designated by RMAC/RSAP.
3) **RECEIVE_XXX_PDU:** The PDU shall be processed by the LLC procedures specified in this International Standard.
4) **RIF=ARE:** The RCC shall send the RDE PDU on all paths as follows. The RIF shall contain the parameters of RTYPE=ARE, LTH=2, D=0, LF=111111. (The LF field may be set to a value to represent the maximum capability of the station.)
5) **RIF=REVERSE:** The RCC shall send the RDE PDU on the reverse path as follows. The RIF shall be the RIF from the received PDU with RTYPE=SRF and the Direction (D) bit inverted.
6) **RIF_INDICATION:** The RCC shall indicate the received RIF to the appropriate RDC. The RDC is designated by the source_address and the destination_address parameters.
7) **SAVE_ROUTE(SA,SSAP)=NULL:** The RCC shall store the received RIF referenced to the LSAP address specified by the SA:SSAP of the received PDU for use with the response PDU. The saved ROUTE is null, indicating NSR.
8) **SAVE_ROUTE(SA,SSAP)=REVERSE:** The RCC shall store the received RIF referenced to the LSAP address specified by the SA:SSAP of the received PDU for use with the response PDU. The saved ROUTE shall be the received RIF with the RTYPE set to SRF and the direction (D) bit inverted.
9) **SEND_RQ_R:** The RCC shall create an RDE UI PDU with the following parameters. DA=SA of received PDU, PROTOCOL_ID=0, RDE_Ver=1, PTYPE=RQR, ORIG_MAC=ORIG_MAC of received PDU, ORIG_SAP=ORIG_SAP of received PDU, TARGET_MAC=MAC address of this station, TARGET_SAP=TARGET_SAP of received PDU, and an OPTIONS field reflecting only those options supported by the local RDE.
10) **SEND_TEST_R:** The RCC shall create a TEST response PDU, using the SSAP address of the TEST command PDU as the DSAP address of the response PDU, and using the RDE LLC address.
11) **SEND_XID_R:** The RCC shall create an XID response PDU, using the SSAP address of the command PDU as the DSAP address of the response PDU, and using the RDE LLC address.

9.6.3.5 Route control SEND event/action specifications

The state machine for the send actions is shown in table 8.

9.6.3.5.1 Route control SEND event specifications

1) **DA_I/G=0:** The MAC destination address is an individual address as specified by the I/G bit being 0.
2) **DA_I/G=1:** The MAC destination address is a group address as specified by the I/G bit being 1.
3) **GET_ROUTE_CONFIRM:** The RDC has responded to the GET_ROUTE_REQUEST primitive and returned a route.
4) **PTYPE=RQC:** The RDE_PDU_REQUEST primitive is for an RQC PDU.
5) **PTYPE=RS:** The RDE_PDU_REQUEST primitive is for an RS PDU.
6) **RDE_PDU_REQUEST:** The RDC has requested that an RDE_PDU be sent to its peer.
7) **ROUTE<>NULL:** The GET_ROUTE_CONFIRM primitive has provided a ROUTE that is not null (i.e., SRF).
8) **ROUTE=NULL:** The GET_ROUTE_CONFIRM primitive has provided a ROUTE that is null.
9) **SEND_NULL_XXX_C:** The LLC entity is sending a command PDU from the null SAP (SSAP=00000000).

Table 8—Route control SEND actions

	Event	Action
1	SEND_NULL_XXX_C & STR_INDICATOR=NSR	MA_UNITDATA_REQUEST; RIF=NULL INC_ST_PDU_COUNT
2	SEND_NULL_XXX_C & STR_INDICATOR=STE	MA_UNITDATA_REQUEST; RIF=STE INC_ST_PDU_COUNT
3	SEND_NULL_XXX_R & SAVED_ROUTE (DA,DSAP)=<>NULL	MA_UNITDATA_REQUEST; RIF=NULL
4	SEND_NULL_XXX_R & SAVED_ROUTE (DA,DSAP_<>NULL	MA_UNITDATA_REQUEST; RIF= SAVED_ ROUTE(DA,DSAP) INC_SRF_PDU_COUNT
5	SEND_YYY_PDU & DA_I/G=1 & STR_ INDICATOR=NSR	MA_UNITDATA_REQUEST; RIF=NULL
6	SEND_YYY_PDU & DA_I/G=1 & STR_ INDICATOR=STE	MA_UNITDATA_REQUEST; RIF=STE
7	SEND_YYY_PDU & DA_I/G=0	GET_ROUTE_REQUEST
8	GET_ROUTE_CONFIRM & ROUTE=NULL & STR_INDICATOR=NSR	MA_UNITDATA_REQUEST; RIF=NULL INC_ST_PDU_COUNT
9	GET_ROUTE_CONFIRM & ROUTE=NULL & STR_INDICATOR=STE	MA_UNITDATA_REQUEST; RIF=STE INC_ST_PDU_COUNT
10	GET_ROUTE_CONFIRM & ROUTE<>NULL	MA_UNITDATA_REQUEST; RIF=ROUTE INC_SRF_PDU_COUNT
11	RDE_PDU_REQUEST & PTYPE=RQC & STR_INDICATOR=NSR	MA_UNITDATA_REQUEST; PDU=RQC RIF=NULL
12	RDE_PDU_REQUEST & PTYPE=RQC & STR_INDICATOR=STE	MA_UNITDATA_REQUEST; PDU=RQC RIF=STE
13	RDE_PDU_REQUEST & PTYPE=RS	MA_UNITDATA_REQUEST; PDU=RS RIF=SRF

10) **SEND_NULL_XXX_R:** The LLC entity is sending a response PDU from the null SAP (SSAP=10000000).
11) **SEND_YYY_PDU:** The LLC entity is sending any PDU from a SAP whose LLC address is not the null address (SSAP=ssssssx where sssssss is not 0000000).
12) **STR_INDICATOR=NSR:** The station's spanning tree route indicator requires that PDUs sent on the spanning tree route do not use source routing (RIF=NULL).
13) **STR_INDICATOR=STE:** The station's spanning tree route indicator requires that PDUs sent on the spanning tree route use the STE RIF.

9.6.3.5.2 Route control SEND action specifications

1) **GET_ROUTE_REQUEST:** The RCC shall issue a get_route request primitive to the RDC designated by the source_address and destination_address parameters. The PDU shall be held pending the get_route response primitives.

2) **INC_SRF_PDU_COUNT:** If implemented, increment the SRF counter.

3) **INC_ST_PDU_COUNT:** If implemented, increment the spanning tree routed PDU counter.

4) **MA_UNITDATA_REQUEST:** The RCC shall generate an MA_UNITDATA request primitive to the MAC with the indicated RIF.

5) **PDU=RQC:** The RCC shall generate an RDE UI PDU with the following parameters: PROTOCOL_ID=0, RDE_Ver=1, TARGET_MAC=RMAC, ORIG_MAC=MA, SSAP= DSAP=RDE LLC address, PTYPE=RQC, ORIG_SAP=LOCAL_LSAP, TARGET_SAP=RSAP.

6) **PDU=RS:** The RCC shall generate an RDE UI PDU with the following parameters: PROTOCOL_ID=0, RDE_Ver=1, TARGET_MAC=RMAC, ORIG_MAC=MA, SSAP=DSAP=RDE LLC address, PTYPE=RQC, ORIG_SAP=LOCAL_LSAP, TARGET_SAP=RSAP. The RIF shall be an SRF specified by the RDC.

7) **RIF=NULL:** The RIF for the MA_UNITDATA request primitive is null indicating NSR.

8) **RIF=ROUTE:** The RCC shall use the RIF provided by the get_route response primitive for the MA_UNITDATA request primitive. Note that if the HOLD policy is selected, multiple PDUs may be pending for a particular RDC. All waiting PDUs with SSAP=LOCAL_LSAP, DA=RMAC, and DSAP=RSAP being held for the get_route_response primitive shall be sent.

9) **RIF=SAVED_ROUTE(DA,DSAP):** The RCC shall use the ROUTE saved for the LSAP address specified by DA:DSAP of the PDU to be sent in the MA_UNITDATA request primitive.

10) **RIF=SRF:** The RIF for the MA_UNITDATA request primitive shall be an SRF, specified by the RDC.

11) **RIF=STE:** The RIF for the MA_UNITDATA request primitive shall be an STE. The RIF shall contain the parameters of RTYPE=STE, LTH=2, D=0, LF=111111. (The LF field may be set to a value to represent the maximum capability of the station.)

9.6.3.6 Route control reset EVENT/ACTION specifications

The state machine for the reset actions is shown in table 9.

Table 9—Route control reset actions

	Event	Action
1	DL-RESET_REQUEST	RESET_ROUTE_REQUEST
2	SEND_TEST_C & RESET_ON_TEST_ENABLED	RESET_ROUTE_REQUEST

9.6.3.6.1 Route control reset EVENT specifications

1) **DL-RESET_REQUEST:** The network layer entity is requesting that the LLC reset the specified data link connection.

2) **RESET_ON_TEST_ENABLED:** The RESET_ON_TEST parameter is enabled (set to true).

3) **SEND_TEST_C:** The LLC entity (SAP component) is sending a TEST command PDU to a remote SAP. This function is controlled by the value of the RESET_ON_TEST parameter.

9.6.3.6.2 Route control reset ACTION specifications

1) **RESET_ROUTE_REQUEST:** Issue the reset signal to the appropriate RDC(s).

9.7 The route determination component (RDC)

The RDC is responsible for processing the events that affect the selection of a route for a specific data link. One RDC shall exist for each data link supported by the LLC.

9.7.1 The route determination operating parameters

Each RDC is designated by both the local LLC address it serves and remote LSAP address to which it provides the route. The LSAP address is designated by the combination of the MAC address and the LLC address. The local LLC parameter shall be designated by LOCAL_LSAP, which is the combination of the station's MAC address and the local LLC address. The remote LSAP address shall be designated by the combination of the remote MAC address (RMAC) and the remote LLC address (RSAP).

The RDC operation is controlled by certain RDE policies established for each local SAP. These parameters are

— **REPLACE:** An RDE policy flag that, when set true, enables the RCC to update continually the route to the one being used by its peer if the route meets the route criteria. When false, it disables the automatic updating of the route while an acceptable route is selected.
— **HOLD:** An RDE policy flag that, when set true, requires that a new route not be used (delayed) until all PDUs using the old route have had time to exit the system.

In addition, certain parameters are established for minimum route criteria. These parameters are

— **MAX_RESP_TIME:** A mandatory RDE parameter that establishes the maximum round-trip response time for the route query process.
— **MAX_RD:** An optional RDE parameter that establishes the maximum number of route descriptors that a valid route may contain.
— **MIN_PDU_SIZE:** An optional RDE parameter that establishes the minimum PDU size that must be supported by the route.

9.7.2 Precise description of route determination procedures

9.7.2.1 Precise specification

The operation of the RDC is logically divided into two state machines. Each state machine characterizes a set of protocol operations performed by the RDC and is defined using a protocol state machine description. These state machines do not specify particular implementation techniques; rather, they are intended to describe the "external" characteristics of an LLC entity as perceived by an LLC entity in a remote station.

The route determination operation is described using the following state machines:

1) **The route selection state machine:** This state machine is responsible for the route discovery process and the selection of a suitable route.
2) **The route administration state machine:** This state machine is responsible for providing the selected route for service. This state machine provides the response to the GET_ROUTE request primitive. The main purpose of this state machine is to enforce the HOLD policy.

9.7.3 Route selection state machine

9.7.3.1 Route selection state machine overview

The route selection state machine provides the route discovery for the data link. It is responsible for generating RQCs and using the RQRs to select the route that will be used by PDUs generated by the local LSAP. The state transitions are shown in table 10.

Table 10—Route selection state machine

State	Event	Action	Next
RESET	GET_ROUTE_REQUEST	START_SELECTION:RESET_TRS; RDE_PDU_REQUEST(RQC)	QUERY
	RIF_INDICATION & RIF>=CRITERIA & REPLACE_POLICY	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
	RDE_PDU_INDICATION=RS & RIF>=CRITERIA	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_ INDICATION	NEW
	RDE_PDU_INDICATION=RQC	RESET_TRS	WAIT
QUERY	TRS_EXPRIES	ROUTE=NULL	ROUTE
	SELECTION_COMPLETE	ROUTE=SELECTION; RESET_TRS; RDE_PDU_REQUEST (RS); NEW_ROUTE_INDICATION	NEW
	RIF_INIDICATION & RIF>=CRITERIA & REPLACE_POLICY	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
	RDE_PDU_INDICATION=RS & RIF>= CRITERIA & REPLACE_POLICY	ROUTE=REVERSE;RESET_TRS; NEW_ROUTE_INDICATION	NEW
WAIT	TRS_EXPIRES	START_SELECTION; RESET_TRS; RDE_PDU_REQUEST(RQC)	QUERY
	RDE_PDU_INDICATION=RS & RIF<CRITERIA	START_SELECTION; RESET_TRS; RDE_PDU_REQUEST (RQC)	QUERY
	RIF_INDICATION & RIF>=CRITERIA & REPLACE_POLICY	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
	RDE_PDU_INDICATION=RS & RIF>=CRITERIA	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
NEW	TRS_EXPIRES		ROUTE
ROUTE	RESET_ROUTE_REQUEST	START SELECTION; RESET TRS; ROUTE=NULL; RDE_PDU_REQUEST (RQC); NEW_ROUTE_INDICATION	QUERY
	RIF_INDICATION & RIF<>ROUTE & REPLACE_POLICY & RIF>=CRITERIA	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
	RDE_PDU_INDICATION=RS & RIF>=CRITERIA & REPLACE_POLICY & RIF<>ROUTE	ROUTE=REVERSE; RESET_TRS; NEW_ROUTE_INDICATION	NEW
	RDE_PDU_INDICATION =RQC	RESET_TRS; ROUTE=NULL NEW_ROUTE_INDICATION	WAIT
ANY	FLUSH_ROUTE	ROUTE=NULL; NEW_ROUTE_INDICATION	RESET

9.7.3.1.1 Modes of operation

There are two modes of operation, and they are controlled by the REPLACE policy. The first mode is selected when the value of the REPLACE policy flag is false. In this mode, once a route has been selected, it remains valid until a reset command is issued to the RDC or the peer invokes the route discovery process.

The other mode is selected when the REPLACE policy value is set to true. In this mode, the state machine learns and selects the route used by its peer (if the route meets the required criteria).

9.7.3.1.2 Procedure for route discovery

When a route must be determined, the RDC will send an RQC PDU to the remote station (its peer) and enable the selection function. The remote station responds to the RQC with an RQR PDU using the ARE route type. For each unique path between the stations, an RQR PDU will be received by the originating station with the route indicated by the RIF. The selection function selects the most appropriate response.

9.7.3.1.3 Procedure for route selection

1) **Route reset phase:** This phase is entered when the station is initialized and whenever management flushes the route. No route has been selected (ROUTE=NULL) and the RDC stays in this state as long as the data link is not in active use. This state is exited when an event occurs that indicates the data link is in use. This phase is represented by the reset state.
2) **Query route phase:** This phase is entered when a route is needed and is not known. Entry into this phase starts the route discovery process. This phase is represented by the query state.
3) **Waiting route phase:** This phase is entered when the peer has started the route discovery process and this station is waiting to use the result. This phase is represented by the wait state.
4) **New route phase:** This phase indicates that a new route has recently been acquired and route discovery or replacement is prohibited. This phase is entered when a new route has been selected. During this phase, further action is inhibited until the system has converged. This prevents a phasing problem between two stations each causing the other to restart the discovery process. This phase is represented by the new state.
5) **Source route phase:** This phase is entered once a route has been selected and sufficient time has elapsed to allow the system to settle, This phase is represented by the route state.

9.7.3.1.4 Procedure for route reset

The state machine may be reset by management action issuing the flush command, in which case the state machine discards the current route, enters the reset state, and waits for an event that would require a route to be selected. The RCC can issue a reset request that will cause the current route to be discarded and restart the route discovery process.

9.7.3.1.5 Route selection elements

1) **Timer function:** The route selection timer (TRS) is used to limit the time RDE remains in certain phases of the route selection process. The timer action of reset causes the timer value to be reset to zero and the timer started. When the timer reaches the TRS value specified by management, the timer is said to have expired and will be indicated to the state machine. The primary use of this timer is to limit the time the RDE remains in the deciding state. The TRS expiration value represents (and should be equal to or greater than) the maximum round-trip latency.
2) **Selection function:** The actual route selection function is not specified, thus allowing implementations to specify their own criteria for the actual selection. Many different procedures exist for the selection function; some favor maximum advantage for the station, while others favor maximum advantage for the bridged network.

The selection function is activated when the RQC is sent and it performs its selection based on the responses. The output from the function is the selected route and an indication that the route selection function has completed.

3) **Aging function:** The aging function is a mechanism used to detect and delete stale routes from the source route cache.

9.7.3.2 Route selection STATE specifications

The route selection state diagram is shown in figure 35.

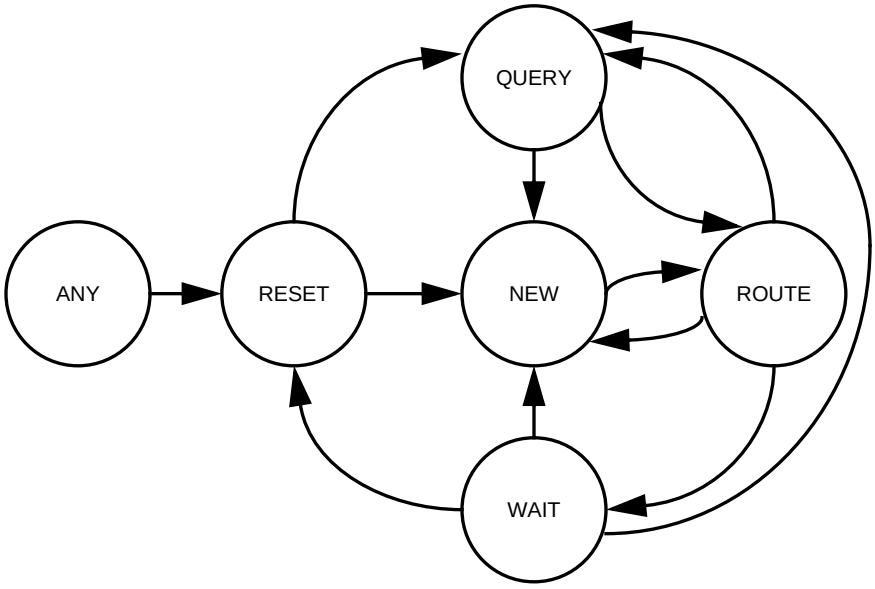


Figure 35—Route selection state diagram

1) **ANY:** The action is not dependent on the current state.
2) **NEW:** A new route has been determined and the route discovery process is inhibited until the route has aged to allow the system to converge.
3) **QUERY:** This RDC has initiated the route discovery process and is waiting for the route selection function to provide a route.
4) **RESET:** The route selection is in the RESET state. In this state, the state machine is not active. The ROUTE is set to NULL and the timer TRS is disabled. The RDC will wait for activity that indicates a route is needed for this data link.
5) **ROUTE:** The route has been selected and will be used until there is an indication that a new route is needed.
6) **WAIT:** The peer is in the process of route discovery and this RDC is waiting for the peer to select a route.

9.7.3.3 Route selection EVENT/CONDITIONS specifications

1) **FLUSH_ROUTE:** A signal from system management to discard the current route. This does not cause RDE to discover a new route.
2) **GET_ROUTE_REQUEST:** A request from the RCC that a PDU is being sent and that routing information for this data link is required.
3) **RDE_PDU_INDICATION=RQC:** A signal indicating that an RQC PDU was received for this data link.

4) **RDE_PDU_INDICATION=RS:** A signal indicating that an RS PDU was received for this data link.
5) **REPLACE_POLICY:** The replace policy for this local LSAP is set to true.
6) **RESET_ROUTE_REQUEST:** A signal from RCC to discard the current route and discover another.
7) **RIF>=CRITERIA:** The RIF of the received PDU meets the criteria for an acceptable route.
8) **RIF_INDICATION:** An indication that an RIF for this data link was received.
9) **RIF<>ROUTE:** As a minimum, the RIF must be different than the current ROUTE, the LTH field cannot be greater than MAX_RDx2+2, and the RIF LF bits cannot indicate an information field less than MIN_PDU_SIZE.
10) **SELECTION_COMPLETE:** A signal indicating that the route selection function has selected a route.
11) **TRS_EXPIRES:** The timer TRS has expired.

9.7.3.4 Route selection ACTION specifications

1) **NEW_ROUTE_INDICATION:** An indication to the route administration state machine that the ROUTE has changed, and is only required if the HOLD policy is implemented.
2) **RDE_PDU_REQUEST(RQC):** The RDC shall generate a request to the RCC to send an RQC PDU to the remote RDE.
3) **RDE_PDU_REQUEST(RS):** The RDC shall generate a request to the RCC to send an RS PDU to the remote RDE.
4) **RESET_TRS:** The timer TRS shall be reset to its configured value.
5) **ROUTE=NULL:** A specific path is not available and the ROUTE to be used is the spanning tree path.
6) **ROUTE=REVERSE:** The RDC shall store the received RIF as the ROUTE as follows. The saved ROUTE shall be the received RIF with the RTYPE set to SRF and the direction (D) bit inverted.
7) **ROUTE=SELECTION:** The route to be used for subsequent PDUs will be the route output of the route selection function.
8) **START_SELECTION:** The RDC shall generate a signal to the selection function that the query command was sent and the selection function shall select a route based on its responses.

9.7.4 Route administration state machine

9.7.4.1 Route administration state machine overview

The route administration state machine supplies the response to the GET_ROUTE request primitive and therefore provides the ROUTE to the RCC. This state machine is responsible for enforcing the HOLD policy and thus ensuring that changing routes will not cause frame misordering. The state transitions are shown in table 11.

9.7.4.1.1 Modes of operation

There are two effective modes of operation. During normal operation, the GET_ROUTE request primitive from the RCC is serviced immediately returning the ROUTE. When the HOLD policy is in effect and the route has been changed, all additional requests will be held until the route timer has expired.

9.7.4.1.2 Procedure for sending PDUs

The RCC will issue the GET_ROUTE request primitive to the appropriate RDC and hold the PDU pending the GET_ROUTE confirm primitive. The RDC will process the request according to the route administration state machine, returning the selected ROUTE. The RCC will then generate an MA_UNITDATA request primitive to the MAC specifying the route returned by the RCC.

Table 11—Route administration state machine

State	Event	Action	Next
EMPTY	GET_ROUTE_REQUEST	GET_ROUTE_CONFIRM; RESET_TRR	IN-USE
IN-USE	NEW_ROUTE_INDICATION & HOLD_POLICY		NEW
	GET_ROUTE_REQUEST	GET_ROUTE_CONFIRM; REST_TRR	
	TRR_EXPIRES		EMPTY
NEW	GET_ROUTE_REQUEST		HOLD
	TRR_EXPIRES		EMPTY
HOLD	TRR_EXPIRES	GET_ROUTE_CONFIRM; RESET_TRR	IN-USE

9.7.4.1.3 Procedure for route administration

1) **System empty phase:** In this phase no PDUs are currently in the system.
2) **Route in-use phase:** PDU has been sent on the current route but has not had time to traverse the system.
3) **New route pending phase:** A new route has been selected and there is a PDU using a previous route that has not had time to traverse the system.
4) **Holding phase:** A PDU is waiting to use the new route while a PDU using a previous route is traversing the system.

9.7.4.1.4 Route administration elements

1) **Timer Function:** The route response timer (TRR) is used to time PDUs in the system. When a PDU is sent, TRR is reset to zero and started. When it reaches its threshold, it is said to have expired. TRR should have an expiration value equal to or greater than the maximum one-way latency. The primary use of this timer is to pace the time the RDE waits after using a route before a new route may be used. The state machine allows for the implementation of a single timer (when TRR = TRS). TRR shall have an expiration value not less than one half of the TRS expiration value and not greater than the TRS expiration value.

9.7.4.2 Route administration states

The route administration state diagram is shown in figure 36.

1) **EMPTY:** This is the initial state and is entered after all sent PDUs have had time to traverse the system (i.e., the system is empty). Any requests for a route will be serviced immediately. The route timer will be started and transition is made to the in-use state.
2) **HOLD:** This is the state that is entered when a route is in use, a new path has been selected, the HOLD policy is in effect, and another PDU needs to be sent. In this state, any additional route requests will be deferred until the current PDUs have exited the system. When the route timer expires, the outstanding route request is serviced, and transition is made to the in-use state.

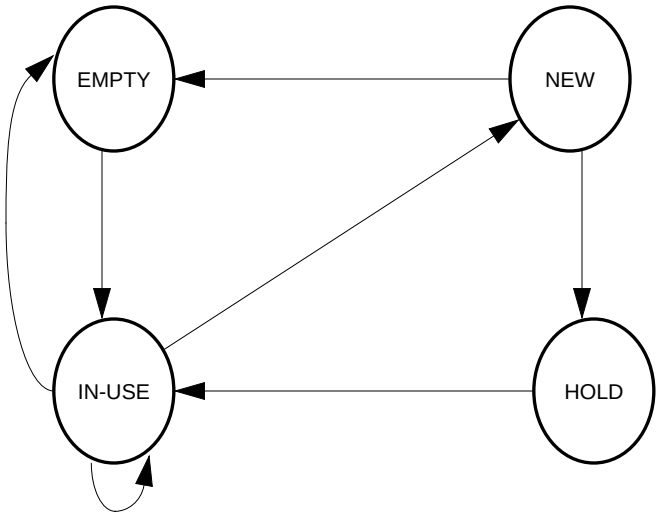


Figure 36—Route administration state diagram

3) **IN-USE:** This is the state that is entered when a route is being used. Any requests for a route will be serviced immediately and the route timer reset. When the route timer expires, transition will be made back to the empty state. If a new route is selected and the HOLD policy is in effect, transition is made to the new state.
4) **NEW:** This is the state that is entered if when a route is in use, a new path has been selected, and the HOLD policy is in effect. When the route timer expires, transition will be made back to the empty state. Requests for routes will not be honored and will cause transition to the hold state.

9.7.4.3 Route administration EVENTS/CONDITIONS

1) **GET_ROUTE_REQUEST:** The RCC has issued a GET_ROUTE request primitive to this component.
2) **HOLD_POLICY:** The HOLD policy for the local LSAP is enabled.
3) **NEW_ROUTE_INDICATION:** A new route has been selected.
4) **TRR_EXPIRES:** The route timer (TRR) has expired.

9.7.4.4 Route administration ACTIONS

1) **GET_ROUTE_CONFIRM:** The RDC shall generate a GET_ROUTE confirm primitive to the RCC providing the current ROUTE. The RCC uses this response to send all held data frames.
2) **RESET TRR:** The route timer shall be restarted from zero.

10. LLC sublayer managed objects

The LLC sublayer consists of several different managed objects. The definitions for these managed objects are based on the modeling and notational concepts embodied in the structure of management information series of standards, ISO/IEC 10165-1 : 1993 and ISO/IEC 10165-4 : 1992.

Management of the LLC sublayer is achieved via several managed objects, as shown in figure 37.

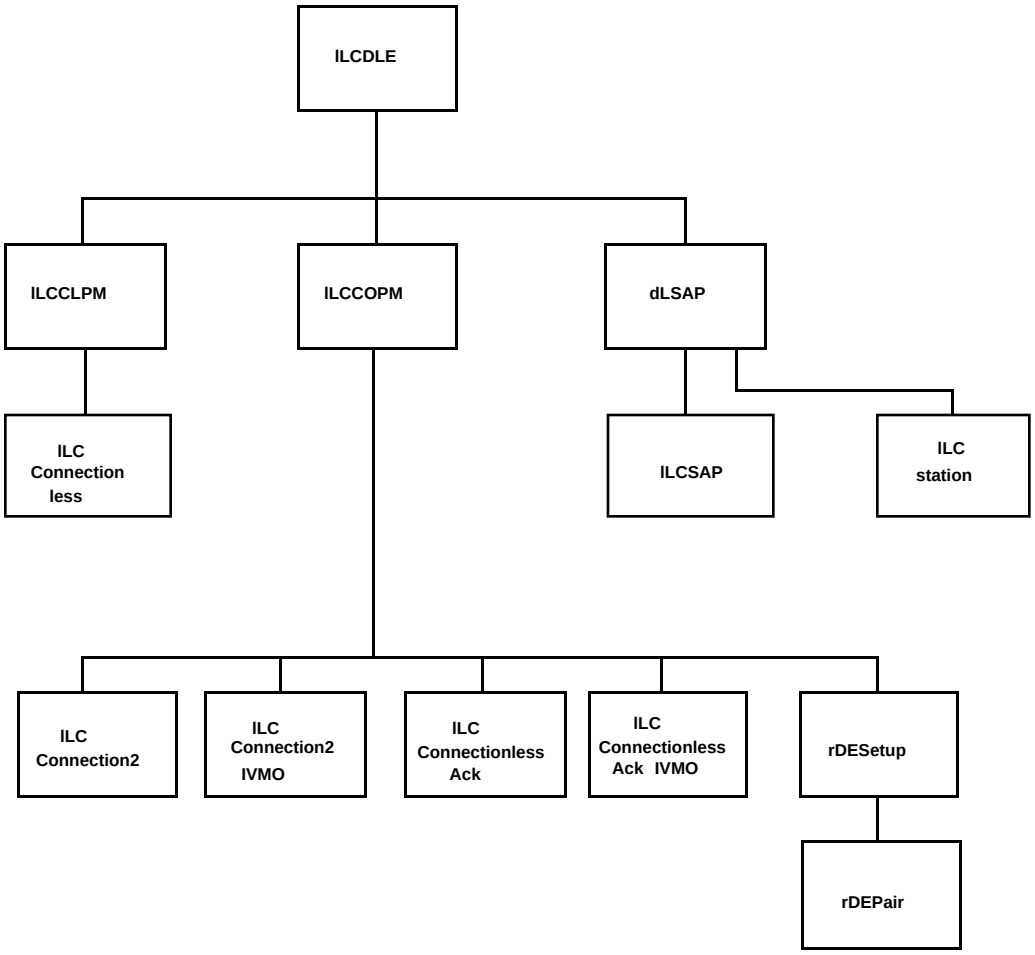


Figure 37—Logical link control containment hierarchy

1) **lLCStation Managed Object:** Contains management attributes, actions, and notifications that exist per station, such as active connections.

2) **lLCSAP Managed Object:** Contains management attributes, actions, and notifications that exist per LLC SAP.

3) **lLCConnectionless Managed Object:** Contains all the attributes, actions, and notifications used in a connectionless (Type 1) LLC environment. A separate instance of this MO exists for each LSAP within a station.

4) **lLCConnection2 Managed Object:** Contains all the attributes, actions, and notifications used in a connection (Type 2) environment.

5) **lLCConnection2IVMO Managed Object:** May be used to supply initial values for the attributes of lLCConnection2 MOs. Different instances of lLCConnection2IVMO may contain different initial values.
6) **lLCConnectionlessAck Managed Object:** Contains all the attributes, actions, and notifications used in an acknowledge connectionless (Type 3) environment.
7) **lLCConnectionlessAckIVMO Managed Object:** May be used to supply initial values for the attributes of lLCConnectionlessAck MOs. Different instances of lLCConnectionlessAckIVMO may contain different initial values.
8) **rDESetup Managed Object:** Contains all the attributes required for setup if an instance of lLCSAP supports RDE.
9) **rDEPair Managed Object:** Exists for each RDE pair used by any Type of LLC service provided. This managed object contains the local MAC address and local SAP and the remote MAC address and remote SAP, plus counters.

10.1 LLCStation managed object

This managed object is responsible for processing the events that affect the entire LLC entity. One LLCStation object shall exist for each MAC service access point present on the local area network. The formal description is as follows:

```
lLCStation MANAGED OBJECT CLASS
  DERIVED FROM "DML": dLSAP ;
  CHARACTERIZED BY lLCStation-P ;
  CONDITIONAL PACKAGES
    lLCDupAddress-P
        PRESENT IF Duplicate station address detection is supported,
    lLCType3-P
        PRESENT IF Type 3 LLC is supported,
    lLCBuffer-P
        PRESENT IF Buffer Management is supported,
    pDUsDiscarded-P
        PRESENT IF PDUs Discarded counter is supported,
    rDE-P
        PRESENT IF RDE is supported by the Station,
    type1AcknowledgmentTimerTimeouts-P
        PRESENT IF PDUs Type 1 Acknowledgment Timer is supported ;
  REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectclass(3) llcstation(9)};

-- Name Bindings --

 lLCStation-NB NAME BINDING
 SUBORDINATE OBJECT CLASS   lLCStation AND SUBCLASSES ;
 NAMED BY
 SUPERIOR OBJECT CLASS "DMI :1992":lLCDLE AND SUBCLASSES;
 WITH ATTRIBUTE      lLCName ;
 REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       llcstationnb(10)};

-- Packages --

lLCStation-P PACKAGE
   BEHAVIOUR
    lLCStation-B BEHAVIOUR
      DEFINED AS
         !The lLCStation package contains the definition of all attributes and
          actions that are common to the LLC station as a whole, regardless of which
         LLC operations are supported.! ;;
   ATTRIBUTES
```

```
     lLCName                          GET,
     maximumLSAPsConfigured           GET,
     numberOfActiveLSAPs              GET,
     supportedServicesTypes           GET,
     status                           GET;
   ACTIONS
      reinitialize-AC ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcstationp(33)} ;

lLCDupAddress-P PACKAGE
  BEHAVIOUR
      lLCDupAddress-B BEHAVIOUR
        DEFINED AS
          !This is a conditional part of the LLCStationComponent object. It
           exists if the LLC object supports duplicate station address detection
          (see  6.9.2.1).! ;;
  ATTRIBUTES
        type1AcknowledgeTimeoutValue   GET-REPLACE ,
        type1MaximumRetryCount         GET-REPLACE ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcdupaddress(2)} ;

lLCType3-P PACKAGE
  BEHAVIOUR
    lLCType3-B BEHAVIOUR
      DEFINED AS
         !This is a conditional part of the LLCStationComponent Object.
         It exists if the LLC object supports LLC Type 3 service.! ;;
  ATTRIBUTES
     maximumPDUN3                   GET-REPLACE ,
     maximumRetransmissions4        GET-REPLACE ,
     receiveVariableLifetime        GET-REPLACE ,
     transmitVariableLifetime       GET-REPLACE ,
     type3AcknowledgeTimeoutValue   GET-REPLACE ,
     type3Retransmissions           GET ;
 NOTIFICATIONS
     lLCStationEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llctype3(3)};

lLCBuffer-P PACKAGE
  BEHAVIOUR
     lLCBuffer-B BEHAVIOUR
      DEFINED AS
         !This package provides the total amount of buffer space available for
         this station, the amount of buffer space in use and the average buffer
         space in use by this station.! ;;
  ATTRIBUTES
    avgBufferUseSize    GET-REPLACE ,
    bufferProblems      GET ,
    bufferSize          GET-REPLACE ,
    maxBufferUseSize    GET-REPLACE ;
  NOTIFICATIONS
    lLCStationEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcbuffer(4)} ;

pDUsDiscarded-P PACKAGE
  BEHAVIOUR
    pDUsDiscarded-B BEHAVIOUR
      DEFINED AS
        !This package is present if the implementation supports the counting
        of the number of invalid PDUs discarded also if implementation
```



```
        supports the counting of PDUs discarded due to the specification
        of an inactive DSAP.!;;
  ATTRIBUTES
    inactiveLSAP          GET ,
    pDUsDiscarded         GET ;
  NOTIFICATIONS
    lLCStationEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       pdusdiscarded(5)} ;

rDE-P PACKAGE
  BEHAVIOUR
    rDE-P-B BEHAVIOUR
      DEFINED AS
        !If RDE is supported by this station then two attributes will be present
        at the Station level.! ;;
  ATTRIBUTES
     sTRIndicator        GET-REPLACE ,
     versionNumber       GET;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       rdep(6)};

type1AcknowledgmentTimerTimeouts-P PACKAGE
   BEHAVIOUR
      bType1AcknowledgmentTimerTimeouts BEHAVIOUR
         DEFINED AS
           !This package is present if the implementation supports the
           Acknowledgment Timer attribute.! ;;
   ATTRIBUTES
      type1AcknowledgmentTimerTimeouts  GET-REPLACE ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type1acknowledgmenttimertimeouts(7)};

-- Attributes --

avgBufferUseSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.AvgBufferUseSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     avgBufferUseSize-B BEHAVIOUR
      DEFINED AS
        !Specifies the average amount of buffer space (in octets) in
        use at the same time by LLC for this station! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       avgbufferusesize(0)};

bufferProblems ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    bufferProblems-B BEHAVIOUR
      DEFINED AS
       !This counter provides a count of PDUs discarded due to buffer
        limitations! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       bufferproblems(1)};

bufferSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.BufferSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
```

```
     bufferSize-B BEHAVIOUR
       DEFINED AS   !The amount of buffer space (in octets) for use by LLC! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       buffersize(2)};

inactiveLSAP ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    inactiveLSAP-B BEHAVIOUR
      DEFINED AS
       !This counts the number of PDUs discarded due to the specification of
       an inactive DSAP! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       inactivelsap(3)};

lLCName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX lLCDefinitions.LLCName ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    lLCName-B BEHAVIOUR
      DEFINED AS
        !This attribute is used to name the instance of the lLCStation
        managed object.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcname(4)} ;

maxBufferUseSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaxBufferUseSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    maxBufferUseSize-B  BEHAVIOUR
      DEFINED AS
        !Specifies the maximum amount of buffer space (in octets) in use at
        the same time by LLC! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maxbufferusesize(5)};

maximumLSAPsConfigured ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumLSAPsConfigured ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     maximumLSAPsConfigured-B  BEHAVIOUR
       DEFINED AS
         !The maximum number of LSAPs that can support at any moment in time.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumlsapsconfigured(6)} ;

maximumPDUN3 ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumPDUN3 ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    maximumPDUN3-B BEHAVIOUR
     DEFINED AS   !The maximum size of a Type 3 command PDU! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumpdun3(7)};

maximumRetransmissions4 ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumRetransmissions4 ;
```



```
  MATCHES FOR EQUALITY ;
 BEHAVIOUR
    maximumRetransmissions4-B BEHAVIOUR
      DEFINED AS
        !The number of retransmissions of a Type 3 PDU due to error
        conditions preventing receipt of an acknowledgment.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
            maximumretransmissions4(8)};

numberOfActiveLSAPs ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.NumberofActiveLSAPs ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     numberOfActiveLSAPs-B  BEHAVIOUR
        DEFINED AS
         !The number of active LSAPs that the station currently supports.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       numberofactivelsaps(9)} ;

pDUsDiscarded ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    pDUsDiscarded-B BEHAVIOUR
     DEFINED AS
         !Provides a total count of invalid PDUs that are discarded. A Discarded
         (invalid) PDU shall be defined as one that meets at least one of
         the following conditions:
         1) It is identified as such by the PHY or the MAC sublayer.
         2) It is not an integral number of octets in length.
         3) It does not contain two properly formatted address fields, one control
         field, and optionally an information field in their proper order.
         4) Its length is less than 3 octets (one-octet control field) or  4 octets
         (two-octet control field).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       pdusdiscarded(10)};

receiveVariableLifetime ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.ReceiveVariableLifetime ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     receiveVariableLifetime-B BEHAVIOUR
       DEFINED AS
        !The amount of time the Type 3 LLC receive state variables are
        maintained! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       receivevariablelifetime(12)};

status ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Status ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    status-B BEHAVIOUR
       DEFINED AS
        !This attribute allows a manager to determine what state (UP, DOWN, etc.)
        the lLCStation entity is in.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       status(13)} ;

strIndicator ATTRIBUTE
```

```
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.STRIndicator ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    sTRIndicator-B BEHAVIOUR
      DEFINED AS
        !A value that indicates if the spanning tree will be traversed by a NSR
        or STE routing.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       strindicator(14)};

supportedServicesTypes ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.SupportedServicesTypes ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     supportedServicesTypes-B  BEHAVIOUR
       DEFINED AS
         !The type or types of service supported (i.e., Type 1, Type 2, Type 3.)! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       supportedservicestypes(15)} ;

transmitVariableLifetime ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.TransmitVariableLifetime ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     transmitVariableLifetime-B BEHAVIOUR
       DEFINED AS
         !The amount of time the Type 3 LLC transmit state variables are
         maintained! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       transmitvariablelifetime(16)};

type1AcknowledgeTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.AcknowledgeTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type1AcknowledgeTimeoutValue-B BEHAVIOUR
     DEFINED AS
       !The timeout value of the Ack timer, which is measured in milliseconds
       (6.9.2.2 and 6.9.2.3).!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type1AcknowledgeTimeoutValue(17)};

type1AcknowledgmentTimerTimeouts ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     type1AcknowledgmentTimerTimeouts-B BEHAVIOUR
       DEFINED AS
        !Specifies the number of times the ACK timer has expired
         (see 6.9.2.2 and 6.9.2.3).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type1acknowledgmenttimertimeouts(18)};

type1MaximumRetryCount ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumRetryCount ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type1MaximumRetryCount-B BEHAVIOUR
```



```
      DEFINED AS
        !The maximum number of retries during duplicate address checking
       (see  6.9.2.2 and 6.9.2.3).!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type1maximumretrycount(19)};

type3AcknowledgeTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Type3AcknowledgeTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     type3AcknowledgeTimeoutValue-B BEHAVIOUR
        DEFINED AS
          !Specifies the time interval during which the LLC expects to
          receive a response to an acknowledged connectionless request!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       attribute(7)type3acknowledgetimeoutvalue(20)};

type3Retransmissions ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3Retransmissions-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the number of PDUs which were retransmitted
        (at least once) by the Type 3 LLC service! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3retransmissions(21)};

versionNumber ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.VersionNumber ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    versionNumber-B BEHAVIOUR
     DEFINED AS
       !The version of sap RDE protocol implemented at this staion.
       The value of 1 will be used to represent this current version.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       versionnumber(22)};

-- Actions --

reinitialize-AC ACTION
  BEHAVIOUR
    reinitialize-B BEHAVIOUR
      DEFINED AS
        !This CONFIRMED action causes the entire LLC sublayer to be reset to its
        initial configuration! ;;
  MODE   CONFIRMED ;
  WITH INFORMATION SYNTAX LLCDefinitions.ReinitializeData;
  WITH REPLY SYNTAX     LLCDefinitions.ReinitializeResponse;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       reinitialize(0)};

-- Notifications --

lLCStationEvent-N NOTIFICATION
  BEHAVIOUR
    lLCStationEvent-B BEHAVIOUR
      DEFINED AS
        !This notification will be sent for either pdusdiscarded, buffer problems,
        or Type 3 Retransmissions ! ;;
```

```
   WITH INFORMATION SYNTAX  LLCDefinitions.LLCStationEvent ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) notification(10)
       llcstationevent(0) } ;
```

10.2 lLCSAP managed object

This managed object contains attributes associated with an LLC SAP that are independent of any particular Type of Operation. The formal description is as follows:

```
lLLSAP MANAGED OBJECT CLASS
  DERIVED FROM "DML":dLSAP;
  CHARACTERIZED BY lLCSAP-P;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectclass(3) llcsap(1)};

-- Name Bindings --

lLCSAP-NB NAME BINDING
  SUBORDINATE OBJECT CLASS lLCSAP AND SUBCLASSES;
  NAMED BY
  SUPERIOR OBJECT CLASS "DMI:1992";lLCDLE AND SUBCLASSES;
  WITH ATTRIBUTE lLCSAPName;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       llcsapnb(1)};

-- Packages --

lLCSAP-P PACKAGE
  BEHAVIOUR
   lLCSAP-B BEHAVIOUR
      DEFINED AS
         !The lLCSAP package contains the definition of all attributes and actions
         associated with an LLC SAP that are independent of that LLC types of
         operation are supported.!;;
  ATTRIBUTES
    lLCSAPName         GET,
    lLCAddress         GET,
    rDE                GET;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcsapp(1)};

-- Attributes --

lLCSAPName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
     LLCDefinitions.LLCSAPName;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
     lLCSAPName-B BEHAVIOUR
       DEFINED AS  !Used to name the instance of the lLCSAP managed object!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcsapname(110)};

lLCAddress ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.LSAP;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    lLCAddress-B BEHAVIOUR
      DEFINED AS    !The individual LLC address identifying this LSAP!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcaddress(111)};
```



```
rDE ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.RDE ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     rDE-B  BEHAVIOUR
        DEFINED AS
          !A boolean value that, if set to 1, means that RDE is supported by this
          SAP.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       rde(11)} ;
```

10.3 LLCConnectionless managed object

This clause contains those attributes, actions, and event notifications associated with an LLC connectionless environment. The formal description is as follows:

```
lLCConnectionless  MANAGED OBJECT CLASS
  DERIVED FROM "DML": lLCCLPM;
  CHARACTERIZED BY lLCConnectionless-P ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectclass(3) llcconnectionless(2)} ;

-- Name Bindings --

lLCConnectionless-NB NAME BINDING
  SUBORDINATE OBJECT CLASS
     lLCConnectionless AND SUBCLASSES ;
  NAMED BY
  SUPERIOR OBJECT CLASS "DML": lLCCLPM AND SUBCLASSES ;
  WITH ATTRIBUTE      lLCClessName ;
  CREATE ;
  DELETE ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       llcconnectionlessnb(2)} ;

-- Packages --

lLCConnectionless-P PACKAGE
  BEHAVIOUR
     lLCConnectionless-B BEHAVIOUR
        DEFINED AS
           !This managed object is responsible for processing the attributes,
           actions, and notifications that are associated with the LLC
          connectionless service.!;;
  ATTRIBUTES
    lLCClessName                          GET ,
    maximumLLCInformationFieldSize        GET ,
    tESTReceivedABBResponse               GET ,
    tESTReceivedCommand                   GET ,
    tESTReceivedResponse                  GET ,
    tESTSentABBResponse                   GET ,
    tESTSentCommand                       GET ,
    tESTSentResponse                      GET ,
    uIReceived                            GET ,
    uISent                                GET ,
    xIDReceivedCommand                    GET ,
    xIDReceivedResponse                   GET ,
    xIDSentCommand                        GET ,
    xIDSentResponse                       GET ;
  ACTIONS
    tESTSendCommand-AC ,
    xIDSendCommand-AC ;
```

```
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcsapcomponent(8)} ;

-- Attributes --

lLCClessName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.LLCClessName ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    lLCClessName-B BEHAVIOUR
      DEFINED AS
        !Provides the name of the higher level entity associated with this
        object class.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcclessname(23)};

maximumLLCInformationFieldSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumLLCInformationFieldSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    maximumLLCInformationFieldSize-B BEHAVIOUR
      DEFINED AS   !The maximum length SDU that the LSAP will accommodate! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumllcinformationfieldsize(24)};

tESTReceivedABBResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    tESTReceivedABBResponse-B BEHAVIOUR
      DEFINED AS
        !Provides a count of TEST responses received by this LSAP
        without the LSDU due to limited resources at the remote station.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       testreceivedabbresponse(25)};

tESTReceivedCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    tESTReceivedCommand-B BEHAVIOUR
      DEFINED AS
        !Provides a count of TEST commands received from the MAC sublayer by
        this LSAP. This includes commands received with any group address
        recognized by this LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)  attribute(7)
       testreceivedcommand(26)};

tESTReceivedResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    tESTReceivedResponse-B BEHAVIOUR
      DEFINED AS
        !Provides a count of TEST responses received from the MAC sublayer by
        this LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       testreceivedresponse(27)};
```



```
tESTSentABBResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
     tESTSentABBResponse-B BEHAVIOUR
       DEFINED AS
         !Provides a count of TEST responses that were delivered to the
         MAC without the LSDU data due to limited resources.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       testsentabbresponse(28)};

tESTSentCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    tESTSentCommand-B BEHAVIOUR
      DEFINED AS
        !Provides a count of TEST commands sent to the MAC sublayer by this
        LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       testsentcommand(29)};

tESTSentResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    tESTSentResponse-B BEHAVIOUR
      DEFINED AS
        !Provides a count of TEST responses sent to the MAC sublayer by
        this LSAP.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       testsentresponse(30)};

uIReceived ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    uIReceived-B BEHAVIOUR
      DEFINED AS
        !Provides a count of user data PDUs that were received by the LSAP from
        the MAC sublayer and passed to the user layer.  This includes commands
        received with group addresses of which this LSAP is a member.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       uireceived(31)};

uISent ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    uISent-B BEHAVIOUR
      DEFINED AS
        !Provides a count of user data PDUs that were accepted by the LSAP from
        the user layer and delivered to the MAC sublayer.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       uisent(32)};

xIDReceivedCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
```

```
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    xIDReceivedCommand-B BEHAVIOUR
      DEFINED AS
        !Provides a count of XID commands received from the MAC sublayer by this
        LSAP. This includes commands received with any group address recognized
        by this LSAP.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       xidreceivedcommand(33)};

xIDReceivedResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    xIDReceivedResponse-B    BEHAVIOUR
      DEFINED AS
        !Provides a count of XID responses received from the MAC sublayer by this
        LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       xidreceivedresponse(34)};

xIDSentCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    xIDSentCommand-B BEHAVIOUR
      DEFINED AS
        !Provides a count of XID commands sent to the MAC sublayer by
        this LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       xidsentcommand(35)};

xIDSentResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    xIDSentResponse-B BEHAVIOUR
      DEFINED AS
        !Provides a count of XID responses sent to the MAC sublayer by this
        LSAP.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       xidsentresponse(36)};

-- Actions --

tESTSendCommand-AC ACTION
  BEHAVIOUR
    tESTSendCommand-AC-B  BEHAVIOUR
      DEFINED AS
        !This action causes the LLCSAP Component to send a TEST command PDU to a
         specified remote LSAP!;;
  WITH INFORMATION SYNTAX LLCSAP.TESTSendCommand ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       testsendcommand(2)} ;

xIDSendCommand-AC ACTION
  BEHAVIOUR
    xIDSendCommand-AC-B BEHAVIOUR
      DEFINED AS
        !Causes the LLCSAP component to send an XID command PDU to a specified
         remote LSAP. The information in the XID PDU (supported LLC types and send
```



```
         window size value) is determined by the current configuration of the LSAP
         and LSAP pair!;;
  WITH INFORMATION SYNTAX LLCDefinitions.XIDSendCommand ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       xidsendcommand(3)} ;
```

10.4 LLCConnection2 managed object

This clause contains those attributes, actions, and event notifications associated with an LLC Type 2 (connection) environment. This managed object is responsible for processing the events that affect the current data link connection during its lifetime (i.e., timeframe of link established until link disconnect). One LLC Connection managed object shall exist for each data link (LLC Type 2) connection supported in the LLC entity. The formal description is:

```
lLCConnection2 MANAGED OBJECT CLASS
  DERIVED FROM "DML": lLCCOPM ;
  CHARACTERIZED BY lLCConnection2-P ;
  CONDITIONAL PACKAGES
     connectionBusy-P
        PRESENT IF the local and remote busy counters are used with LLC Type 2,
     connectionReset-P
        PRESENT IF the local, remote and provider initiated reset counters are
              used with LLC Type 2,
     connectionRoute-P
        PRESENT IF route known to the LSAP pair object,
     type2FlowControl-P
        PRESENT IF flow control is used with LLC Type 2,
     type2I-P
        PRESENT IF the sent and receive I counters are used with LLC Type 2,
     type2Iack-P
        PRESENT IF the sent and received acknowledgment counters are used with
               LLC Type 2,
     type2FRMR-P
        PRESENT IF the sent and receive FRMR counters are used with LLC Type 2,
     type2RR-P
        PRESENT IF the sent and receive RR counters are used with LLC Type 2,
     type2RNR-P
        PRESENT IF the sent and receive RNR counters are used with LLC Type 2,
     type2REJ-P
        PRESENT IF the sent and receive REJ counters are used with LLC Type 2,
     type2SABME-P
        PRESENT IF the sent and receive SABME counters are used with LLC Type 2,
     type2UA-P
        PRESENT IF the sent and receive UA counters are used with LLC Type 2,
     type2DISC-P
        PRESENT IF the sent and receive DISC counters are used with LLC Type 2,
     typ2DM-P
        PRESENT IF the sent and receive DM counters are used with LLC Type 2,
     type2PDUs-P
        PRESENT IF using PDUs event notifications with LLC Type 2,
     type2OptTolIPDUs-P
        PRESENT IF Optional Toleration of Duplicate I PDUs is implemented in
              LLC Type 2,
     type2Violation-P
        PRESENT IF counting violations with LLC Type 2;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectclass(3)
              llcconnection2(3)};
```

-- Name Bindings --

```
lLCConnection2-NB NAME BINDING
SUBORDINATE OBJECT CLASS   lLCConnection2 AND
SUBCLASSES;
NAMED BY
SUPERIOR OBJECT CLASS  "DML": lLCCOPM AND SUBCLASSES;
WITH ATTRIBUTE    lLCConnection2Name;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       llcconnection2nb(3)} ;

-- Packages --

lLCConnection2-P PACKAGE
  BEHAVIOUR
    lLCConnection2-B BEHAVIOUR
       DEFINED AS
        !The LLC Connection object is responsible for processing the attributes,
         actions and notifications that affect a specific data link connection
         (for LLC Type 2 connections).
          One data connection shall exist for each data link connection supported.!;;
  ATTRIBUTES
    lLCConnection2Name                    GET ,
    maximumRetransmissions                GET-REPLACE ,
    receiveWindowSize                     GET-REPLACE ,
    sendWindowSize                        GET-REPLACE ,
    type2AcknowledgeTimeoutValue          GET-REPLACE ,
    type2BusyStateTimeoutValue            GET-REPLACE ,
    type2PBitTimeoutValue                 GET-REPLACE ,
    type2RejectTimeoutValue               GET-REPLACE ,
  ACTIONS
    correlatorExchange-AC ,
  NOTIFICATIONS
    communicationAlarm-N ,
    lLCConnection2Event-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcconnection2p(9)} ;

connectionBusy-P PACKAGE
  BEHAVIOUR
    connectionBusy-B BEHAVIOUR
      DEFINED AS  !Contains attributes that count instances of local or remote
busy conditions!;;
  ATTRIBUTES
     localBusy           GET,
     remoteBusy          GET;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       connectionbusy(34)};


connectionReset-P PACKAGE
  BEHAVIOUR
    connectionReset-B BEHAVIOUR
      DEFINED AS
        !Contains attributes that count instances of particular types of reset!;;
  ATTRIBUTES
     remoteReset         GET,
     localReset          GET,
     providerReset       GET;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       connectionReset(35)};

connectionRoute-P PACKAGE
  BEHAVIOUR
    connectionRoute-B BEHAVIOUR
```



```
      DEFINED AS
        !This is a conditional part of the LLC Connection Object. It is present
        if source routing is used and the connection is cognizant of its route.
        This conditional package contains one attribute, the source route that the
        LSAP Pair uses to communicate. Source Routing is described in ISO/IEC
        10038 : 1993! ;;
  ATTRIBUTES
    route               GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       connectionroute(10)} ;

type2DISC-P PACKAGE
  BEHAVIOUR
    type2DISC-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of DISC PDUs either delivered or received from the
        MAC.! ;;
  ATTRIBUTES
    type2ReceivedDISC     GET ,
    type2SentDISC         GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2disc(11)} ;

type2DM-P PACKAGE
  BEHAVIOUR
    type2DM-B BEHAVIOUR
      DEFINED AS  !Specifies a count of DM PDUs either delivered or received from
the MAC.!;;
  ATTRIBUTES
    type2ReceivedDM       GET ,
    type2SentDM           GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2dm(12)} ;

type2FlowControl-P PACKAGE
  BEHAVIOUR
    type2FlowControl-B BEHAVIOUR
      DEFINED AS
        !This conditional package contains two attributes that are used to provide
        flow control.! ;;
  ATTRIBUTE
    kStep                   GET-REPLACE ,
    maxSendWindowSize       GET-REPLACE ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2flowcontrol(13)} ;

type2FRMR-P PACKAGE
  BEHAVIOUR
    type2FRMR-P-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of FRMR PDUs delivered to the MAC and a
        count of FRMR PDUs received from the MAC and not discarded.! ;;
  ATTRIBUTES
    type2ReceivedFRMR    GET ,
    type2SentFRMR        GET ;
  NOTIFICATIONS
    lLCConnection2Event-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2frmr(14)} ;

type2I-P PACKAGE
  BEHAVIOUR
    type2I-B BEHAVIOUR
      DEFINED AS
```

```
        !This conditional package counts the number of I PDUs received
        and sent (see  7.5.1, 7.5.6, 7.5.8, 7.5.9, and 7.8.4).!;;
  ATTRIBUTES
    type2ReceivedIGET,
    type2SentIGET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2i(15)} ;

type2Iack-P PACKAGE
  BEHAVIOUR
    type2Iack-B BEHAVIOUR
      DEFINED AS
         !Contains attributes that count acknowledgments sent and received!;;
  ATTRIBUTES
     sentAcks            GET ,
     receivedAcks        GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2iack(36)};

type2OptTolIPDUs-P PACKAGE
  BEHAVIOUR
    type2OptTolIPDUs-B BEHAVIOUR
      DEFINED AS
        !This optional package contains two attributes: a Boolean to indicate
        support for the Optional Toleration of Duplicate I PDUs function and
        an event counter to count the number of Duplicate I PDUs received.!;;
   ATTRIBUTES
     optionalTolerationIPDUsGET,
     duplicateIPDUsReceivedGET;
   NOTIFICATIONS
     lLCConnection2Event-N;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package (4)
        type2OptTolIPDUs (37)};

type2PDUs-P PACKAGE
  BEHAVIOUR
    type2PDUs-B BEHAVIOUR
      DEFINED AS
        !This conditional package contains attributes that count a number of
        different types of discarded PDUs.!;;
  ATTRIBUTES
    pDUsDiscarded1       GET ,
    pDUsDiscarded2       GET ,
    pDURetransmissions   GET ;
  NOTIFICATIONS
    lLCConnetion2Event-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2pdus(16)} ;

type2REJ-P PACKAGE
  BEHAVIOUR
    type2REJ-B BEHAVIOUR
      DEFINED AS
        !This conditional package contains two attributes to count sent and
        received REJ PDUs (see Clause 7.5.9).!;;
  ATTRIBUTES
    type2ReceivedREJ     GET ,
    type2SentREJ         GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2rej(17)} ;

type2RNR-P PACKAGE
  BEHAVIOUR
    type2RNR-B BEHAVIOUR
```



```
      DEFINED AS
        !This conditional package contains attributes that specify the
        count of RNR PDUs either delivered or received from the MAC.!;;
  ATTRIBUTES
    type2ReceivedRNR     GET ,
    type2SentRNR         GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2rnr(18)} ;

type2RR-P PACKAGE
  BEHAVIOUR
    type2RR-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of RR PDUs either delivered or received from the MAC.!;;
  ATTRIBUTES
    type2ReceivedRR      GET ,
    type2SentRR          GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2rr(19)} ;

type2SABME-P PACKAGE
  BEHAVIOUR
    type2SABME-B BEHAVIOUR
      DEFINED AS
        !This conditional package contains attributes that specify the
        count of SABME PDUs either delivered or received.!;;
  ATTRIBUTES
    type2ReceivedSABME   GET ,
    type2SentSABME       GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2sabme(20)};

type2UA-P PACKAGE
  BEHAVIOUR
    type2UA-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of UA PDUs either delivered or received from the MAC.!;;
  ATTRIBUTES
    type2ReceivedUA      GET ,
    type2SentUA          GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2ua(21)} ;

type2Violation-P PACKAGE
  BEHAVIOUR
    type2Violation-P-B BEHAVIOUR
      DEFINED AS
         !This conditional package provides an attribute and notification for
         type 2 violations.!;;
  ATTRIBUTES
    type2Violation       GET ;
  NOTIFICATIONS
    lLCConnetion2Event-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type2violation(22)} ;

-- Attributes --

dupIPDUsReceived ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    Definitions.EventCounter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    dupIPDUsReceived-B BEHAVIOUR
```

```
      DEFINED AS
        !Provides a count of the number of duplicate I PDUs received for this
        connection.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       dupIPDUsReceived(120)};

kStep ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.KStep ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    kStep-B BEHAVIOUR
      DEFINED AS  !The value of the flow control parameter! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       kstep(38)};

lLCConnection2Name ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.LLCConnectionName ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    lLCConnection2Name-B BEHAVIOUR
      DEFINED AS
        !The localLSAP and remoteLSAPId and the local and remote MACAddress! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcconnectionname2(37)};

localBusy ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
   localBusy-B BEHAVIOUR
     DEFINED AS  !Counts occurrences of entry into the busy condition!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       localBusy(112)};

localReset ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
   localReset-B BEHAVIOUR
     DEFINED AS
       !Counts instances of user request for reset while in a connected state!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       localReset(113)};

maximumRetransmissions ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaximumRetransmissions ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    maximumRetransmissions-B BEHAVIOUR
      DEFINED AS
        !The number of times that the LLC Type 2 should attempt to realize a
        successful PDU transfer! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumretransmissions(39)};

maxSendWindowSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.MaxSendWindowSize ;
  MATCHES FOR EQUALITY ;
```



```
  BEHAVIOUR
    maxSendWindowSize-B BEHAVIOUR
      DEFINED AS   !The maximum size of the send window! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maxSendWindowSize(40)};

optionalTolerationIPDUs ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    Definitions.OptTolIPDU;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    optionalTolerationIPDUs-B BEHAVIOUR
      DEFINED AS
        !Indicates if the Toleration of Duplicate I PDUs is on or off.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       optionalTolerationIPDUs(119)};

pDUsDiscarded1 ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    pDUsDiscarded1-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of PDUs discarded due to insufficient resources for
        this connection.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       pdusdiscarded1(41)};

pDUsDiscarded2 ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    pDUsDiscarded2-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of I PDUs discarded due to unexpected or invalid sequence
        number for this connection.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       pdusdiscarded2(42)};

pDURetransmissions ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    pDURetransmissions-B BEHAVIOUR
      DEFINED AS
        ! Specifies a count of the number of times the Type 2 LLC service
        retransmitted PDUs for this connection.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       pduretransmissions(43)};

providerReset ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
     LLCDefinitions.Counter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
   providerReset-B BEHAVIOUR
     DEFINED AS
        ! Counts instances of provider-initiated reset due to FRMR PDU sent, FRMR
        received, or retry exhaustion!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       providerReset(114)};
```

```
receivedAcks ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    receivedAcks-B BEHAVIOUR
       DEFINED AS  ! Counts I PDUs acknowledged by connection partner!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       receivedacks(115)};

receiveWindowSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.ReceiveWindowSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    receiveWindowSize-B  BEHAVIOUR
      DEFINED AS  !The value of the receive window! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       receivewindowsize(44)};

remoteBusy ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    remoteBusy-B BEHAVIOUR
      DEFINED AS
        ! Counts RNR PDUs received when the remote busy condition is not
        already set!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       remoteBusy(116)};

remoteReset ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
     LLCDefinitions.Counter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
    remoteReset-B BEHAVIOUR
      DEFINED AS
        !Counts instances of partner-initiated reset (SABME PDU received in a
        connected state)!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       remoteReset(117)};

route ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Route ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    route-B BEHAVIOUR
      DEFINED AS
          !This is a conditional part of the LSAP Pair Object. It is present if
          source routing is used and the LSAP Pair is cognizant of its route.  This
          attribute is used by the LSAP Pair to communicate. Source Routing is de-
          scribed in ISO/IEC 10038 : 1993! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       route(45)};

sendWindowSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.SendWindowSize ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    sendWindowSize-B BEHAVIOUR
```



```
      DEFINED AS     !The value of the send window! ; ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       sendwindowsize(46)};

sentAcks ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
     LLCDefinitions.EventCounter;
  MATCHES FOR EQUALITY;
  BEHAVIOUR
   sentAcks-B BEHAVIOUR
      DEFINED AS  !Counts acknowledgments sent for received I PDUs!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       sentAcks(118)};


type2AcknowledgeTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.AcknowledgeTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2AcknowledgeTimeoutValue-B BEHAVIOUR
      DEFINED AS
        !This attribute defines the time interval during which the LLC shall ex-
        pect to receive an acknowledgment to one or more outstanding I PDUs or an
        expected response PDU to a sent unnumbered command PDU. This value is mea-
        sured in milliseconds (see  7.4.1, 7.4.3, 7.4.5, 7.5, 7.5.5, 7.5.9, 7.6,
        7.8.1.1, 7.9.2, 7.9.2.2, and 7.9.2.3).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2acknowledgetimeoutvalue(47)};

type2BusyStateTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.BusyStateTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2BusyStateTimeoutValue-B BEHAVIOUR
      DEFINED AS
        !This attribute defines the time interval during which the LLC shall wait
        for an indication of the clearance of a busy condition at the other LLC.
        This time value is measured in milliseconds (see 7.5.7, 7.5.9, 7.8.1.4,
        7.9.2, 7.9.2.2, and 7.9.2.3).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2busystatetimeoutvalue(48)};

type2PBitTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.PBitTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2PBitTimeoutValue-B BEHAVIOUR
      DEFINED AS
        !The P-bit timer defines the time interval during which the LLC shall ex-
        pect to receive a PDU with the F bit set to "1" in response to a sent Type
        2 command with the P bit set to "1". This value is measured in seconds
        (see 7.5.9, 7.8.1.2, 7.9.2, 7.9.2.2, and 7.9.2.3).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2pbittimeoutvalue(49)};

type2ReceivedDISC ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedDISC-B BEHAVIOUR
```

```
      DEFINED AS
        !Specifies a count of DISC PDUs received from the MAC and not
        discarded.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receiveddisc(50)};

type2ReceivedDM ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedDM-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of DM PDUs received from the MAC and not
        discarded.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receiveddm(51)};

type2ReceivedFRMR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedFRMR-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of FRMR PDUs received from the MAC and not
        discarded. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedfrmr(52)};

type2ReceivedI ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedI-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of I PDUs received from the MAC and passed to the LLC
        user. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedI(53)};

type2ReceivedREJ ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedREJ-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of REJ PDUs received from the MAC and not discarded.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedrej(54)};

type2ReceivedRNR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedRNR-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of RNR PDUs received from the MAC and not discarded. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedrnr(55)};
```



```
type2ReceivedRR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedRR-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of RR PDUs received from the MAC and not discarded.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedrr(56)};

type2ReceivedSABME ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedSABME-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of SABME PDUs received from the MAC and not
        discarded.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedsabme(57)};

type2ReceivedUA ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2ReceivedUA-B BEHAVIOUR
      DEFINED AS
        !Specifies a count of UA PDUs received from the MAC and not discarded. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2receivedua(58)};

type2RejectTimeoutValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.RejectTimeoutValue ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2RejectTimeoutValue-B BEHAVIOUR
      DEFINED AS
        !The Reject timer defines the time interval during which the LLC shall
        expect to receive a reply to a sent REJ PDU. This value is measured in
        milliseconds  (see  7.5.4, 7.8.1.3, 7.9.2, 7.9.2.2, and 7.9.2.3).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2rejecttimeoutvalue(59)};

type2SentDISC ATTRIBUTE
  WITH ATTRIBUTE SYNTAX  LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentDISC-B BEHAVIOUR
      DEFINED AS  !Specifies a count of DISC PDUs delivered to the MAC. !;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentdisc(60)};

type2SentDM ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentDM-B BEHAVIOUR
      DEFINED AS  !Specifies a count of DM PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
```

```
       type2sentdm(61)};

type2SentFRMR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentFRMR-B BEHAVIOUR
      DEFINED AS   !Specifies a count of FRMR PDUs delivered to the MAC.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentfrmr(62)};

type2SentI ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentI-B BEHAVIOUR
      DEFINED AS  !Specifies a count of I PDUs delivered to the MAC!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentI(63)};

type2SentREJ ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentREJ-B BEHAVIOUR
      DEFINED AS  !Specifies a count of REJ PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentrej(64)};

type2SentRNR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentRNR-B BEHAVIOUR
      DEFINED AS  !Specifies a count of RNR PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentrnr(65)};

type2SentRR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentRR-B BEHAVIOUR
      DEFINED AS  !Specifies a count of RR PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentrr(66)};

type2SentSABME ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentSABME-B BEHAVIOUR
      DEFINED AS  !Specifies a count of SABME PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentsabme(67)};

type2SentUA ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
```



```
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2SentUA-B BEHAVIOUR
      DEFINED AS  !Specifies a count of UA PDUs delivered to the MAC.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type2sentua(68)};

type2Violation ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type2Violation-B BEHAVIOUR
      DEFINED AS
       !Provides a count of the number of protocol violations detected by this
        connection.  Protocol violations include the following:
        1) those that result in the LLC entity sending an FRMR frame
        2) those that result in a data link resetting action to be initiated by
        the receiving LLC entity itself.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
        type2violation(69)};

-- Actions --

correlatorExchange-AC ACTION
  BEHAVIOUR
    correlatorExchange-B BEHAVIOUR
      DEFINED AS
        !The action is used when two ends of a link connection report errors
        relating to the same problem. A correlator is reported in the events so
        that the managing process can assume that the errors are related to the
        same failure. This section describes the function that uses the CMIP
        protocol verb Confirmed Action to facilitate a correlator exchange between
        the two ends of the link connection. At the end of the link connection
        set-up (when the LSAP Pair has initially entered into LINK-OPENED state),
        a Manager will send a confirmed Correlator Exchange Action to the Manager
        of the adjacent link station. Upon receipt of this Action, the Manager
        will compute a correlator. Once this correlator is generated, it is
        returned to the initiating Manager. This correlator will be sent in events
        relating to the link connection. If the response is not received, no
        correlator is sent in the events. If the MAC address of the two ends of
        the link connection are the same, the link station that initiated the
        connection (sent SABME) should initiate the Correlator Exchange Action.!;;
   MODE    CONFIRMED ;
   WITH INFORMATION SYNTAX LLCDefinitions.CorrelatorData ;
   WITH REPLY SYNTAX LLCDefinitions.CorrelatorRspData ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       correlatorexchange(4)} ;

-- Notifications --

communicationAlarm-N NOTIFICATION
  BEHAVIOUR
    communicationAlarm-B BEHAVIOUR
      DEFINED AS
        !Defined in the ISO Standards - Alarm Reporting Function!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) notification(10)
       communicationalarm(1)} ;

lLCConnection2Event-N NOTIFICATION
  BEHAVIOUR
    lLCConnection2Event-B BEHAVIOUR
      DEFINED AS
```

```
        !This event will be sent for the following conditions:
            DupIPDUs Received
            PDUs Discarded1
            PDUs Discarded2
            PDU Retransmissions
            Type2 Violation
            Type2 Acknowledgment Timeout Value
            Type2 Reject Timeout Value
            Type2 Busy-State
            Timeout Value Type2
            P-Bit Timeout Value
            Retransmissions   !;;
  WITH INFORMATION SYNTAX   LLCDefinitions.LLCConnection2Event;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) notification(10)
       llcconnection2event(2)} ;
```

10.5 LLCConnection2IVMO managed object

A 1LCConnection2IVMO may be used to supply initial values for the attributes of 1LCConnection2 MOs. Different instances of lLCConnection2IVMO may contain different initial values.

```
lLCConnection2IVMO MANAGED OBJECT CLASS
DERIVED FROM "DMI : 1992":top;
CHARACTERIZED BY
  lLCConnection2-P,
  lLCConnection2IVMO-P;
CONDITIONAL PACKAGES
  connectionRoute-P
    PRESENT IF route known to LSAP Pair object,
  type2FlowControl-P
    PRESENT IF flow control is supported,
  type2I-P
    PRESENT IF the sent and receive I counters are supported,
  type2FRMR-P
    PRESENT IF the sent and receive FRMR counters are supported,
  type2RR-P
    PRESENT IF the sent and receive RR counters are supported,
  type2RNR-P
    PRESENT IF the sent and receive RR counters are supported,
  type2REJ-P
    PRESENT IF the sent and receive REJ counters are supported,
  type2SABME-P
    PRESENT IF the sent and receive SABME counters are supported,
  type2UA-P
    PRESENT IF the sent and receive UA counters are supported,
  type2DISC-P
    PRESENT IF the sent and receive DISC counters are supported,
  type2DM-P
    PRESENT IF the sent and receive DM counters are supported,
  type2PDUs-P
    PRESENT IF using PDUs event notification,
  type2OptTolIPDUs-P
    PRESENT IF counting received I PDUs recognized as duplicates is supported,
type2Violation-P
    PRESENT IF counting violations ;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectClass(3) llcconnection2ivmo(4)};

-- Name Bindings --

lLCConnection2IVMOB-NB NAME BINDING
SUBORDINATE OBJECT CLASS
  lLCConnection2IVMO AND SUBCLASSES;
```



```
NAMED BY SUPERIOR OBJECT CLASS
  lLCConnection2 AND SUBCLASSES;
WITH ATTRIBUTE  lLCConnection2IVMOName ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) namebinding(6)
       llcconnection2ivmonb(4)};

-- Packages --

lLCConnection2IVMO-P PACKAGE
  ATTRIBUTES
    lLCConnection2IVMOName      GET;
REGISTERED AS { iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcconnection2ivmop(23)};

-- Attributes --

lLCConnection2IVMOName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.GraphicString;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    lLCConnection2IVMOName-B
      DEFINED AS  !Naming attribute for the lLCConnection2IVMO managed object.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcconnection2ivmoname(70)};
```

10.6 LLCConnectionlessAck managed object

This clause contains those attributes, actions and event notifications associated with an LLC Type 3 (connectionless acknowledge) environment. The formal description is:

```
lLCConnectionlessAck MANAGED OBJECT CLASS
  DERIVED FROM "DML": lLCCOPM;
  CHARACTERIZED BY
    lLCConnectionlessAck-P ;
  CONDITIONAL PACKAGES
    type3Retransmissions-P
      PRESENT IF the Retransmissions counter is used with LLC Type 3 ,
    type3NoResponse-P
      PRESENT IF the NoResponse counter is used with LLC Type 3,
    type3Command-P
      PRESENT IF the sent and receive command counters are used with LLC Type 3 ,
    type3Response-P
      PRESENT IF the sent and receive command counters are used with LLC Type 3 ,
    type3Violation-P
      PRESENT IF violation detect counter is used with LLC Type 3 ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectclass(3) llcconnectionlessack(5)};

-- Name Bindings

lLCConnectionlessAck-NB NAME BINDING
SUBORDINATE OBJECT CLASS lLCConnectionlessAck AND SUBCLASSES;
NAMED BY
SUPERIOR OBJECT CLASS "DML": lLCCOPM  AND SUBCLASSES;
WITH ATTRIBUTE lLCConnectionlessAckName;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       llcconnectionlessacknb(5)} ;

-- Packages --

lLCConnectionlessAck-P PACKAGE
  BEHAVIOUR
```

```
    lLCConnectionlessAck-B BEHAVIOUR
      DEFINED AS
        !The LLC Connectionless Ack  object is responsible for processing the at-
        tributes, actions and notifications which affect a specific data link con-
        nection (for LLC Type 3 connections). One data connection shall exist for
        each data link connection supported.!;;
  ATTRIBUTES
    lLCConnectionlessAckName         GET,
    maximumLLCInformationFieldSize   GET ,
    maximumRetransmissions           GET-REPLACE ,
    tESTReceivedABBResponse          GET ,
    tESTReceivedCommand              GET ,
    tESTReceivedResponse             GET ,
    tESTSentABBResponse              GET ,
    tESTSentCommand                  GET ,
    tESTSentResponse                 GET ,
    type3ReceiveResources            GET ,
    uIReceived                       GET ,
    uISent                           GET ,
    xIDReceivedCommand               GET ,
    xIDReceivedResponse              GET ,
    xIDSentCommand                   GET ,
    xIDSentResponse                  GET ;
  ACTIONS
    tESTSentCommand-AC ,
    xIDSentCommand-AC ;
  NOTIFICATIONS
    communicationAlarm-N ,
    lLCConnection2Event-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcconnectionlessackp(24)} ;

type3Command-P PACKAGE
  BEHAVIOUR
    type3Command-P-B  BEHAVIOUR
      DEFINED AS
        !This conditional packages contains attributes (counters) that
        count received responses with the command status field set.!;;
  ATTRIBUTES
    type3CommandIP                   GET ,
    type3CommandIT                   GET ,
    type3CommandOK                   GET ,
    type3CommandPE                   GET ,
    type3CommandRS                   GET ,
    type3CommandUE                   GET ,
    type3CommandUN                   GET ,
    type3ReceivedACCommand           GET ,
    type3SentACCommand               GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type3command(25)} ;

type3NoResponse-P PACKAGE
  BEHAVIOUR
    type3NoResponse-P-B BEHAVIOUR
      DEFINED AS
        !Provides a count of failures to obtain a response from the destination
        associated with an LSAPPairComponent, after exhausting the retry counter
        Retransmission4. When this counter is updated an event notification is
        forwarded.! ;;
  ATTRIBUTES
    type3NoResponse   GET ;
  NOTIFICATIONS
    lLCClessACKEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
```



```
       type3noresponse(26)} ;

type3Response-P PACKAGE
  BEHAVIOUR
    type3Response-P-B  BEHAVIOUR
      DEFINED AS
        !This conditional packages contains attributes (counters) that
        count received responses with the response status field set.!;;
  ATTRIBUTES
    type3ResponseIP     GET ,
    type3ResponseIT     GET ,
    type3ResponseNE     GET ,
    type3ResponseNR     GET ,
    type3ResponseOK     GET ,
    type3ResponseRS     GET ,
    type3ResponseUE     GET ,
    type3ResponseUN     GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type3response(27)} ;

type3Retransmissions-P PACKAGE
  BEHAVIOUR
    type3Retransmissions-P-B BEHAVIOUR
      DEFINED AS
       !Provides a count of retransmissions that have occurred to the
       destination associated with this LSAP pair.! ;;
  ATTRIBUTES
    type3Retransmissions  GET ;
  NOTIFICATIONS
    lLCClessACKEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type3retransmissions(28)} ;

type3Violation-P PACKAGE
  BEHAVIOUR
    type3Violation-P-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the number of protocol violations detected by this
        connection.! ;;
  ATTRIBUTES
    type3Violation          GET ;
  NOTIFICATIONS
    lLCClessACKEvent-N ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       type3violation(29)} ;

-- Attributes

lLCConnectionlessAckName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.LLCConnectionlessAckName ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    lLCConnectionlessAckName-B BEHAVIOUR
      DEFINED AS
        !The localLSAP and remoteLSAPId and the local and remote MAC Address! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcconnectionlessackname2(71)};

type3CommandIP ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
```

```
    type3CommandIP-B BEHAVIOUR
      DEFINED AS
         !Provides a count of the received responses with the command status field
         set to indicate "Permanent implementation dependent error," as defined in
         5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandip(72)};

type3CommandIT ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandIT-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "Temporarily implementation dependent error," as defined
        in 5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandit(73)};

type3CommandOK ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandOK-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "Command accepted," as defined in 5.4.3.2. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandok(74)};

type3CommandPE ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandPE-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "Protocol error," as defined in 5.4.3.2. !;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandpe(75)};

type3CommandRS ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandRS-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "Unimplemented or inactivated service," as defined in
        5.4.3.2. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandrs(76)};

type3CommandUE ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandUE-B BEHAVIOUR
```



```
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "LLC user interface error," as defined in 5.4.3.2. ! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3commandue(77)};

type3CommandUN ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3CommandUN-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the command status field
        set to indicate "Resources temporarily unavailable," as defined in
        5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
type3commandun(78)};

type3NoResponse ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3NoResponse-B BEHAVIOUR
      DEFINED AS
        !Provides a count of failures to obtain a response from the destination
        associated with this LSAP pair, after exhausting the retry counter
        MaximumRetransmissions4.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3noresponse(79)} ;

type3ReceiveResources ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Type3ReceiveResources ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ReceiveResources-B BEHAVIOUR
      DEFINED AS
        !Causes the LSAP entity to either enable or disable the resources for
        reception of the data field of Type 3 command PDUs sent to this LSAP.
        If a Type 3 PDU containing data is received while disabled, the UN status
        code is returned in the CCCC field of the response PDU.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3receiveresources(80)};

type3ReceivedACCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ReceivedACCommand-B BEHAVIOUR
      DEFINED AS
        !Provides a count of AC command PDUs received from the other LSAP of the
        connection (LSAP PAIR).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3receivedaccommand(81)};

type3SentACCommand ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3SentACCommand-B BEHAVIOUR
```

```
      DEFINED AS
        !Provides a count of AC command PDUs sent to the other LSAP of the
        connection (LSAP PAIR).! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3sentaccommand(82)};

type3ResponseIP ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseIP-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Permanent implementation dependent error," as defined in
        5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responseip(83)};

type3ResponseIT ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseIT-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Temporary implementation dependent error," as defined in
        5.4.3.2.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responseit(84)};

type3ResponseNE ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseNE-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Response LSDU never submitted," as defined in 5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responsene(85)};

type3ResponseNR ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseNR-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Response LSDU not requested," as defined in 5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responsenr(86)};

type3ResponseOK ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseOK-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
```



```
        set to indicate "Response LSDU present," as defined in 5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responseok(87)};

type3ResponseRS ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseRS-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Unimplemented or inactivated service," as defined in
        5.4.3.2.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responsers(88)};

type3ResponseUE ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseUE-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "LLC user interface error," as defined in 5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responseue(89)};

type3ResponseUN ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.Counter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3ResponseUN-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the received responses with the response status field
        set to indicate "Resources temporarily unavailable," as defined in
        5.4.3.2.!  ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3responseun(90)};

type3Retransmissions ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3Retransmissions-B BEHAVIOUR
      DEFINED AS
        !Provides a count of retransmissions that have occurred to the
        destination associated with this LSAP pair.! ;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       type3retransmissions(91)};

type3Violation ATTRIBUTE
  WITH ATTRIBUTE SYNTAX
    LLCDefinitions.EventCounter ;
  MATCHES FOR EQUALITY ;
  BEHAVIOUR
    type3Violation-B BEHAVIOUR
      DEFINED AS
        !Provides a count of the number of protocol violations detected by this
        connection.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
```

```
       type3violation(92)};

-- Notifications --

lLCClessACKEvent-N NOTIFICATION
  BEHAVIOUR
    lLCClessACKEvent-B BEHAVIOUR
      DEFINED AS
        !This event will be sent for the following conditions:
             Retransmissions
             NoResponse
             Type3 Violation  ! ;;
  WITH INFORMATION SYNTAX LLCDefinitions.LLCConnection2Event;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) notification(10)
       llcclessackevent(3)} ;
```

10.7 LLCConnectionlessAckIVMO managed object

An lLCConnectionlessAckIVMO may be used to supply initial values for the attributes of lLCConnectionlessACK MOs. Different instances of lLCConnectionlessAckIVMO may contain different initial values.

```
lLCConnectionlessAckIVMO MANAGED OBJECT CLASS
  DERIVED FROM "DMI : 1992":top;
  CHARACTERIZED BY
    lLCConnectionlessAck-P,
    lLCConnectionlessAckIVMO-P;
  CONDITIONAL PACKAGES
    type3Retransmissions-P
      PRESENT IF the retransmissions counter is supported,
    type3NoResponse-P
      PRESENT IF the NoResponse counter is supported,
    type3Command-P
      PRESENT IF the sent and receive command counters are supported,
    type3Response-P
      PRESENT IF the sent and receive responses counters are supported,
    type3Violation-P
      PRESENT IF violation detection counter is supported;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectClass(3) llcconnectionlessAckIVMO(6)};

-- Name Bindings --

lLCConnectionlessAckIVMO-NB NAME BINDING
SUBORDINATE OBJECT CLASS
  lLCConnectionlessAckIVMO AND SUBCLASSES;
NAMED BY SUPERIOR OBJECT CLASS
  lLCConnectionlessAck AND SUBCLASSES;
  WITH ATTRIBUTE  lLCConnectionlessAckIVMOName;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) namebinding(6)
       llcconnectionlessackivmonb(6)};

-- Packages --

lLCConnectionlessAckIVMO-P PACKAGE
  ATTRIBUTES
    lLCConnectionlessAckIVMOName     GET ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       llcconnectionlessackivmo(30)};

-- Attributes --

lLCConnectionlessAckIVMOName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.GraphicString;
```



```
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      lLCConnectionlessAckIVMOName-B BEHAVIOUR
        DEFINED AS
        !Naming attribute for the LLCConnectionlessAckIVMO managed object.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       llcconnectionlessackivmoname(93)};
```

10.8 RDE setup managed object

A rDESetup managed object instance is instaniated if an instance of the lLCSAP object enables RDE.

```
rDESetup MANAGED OBJECT CLASS
  DERIVED FROM "DMI : 1992":top;
  CHARACTERIZED BY
    rDESetup-P;
      BEHAVIOUR
        rDESetup-B BEHAVIOUR
          DEFINED AS
            !A rDESetup managed object instance is instaniated
            if an instance of the lLCSAP object enables RDE.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectClass(3) rdesetup(7)};

-- Name Bindings --

rDESetup-NB NAME BINDING
SUBORDINATE OBJECT CLASS
  rDESetup AND SUBCLASSES;
NAMED BY SUPERIOR OBJECT CLASS
  lLCSAP AND SUBCLASSES;
  WITH ATTRIBUTE  rDESetupName;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) namebinding(6)
       rdesetupnb(7)};

-- Packages --

rDESetup-P PACKAGE
  ATTRIBUTES
    agingEnabled                GET-REPLACE,
    agingValue                  GET-REPLACE,
    enableType2Reset            GET-REPLACE,
    maximumRouteDescriptors     GET-REPLACE,
    maximumResponseTime         GET-REPLACE,
    minimumPDUSize              GET-REPLACE,
    rDEHold                     GET-REPLACE,
    rDEReplace                  GET-REPLACE,
    rDESetupName                GET,
    resetOnTestEnabled          GET-REPLACE;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       rdesetupp(31)};


-- Attributes --

agingEnabled ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.AgingEnabled;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      agingEnabled-B BEHAVIOUR
        DEFINED AS
          !A BOOLEAN value representing the policy that, when true,
          allows a manager to age a route.!;;
```

```
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       agingEnabled(94)};

agingValue ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.AgingValue;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      agingValue-B BEHAVIOUR
        DEFINED AS
          !A value used to determine if a route should be flushed due to inactivity
          for specified time period.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       agingValue(95)};

maximumRouteDescriptors ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.MaximumRouteDescriptors;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      maximumRouteDescriptors-B BEHAVIOUR
        DEFINED AS
          !The maximum number or route descriptors that an acceptable route
          may contain.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumRouteDescriptors(96)};

maximumResponseTime ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.MaximumResponseTime;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      maximumResponseTime-B BEHAVIOUR
        DEFINED AS
           !The timer value used to limit the maximum round trip response time
           allowed for route discovery and consequently the maximum time a PDU can
           remain in the system.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       maximumResponseTime(97)};

minimumPDUSize ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.MinimumPDUSize;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      minimumPDUSize-B BEHAVIOUR
        DEFINED AS
          !The minimum number of octets that must be supported by an acceptable
          route as indicated by the LF bits of the routing information field.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       minimumPDUSize(98)};

rdeHold ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.RDEHold;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      rDEHold-B BEHAVIOUR
        DEFINED AS
          !A BOOLEAN value representing the policy that, when true,
          requires RDE to withhold sending PDUs while changing routes.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       rdehold(99)};

rdeReplace ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.RDEReplace;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      rDEReplace-B BEHAVIOUR
```



```
        DEFINED AS
          !A BOOLEAN value representing the policy that, when true,
          enables the RDE to use the route selected by the remote station.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       rdereplace(100)};

rDESetupName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.GraphicString;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      rDESetupName-B BEHAVIOUR
        DEFINED AS  !Naming attribute for the rDESetup managed object.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       rdesetupname(101)};

resetOnTestEnabled ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.ResetOnTestEnabled;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      resetOnTestEnabled-B BEHAVIOUR
        DEFINED AS
          !This attribute is a BOOLEAN value. If the attribute equals TRUE, then
          if a test frame is received the source route is reset.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       resetOnTestEnabled(102)};
```

10.9 1RDE pair managed object

An instance of this managed object will exist for each RDE pair used by any type of LLC service provided. This managed object contains the local MAC address and local SAP and the remote MAC address and remote SAP, plus counters.

```
rDEPair MANAGED OBJECT CLASS
  DERIVED FROM "DMI :1992":top;
  CHARACTERIZED BY
    rDEPair-P;
      BEHAVIOUR
        rDEPair-B BEHAVIOUR
          DEFINED AS
            !An instance of this managed object will exist for each RDE pair used
            by any type of LLC service provided. This managed object contains the
            local MAC address and local SAP and the remote MAC address and remote
            SAP, plus counters.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032)
       managedObjectClass(3) rdepair(8)};

-- Name Bindings --

rDEPair-NB NAME BINDING
SUBORDINATE OBJECT CLASS
  rDEPair AND SUBCLASSES;
NAMED BY SUPERIOR OBJECT CLASS
  rDESetup AND SUBCLASSES;
  WITH ATTRIBUTE  rDEpair;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) namebinding(6)
       rdepairnb(8)};

-- Packages --

rDEPair-P PACKAGE
    ATTRIBUTES
      rDEPairName          GET-REPLACE,
```

```
      discardCounter        GET-REPLACE,
      nSRPDUCounter         GET-REPLACE,
      nSRSelectedCounter    GET-REPLACE,
      rIF                   GET-REPLACE,
      sRFPDUCounter         GET-REPLACE,
      queryCounter          GET-REPLACE;
    ACTIONS
      resetRoute,
      readCounters;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) package(4)
       rdepairp(32)};

-- Attributes --

rDEPairName ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.RDEPairId;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      rDEPair-A-B BEHAVIOUR
        DEFINED AS
          !The local LSAP and remote LSAPId and the local and remote
          MAC Addresses.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       rdepaira(103)};

discardCounter  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.Counter;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      discardCounter-B BEHAVIOUR
        DEFINED AS
          !A value that represents the number of times a new selection has been
          made.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       discardcounter(104)};

nSRPDUCounter  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.Counter2;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      nSRPDUCounter-B BEHAVIOUR
        DEFINED AS
          !A value that represents the number of frames without a routing field.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       nsrpducounter(105)};

nSRSelectedCounter  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.Counter;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      nSRSelectedCounter-B BEHAVIOUR
        DEFINED AS
          !A value that represents the number of times that acceptable
          SRF route existed resulting in the selection of the NSR path.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       nsrselectedcounter(106)};

rIF  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.RIF;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      rIF-B BEHAVIOUR
        DEFINED AS
          !The routing information field varies in length up to 30 octets.  When
```



```
          source routing is not used the RIF is said to be NULL and contains no
          information. The first bit of the MAC source address is defined as the
          Routing Information Indicator (RII). When the RIF is null, the RII is set
          to 0. When the RIF is not null, the RII is set to 1. The first two octets
          are the Rounting Control field (RC) and the remainder of the RIF contains
          Routing Descriptors (RD). Each RD is two octets in length, therefore a
          maximum of 14 RDs can exist in an RIF.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)  rif(107)};

sRFPDUCounter  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.Counter2;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      sRFPDUCounter-B BEHAVIOUR
        DEFINED AS
          !A value that represents the number of frames with routing field.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       sRFPDUcounter(108)};

queryCounter  ATTRIBUTE
  WITH ATTRIBUTE SYNTAX LLCDefinitions.Counter;
  MATCHES FOR EQUALITY ;
    BEHAVIOUR
      queryCounter-B BEHAVIOUR
        DEFINED AS
          !A value that represents the number of Route Query commands
          generated.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)
       querycounter(109)};

-- Actions --

readCounters-AC  ACTION
  BEHAVIOUR
    readCounters-B BEHAVIOUR
        DEFINED AS
          !This action will be sent when management request information concerning
          RDE counters.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       readcounters(5)};

flushRoute-AC ACTION
  BEHAVIOUR
    flushRoute-B BEHAVIOUR
        DEFINED AS
          !This action will be sent when management request an RDE Pair to flush
          its route. This does not cause the RDE to discover a new route.!;;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) action(9)
       flushroute(6)};
```

10.10 Resource Type ID managed object

This is the name binding for the Resource Type Id managed object that is defined in IEEE Std 802.1F-1993 and incorporated in ISO/IEC 10742 : 1994.

```
resourceTypeId-NB NAME BINDING
SUBORDINATE OBJECT CLASS
  resourceTypeId AND SUBCLASSES ;
 NAMED BY SUPERIOR OBJECT CLASS
   IEEE802-1.oResourceTypeId AND SUBCLASSES
 WITH ATTRIBUTE      lLCName ;
REGISTERED AS {iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)
       resourcetypeid(9)};
```

10.11 Conformance

10.11.1 Conformance requirements to this International Standard

An implementation for which conformance to this International Standard as a managed implementation is claimed shall

1) Support the LLC Data Link Entity managed object (lLCDLE) according to ISO/IEC 10742 : 1994,
2) For each supported MO, support at least one name binding defined in this part of 8802-2, for which the MO is the subordinate.

10.11.2 Protocol specific conformance requirements

10.11.2.1 Connectionless-mode LLC

An implementation claiming conformance to the management operation of an ISO/IEC 8802-2 connectionless-mode LLC as a managed implementation shall

1) Conform to ISO/IEC 10742 : 1994 and ISO/IEC 8802-2 as defined above,
2) Support the dLSAP MO and the lLCCLPM MO.

10.11.2.2 Connection-mode LLC

An implementation claiming conformance to the management operation of an ISO/IEC 8802-2 connection-mode LLC as a managed implementation shall

1) Conform to ISO/IEC 10742 and ISO/IEC 8802-2 as defined above,
2) Support the dLSAP MO, the rDESetup MO and at least one class derived from the lLCCOPM MO.

10.12 ASN.1 LLCDefinitions

```
IEEE802-2-LLCDefinitions {iso(1) member-body(2) us(840) ieee802-2(10032)
       llcdefinitions(1) version1(0)}

 DEFINITIONS ::= BEGIN

 IMPORTS
   MACAddress, ResourceInfo, ManufacturerOUI, ManufacturerName,
   ManufacturerProductName, ManufacturerProductVersion
 FROM  IEEE802CommonDefinitions {iso(1) member-body(2) us(840)
           ieee802-1F(10011) asn1Module(2) commondefinitions(0) version1(0)}
 ;  -- End of IMPORTS

 AcknowledgeTimeoutValue ::= INTEGER

 AgingEnabled ::= BOOLEAN

 AgingValue ::= INTEGER

 AvgBufferUseSize ::= INTEGER

 BufferSize ::= INTEGER

 BusyStateTimeoutValue::= INTEGER

 CorrelatorData ::= LSAPPairName
```



```
CorrelatorRspData ::= OCTET STRING

Counter ::= SEQUENCE {
   counterInitTime [0] TimeStamp,
   counterValue    [1]  CounterN }

CounterN ::= INTEGER  -- by convention, range 0 to +2(**N)-1 note that
                      -- ASN.1 encodes the INTEGER type as
                      -- a signed number

Counter2 ::= INTEGER   -- 64 bit counter

EnabledType2Reset ::= BOOLEAN

EventCounter ::= SEQUENCE {
  thresholdValue  [0] IMPLICIT CounterN,
  resetValue  [1]  IMPLICIT CounterN,
  thresholdInitTime [2] TimeStamp,
  counterValue  [3] IMPLICIT CounterN,
  counterInitTime [4] TimeStamp }

FlushRIF ::= LSAPPairID

KStep ::= INTEGER

LLCClessName ::= GraphicString

LLCConnectionlessAckName ::= GraphicString

LLCConnectionName ::= GraphicString

LLCConnection2Event ::= CHOICE {
   pdusDiscarded1      [1] CounterN,
   pdusDiscarded2      [2] CounterN,
   pduRetransmissions  [3] CounterN,
   acknowledgeTimeout  [4] AcknowledgeTimeoutValue,
   busyStateTimeout    [5] BusyStateTimeoutValue,
   rejectTimeout       [6] RejectTimeoutValue,
   pBitTimeout         [7] PBitTimeoutValue,
   type2Violation      [8] CounterN,
   retransmissions     [10] CounterN }

LLCClessACKEvent ::= CHOICE {
   type3Violation [0] CounterN,
   retransmissions [1] CounterN,
   noResponse [2] CounterN  }

LLCName ::= GraphicString

LLCSAPName ::= GraphicString

LLCServices ::= BIT STRING {
   type1 (0),
   type2 (1),
   type3Initiate (2),
   type3ReceiveData (3),
   type3ReturnData (4)  }

LLCStationEvent ::= CHOICE {
   pdusDiscarded        [0] CounterN,
   bufferProblems       [1] NULL }

LocalTime ::= SEQUENCE {
   numberOfStationRestarts [0] IMPLICIT INTEGER,
```

```
   timeSinceLastRestart    [1] IMPLICIT CounterN } --N=15

LSAP ::= OCTET STRING

LSAPId ::= INTEGER   -- 0 - 127

LSAPPairID ::= SEQUENCE {
   localLSAP [0] IMPLICIT LSAP,
   localAddress [1] IMPLICIT MACAddress,
   remoteLSAP [2] IMPLICIT LSAP,
   remoteAddress [3] IMPLICIT MACAddress }

LSAPPairName ::= SEQUENCE {
   lsapId    [0] LSAPId,  -- remote LSAPId
   mACAddr [1] MACAddress }  -- remote MAC Address

 --   MACAddress ::= OCTET STRING  minimum length 6 octets

MACPriority ::= INTEGER  -- 0 to 7

MACServiceClass ::= INTEGER  {
   requestWithNoResponse(0),
   requestWithResponse(1),
   response(2)  }

MaxBufferUseSize ::= INTEGER

MaximumLLCInformationFieldSize ::= INTEGER

MaximumLSAPsConfigured ::= INTEGER

MaxSendWindowSize ::= INTEGER

MaximumPDUN3 ::= INTEGER

MaximumRetransmissions ::= INTEGER

MaximumRetryCount ::= INTEGER

MaximumRouteDescriptors ::= INTEGER

MaximumResponseTime ::= INTEGER

MinimumPDUSize ::= INTEGER

NumberOfActiveLSAPs ::= INTEGER
  OptTolIPDU ::= BOOLEAN

PBitTimeoutValue ::= INTEGER

RDEEnabled ::= BOOLEAN

RDEHold ::= BOOLEAN

RDEReplace ::= BOOLEAN

ReadCounters ::= SEQUENCE {
   discardCounter     [0] IMPLICIT CounterN,
   nsrPDUCounter      [1] IMPLICIT Counter2,
   rsrSelectCounter   [2] IMPLICIT CounterN,
   srfPDUCounter      [3] IMPLICIT Counter2,
   queryCounter       [4] IMPLICIT CounterN }

ReceiveVariableLifetime ::= INTEGER
```

```
ReceiveWindowSize ::= INTEGER

ReinitializeData ::= NULL

ReinitializeResponse ::= NULL

RejectTimeoutValue ::= INTEGER

ResetOnTestEnabled ::= BOOLEAN

RIF ::= SEQUENCE {
  routingControl  [0] RoutingControl,
  routingDescriptors [1] SET OF RoutingDescriptors }

Route ::= SEQUENCE {
   routingControl   [0] IMPLICIT RoutingControl,
   routingDescriptors [1] IMPLICIT RoutingDescriptors }

RoutingControl ::=  OCTET STRING(SIZE(2))

RoutingDescriptors ::=  OCTET STRING(SIZE(2))

SAPDATA ::= LLCServices

SendWindowSize ::= INTEGER

Status ::= INTEGER  {
   up     (0),
   down   (1),
   other  (2)  }

STRIndicator ::= BIT STRING  {
   nsr   (0),
   ste   (1) }

SupportedServicesTypes ::= LLCServices

TESTSentCommand ::= TESTType

TESTType ::=  SEQUENCE {
   sourceLSAP     [0] IMPLICIT LSAPId,
   sourceMAC [1] IMPLICIT MACAddress OPTIONAL,
   -- determined by MAC when TEST is sent

   destinationLSAP [2] IMPLICIT LSAPId,
   destinationMAC  [3] IMPLICIT MACAddress,
   priority [4] IMPLICIT MACPriority OPTIONAL,
   serviceClass [5] IMPLICIT MACServiceClass OPTIONAL,
   information [6] IMPLICIT OCTET STRING  }

TimeStamp ::= CHOICE {
   generalTime [0] GeneralizedTime,
   utcTime     [1] UTCTime,
   localTime   [2] LocalTime }

TransmitVariableLifetime ::= INTEGER
Type3AcknowledgeTimeoutValue ::= INTEGER
Type3ReceiveResources ::= BOOLEAN
UserData ::= ANY
XIDSentCommand ::= XIDType
XIDType ::=  SEQUENCE {
   sourceLSAP [0] IMPLICIT LSAPId,
   sourceMAC [1] IMPLICIT MACAddress OPTIONAL,
   -- determined by MAC when TEST is sent
```

```
    destinationLSAP [2] IMPLICIT LSAPId,
    destinationMAC  [3] IMPLICIT MACAddress,
    priority  [4] IMPLICIT MACPriority OPTIONAL,
    serviceClass [5] IMPLICIT MACServiceClass OPTIONAL,
    typesSupported  [6] IMPLICIT LLCServices OPTIONAL,
    sendWindowSize  [7] IMPLICIT SendWindowSize OPTIONAL }

 VersionNumber ::= INTEGER

END
```

Annex A[1]

(normative)

Protocol Implementation Conformance Statement (PICS) proforma

A.1 Introduction

The supplier of a protocol implementation that is claimed to conform to ISO/IEC 8802-2 : 1998 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 initially 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 Abbreviations and special symbols

A.2.1 Status symbols

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
X	Prohibited
<pred>:	Conditional item symbol, including predicate identification (see A.3.4), applicable to a particular item
<pred>::	Conditional item symbol, including predicate identification (see A.3.4), applicable to a table or a group of tables
<item>:	Conditional symbol, status is dependent on the support marked for <item> (see A.3.4)

A.2.2 General abbreviations

N/A	Not applicable
PICS	Protocol Implementation Conformance Statement

[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.2.3 Item references

The following is a list of item references used in the PICS proforma:

Major capabilities:

CLS	Class of LLC supported
RDE	Route Determination Entity

LLC Type 1:

MIS	Miscellaneous protocol features
TES	TEST PDUs
TSR	Parameters in received TEST PDUs
TST	Parameters in transmitted TEST PDUs
UI	UI PDUs
UIR	Parameters in received UI PDUs
UIT	Parameters in transmitted UI PDUs
XDR	Parameters in received XID PDUs
XDT	Parameters in transmitted XID PDUs
XID	XID PDUs

LLC Type 2:

DIC	DISC PDUs
DMR	DM PDUs
FRR	FRMR PDUs
IC	I_CMD PDUs
IP	I PDUs
IR	I_RSP PDUs
MIS	Miscellaneous protocol features
PPA	Protocol parameters
PPR	Parameters in received PDUs
PPT	Parameters in transmitted PDUs
PRS	Protocol procedures
RJC	REJ_CMD PDUs
RJR	REJ_RSP PDUs
RNC	RNR_CMD PDUs
RNR	RNR_RSP PDUs
RRC	RR_CMD PDUs
RRR	RR_RSP PDUs
SAC	SABME PDUs
UAR	UA PDUs

LLC Type 3:

AnC	ACn command PDUs
A0C	AC0_CMD PDUs
A0P	Parameters in received AC0 PDUs
A0R	AC0_RSP PDUs
A0T	Parameters in transmitted AC0 PDUs
A1C	AC1_CMD PDUs
A1P	Parameters in received AC1 PDUs
A1R	AC1_RSP PDUs

A1T	Parameters in transmitted AC1 PDUs
MIS	Miscellaneous protocol features
PPA	Protocol parameters
PRS	Protocol procedures

Route Determination Entity (RDE):

AGEE	Ageing Enabled indicator
AGEF	Ageing Function
F	Parameters in transmitted RDE PDUs
HLD	Route Determination Parameters—Hold
IRPE	Invalid RDE Frames Counter
ISRF	Specifically Routed Frames Counter
IST	Spanning Tree Routed Frames Counter
MNPD	Route Selection Parameters—Minimum PDU Size
MXRD	Route Selection Parameters—Maximum Number of Route Descriptors
MXRT	Route Selection Parameters—Maximum Response Time
RASM	Route Administration
RCR	Route Control Receive specification
RCS	Route Control Send specification
RDEE	Route Determination Parameters—RDE Enabled
RDEP	Discard PDUs on receipt
ROT	Reset on Test
RPL	Route Determination Parameters—Replace
RQC	ROUTE_QUERY_COMMAND PDUs
RQR	ROUTE_QUERY_RESPONSE PDUs
RRST	Route Reset Procedures
RS	ROUTE_SELECTED PDUs
RSSM	Route Selection
STR	Spanning Tree Route parameters
TRR	Route Response Timer
TRS	Route Selection Timer

A.3 Instructions for completing the PICS proforma

A.3.1 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, divided into subclauses, each containing a number 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 N/A for Not Applicable) or by entering a value or a set or range of values.

NOTE—There are some items for which two or more choices from a set of possible answers can apply. All relevant choices are to be marked in these cases.

Each item is identified by an item reference in the first column. The second column contains the questions to be answered. The third column contains the reference or references to the material that specifies the item in the main body of ISO/IEC 8802-2 : 1998. 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.3.4.

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 labeled A<*i*> or X<*i*>, respectively, for cross-referencing purposes, where <*i*> is an unambiguous identification for the item (e.g., simply a numeral). There are no other restrictions on its format or 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, 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.

A.3.2 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 of information will be supplied and a PICS can be considered complete without any such information. Examples of Additional Information might 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 upon specific application requirements, for the exclusion of features that, although optional, are commonly present in implementations of the ISO/IEC 8802-2 : 1998 protocol.

References to items of Additional Information may be entered next to any answer in the questionnaire, and may be included in items of Exception Information.

A.3.3 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 preprinted 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 Information item itself.

An implementation for which an Exception Information item is required in this way does not conform to ISO/IEC 8802-2 : 1998.

NOTE—A possible reason for the situation described above is that a defect in ISO/IEC 8802-2 : 1998 has been reported, a correction for which is expected to change the requirement not met by the implementation.

A.3.4 Conditional status

A.3.4.1 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.

Where a group of items is subject to the same condition for applicability, a separate preliminary question about the condition appears at the head of the group, with an instruction to skip to a later point in the questionnaire if the "Not Applicable" answer is selected. Otherwise, individual conditional items are indicated by one or more conditional symbols (on separate lines) in the status column.

A conditional symbol is of the form "<pred>:<s>" where "<pred>" is a predicate as described in A.3.4.2 below, and "<s>" is one of the status symbols M, O, O.<*n*>, or X.

If the value of the predicate in any line of a conditional item is true (see A.3.4.2), then 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, the Not Applicable (N/A) answer is to be marked in the relevant line. Each line in a multi-line conditional item should be marked: at most one line will require an answer other than N/A.

A conditional symbol of the form "<pred>::" where "<pred>" is a predicate as described in A.3.4.2 below, may precede a table or a group of tables in a clause or a subclause. If the value of the predicate is true, answers shall be marked in the table or group of tables. Otherwise, the table or group of tables shall be skipped.

A.3.4.2 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) A predicate name, for a predicate defined elsewhere in the PICS proforma (usually in the Major Capabilities section or at the end of the section containing the conditional item): see below.
c) The logical negation symbol "Ø" prefixed to an item-reference or predicate name: the value of the predicate is true if the value of the predicate formed by omitting the "Ø" symbol is false, and vice versa.

A.3.4.2.1 The definition for a predicate name

The definition for a predicate name is one of the following:

a) An item-reference, evaluated as at A.3.4.2a), or
b) A relation containing a comparison operator (=, >, etc.) with at least one of its operands being an item-reference for an item taking numerical values as its answer; the predicate is true if the relation holds when each item-reference is replaced by the value entered in the support column as answer to the item referred to, or
c) A boolean expression constructed by combining simple predicates, as in a) and b), 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.

Each item whose reference is used in a predicate or predicate definition is indicated by an asterisk in the Item column.

A.3.5 Identification of requirements

The information in the PICS proforma does not supersede or augment the conformance requirements in the main body of ISO/IEC 8802-2 : 1998.

A.4 Identification

A.4.1 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 system(s), system names	

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 etc.).

A.4.2 Protocol summary, ISO/IEC 8802-2 : 1998 [ANSI/IEEE Std 802.2, 1998 Edition]

Identification of Protocol Standard	ISO/IEC 8802-2 : 1998 [ANSI/IEEE Std 802.2, 1998 Edition]
Identification of Amendments and Corrigenda to this PICS proforma that have been completed as part of this PICS	ISO/IEC 8802-2 : 1998 [ANSI/IEEE Std 802.2, 1998 Edition] Amd. : Corr. : Amd. : Corr. :
Have any Exception items been required (see A.3.3)? Yes ☐ No ☐ (The answer Yes means that the implementation does not conform to ISO/IEC 8802-2 : 1998 [ANSI/IEEE Std 802.2, 1998 Edition]).	

Date of statement (dd/mm/yy)	

A.5 Major capabilities

Item	Protocol feature	References	Status	Support		
*CLS1a	Is Class I LLC supported?	4.2	O.1		Yes ☐	No ☐
CLS1b	Are LLC Type 1 procedures supported?	4.2	CLS1a:M	N/A ☐	Yes ☐	
*CLS2a	Is Class II LLC supported?	4.2	O.1		Yes ☐	No ☐
CLS2b	Are LLC Type 1 and Type 2 procedures supported?	4.2	CLS2a:M	N/A ☐	Yes ☐	
*CLS3a	Is Class III LLC supported?	4.2	O.1		Yes ☐	No ☐
CLS3b	Are LLC Type 1 and Type 3 procedures supported?	4.2	CLS3a:M	N/A ☐	Yes ☐	
*CLS4a	Is Class IV LLC supported?	4.2	O.1		Yes ☐	No ☐
CLS4b	Are LLC Type 1, Type 2, and Type 3 procedures supported?	4.2	CLS4a:M	N/A ☐	Yes ☐	
*RDE	Is Route Determination supported?	9	O		Yes ☐	No ☐

A.6 LLC Type 1 operation—Unacknowledged connectionless-mode

CLS1a *or* CLS2a *or* CLS3a *or* CLS4a::

All tables in clause A.6 are to be completed if above predicate evaluates to true.

A.6.1 LLC Type 1—Supported PDU types

Item	Protocol feature/ Supported PDU types	References	Status	Support		
UI/1	UI_CMD supported on transmission	6.1, 6.5.1	M		Yes ☐	
UI/2	UI_CMD supported on receipt	6.1, 6.5.2	M		Yes ☐	
*XID/3	XID_CMD supported on transmission	6.6	O		Yes ☐	No ☐
XID/4	XID_CMD supported on receipt	6.6	M		Yes ☐	
XID/5	XID_RSP supported on transmission	6.6	M		Yes ☐	
XID/6	XID_RSP supported on receipt	6.6	XID/3:M	N/A ☐	Yes ☐	
*TES/7	TEST_CMD supported on transmission	6.7	O		Yes ☐	No ☐
TES/8	TEST_CMD supported on receipt	6.7	M		Yes ☐	
TES/9	TEST_RSP supported on transmission	6.7	M		Yes ☐	
TES/10	TEST_RSP supported on receipt	6.7	TES/7:M	N/A ☐	Yes ☐	

A.6.2 LLC Type 1—Supported parameters in PDUs on transmission

Item	Protocol feature/ Supported parameters on transmission	References	Status	Support		
UIT/11	UI_CMD—DSAP address	6.2	M		Yes ☐	
UIT/12	UI_CMD—SSAP address	6.2	M		Yes ☐	
UIT/13	UI_CMD—P-bit = 0	6.3	M		Yes ☐	
UIT/14	UI_CMD—Information	3.3	O		Yes ☐	No ☐
XDT/15	XID_CMD—DSAP address	6.2, 6.6	XID/3:M	N/A ☐	Yes ☐	
XDT/16	XID_CMD—SSAP address	6.2, 6.6	XID/3:M	N/A ☐	Yes ☐	
XDT/17	XID_CMD—P-bit = 1	6.3	XID/3:O.2	N/A ☐	Yes ☐	No ☐
XDT/18	XID_CMD—P-bit = 0	6.3	XID/3:O.2	N/A ☐	Yes ☐	No ☐
XDT/19	XID_CMD—Information	5.4.1.1.2, 6.6	XID/3:M	N/A ☐	Yes ☐	
XDT/20	XID_RSP—DSAP address	6.2, 6.6	M		Yes ☐	
XDT/21	XID_RSP—SSAP address	6.2, 6.6	M		Yes ☐	
XDT/22	XID_RSP—F-bit = P-bit	6.3	M		Yes ☐	
XDT/23	XID_RSP—Information	5.4.1.2.1, 6.6	M		Yes ☐	
TST/24	TEST_CMD—DSAP address	6.2	TES/7:M	N/A ☐	Yes ☐	
TST/25	TEST_CMD—SSAP address	6.2	TES/7:M	N/A ☐	Yes ☐	
TST/26	TEST_CMD—P-bit = 1	6.3	TES/7:O.3	N/A ☐	Yes ☐	No ☐
TST/27	TEST_CMD—P-bit = 0	6.3	TES/7:O.3	N/A ☐	Yes ☐	No ☐
*TST/28	TEST_CMD—Information	5.4.1.1.3, 6.7	TES/7:O	N/A ☐	Yes ☐	No ☐
TST/29	TEST_RSP—DSAP address	6.2	M		Yes ☐	
TST/30	TEST_RSP—SSAP address	6.2	M		Yes ☐	
TST/31	TEST_RSP—F-bit = P-bit	6.3	M		Yes ☐	
TST/32	TEST_RSP—Information	5.4.1.2.2, 6.7	M		Yes ☐	

A.6.3 LLC Type 1—Supported parameters in PDUs on receipt

Item	Protocol feature/ Supported parameters on receipt	References	Status	Support	
UIR/33	UI_CMD—DSAP address	6.2	M	Yes ☐	
UIR/34	UI_CMD—SSAP address	6.2	M	Yes ☐	
UIR/35	UI_CMD—P-bit = 0	6.3	M	Yes ☐	
UIR/36	UI_CMD—Information	3.3	O	Yes ☐	No ☐
XDR/37	XID_CMD—DSAP address	6.2, 6.6	M	Yes ☐	
XDR/38	XID_CMD—SSAP address	6.2, 6.6	M	Yes ☐	
XDR/39	XID_CMD—P-bit = 1	6.3	M	Yes ☐	
XDR/40	XID_CMD—P-bit = 0	6.3	M	Yes ☐	
XDR/41	XID_CMD—Information	5.4.1.1.2, 6.6	M	Yes ☐	
XDR/42	XID_RSP—DSAP address	6.2, 6.6	M	Yes ☐	
XDR/43	XID_RSP—SSAP address	6.2, 6.6	M	Yes ☐	
XDR/44	XID_RSP—F-bit = P-bit	6.3	M	Yes ☐	
XDR/45	XID_RSP—Information	5.4.1.2.1, 6.6	M	Yes ☐	
TSR/46	TEST_CMD—DSAP address	6.2	M	Yes ☐	
TSR/47	TEST_CMD—SSAP address	6.2	M	Yes ☐	
TSR/48	TEST_CMD—P-bit = 1	6.3	M	Yes ☐	
TSR/49	TEST_CMD—P-bit = 0	6.3	M	Yes ☐	
TSR/50	TEST_CMD—Information	5.4.1.1.3, 6.7	M	Yes ☐	
TSR/51	TEST_RSP—DSAP address	6.2	M	Yes ☐	
TSR/52	TEST_RSP—SSAP address	6.2	M	Yes ☐	
TSR/53	TEST_RSP—F-bit = P-bit	6.3	M	Yes ☐	
TSR/54	TEST_RSP—Information	5.4.1.2.2, 6.7	TST/28:M	N/A ☐	Yes ☐

A.6.4 LLC Type 1—Miscellaneous

Item	Protocol feature/ Miscellaneous	References	Status	Support		
MIS/55	Do all transmitted PDUs contain an integral number of octets?	3.3	M	Yes ☐		
	If the following PDUs are received from the MAC sublayer, are they treated as invalid and ignored:					
MIS/56	—contains a non-integral number of octets	3.3.4	M	Yes ☐		
MIS/57	—has a length less than 3 octets	3.3.4	M	Yes ☐		
	Which of the following addresses are supported in the DSAP address field of UI PDUs?					
MIS/58	—individual address	5.4.1.1.1	O.4	Yes ☐	No ☐	
MIS/59	—group address	5.4.1.1.1	O.4	Yes ☐	No ☐	
MIS/60	—global address	5.4.1.1.1	O.4	Yes ☐	No ☐	
MIS/61	—null address	5.4.1.1.1	O.4	Yes ☐	No ☐	
MIS/62	Is the address in the SSAP address field of a UI PDU the originator's individual address?	5.4.1.1.1	M	Yes ☐		
MIS/63	Are all UI PDUs transmitted as UI_CMD PDUs?	6.5.1	M	Yes ☐		
MIS/64	Are all UI_CMD PDUs transmitted with the P-bit = 0?	6.5.1	M	Yes ☐		
MIS/65	If a UI_RSP PDU is received, is the frame discarded?	6.5.2	M	Yes ☐		
	Which of the following addresses are supported in the DSAP address field of XID_CMD PDUs?					
MIS/66	—individual address	5.4.1.1.2	XID/3:O.5	N/A ☐	Yes ☐	No ☐
MIS/67	—group address	5.4.1.1.2	XID/3:O.5	N/A ☐	Yes ☐	No ☐
MIS/68	—global address	5.4.1.1.2	XID/3:O.5	N/A ☐	Yes ☐	No ☐
MIS/69	—null address	5.4.1.1.2	XID/3:O.5	N/A ☐	Yes ☐	No ☐
	Which of the following addresses are supported in the SSAP address field of XID_CMD PDUs?					
MIS/70	—individual address	5.4.1.1.2	XID/3:O.6	N/A ☐	Yes ☐	No ☐
MIS/71	—null address	5.4.1.1.2	XID/3:O.6	N/A ☐	Yes ☐	No ☐
	Which of the following addresses are supported in the DSAP address field of XID_RSP PDUs?					
MIS/72	—individual address	5.4.1.2.1	M	Yes ☐		
MIS/73	—null address	5.4.1.2.1	M	Yes ☐		
	Which of the following addresses are supported in the SSAP address field of XID_RSP PDUs?					
MIS/74	—individual address	5.4.1.2.1	M	Yes ☐		
MIS/75	—null address	5.4.1.2.1	M	Yes ☐		

Item	Protocol feature/ Miscellaneous	References	Status	Support
(Con'd.)				
	Which of the following addresses are supported in the DSAP address field of TEST_CMD PDUs?			
MIS/76	— individual address	5.4.1.1.3	TES/7:O.7	N/A ☐ Yes ☐ No ☐
MIS/77	— group address	5.4.1.1.3	TES/7:O.7	N/A ☐ Yes ☐ No ☐
MIS/78	— global address	5.4.1.1.3	TES/7:O.7	N/A ☐ Yes ☐ No ☐
MIS/79	— null address	5.4.1.1.3	TES/7:O.7	N/A ☐ Yes ☐ No ☐
	Which of the following addresses are supported in the SSAP address field of TEST_CMD PDUs?			
MIS/80	— individual address	5.4.1.1.3	TES/7:O.8	N/A ☐ Yes ☐ No ☐
MIS/81	— null address	5.4.1.1.3	TES/7:O.8	N/A ☐ Yes ☐ No ☐
	Which of the following addresses are supported in the DSAP address field of TEST_RSP PDUs?			
MIS/82	— individual address	5.4.1.2.2	M	Yes ☐
MIS/83	— null address	5.4.1.2.2	M	Yes ☐
	Which of the following addresses are supported in the SSAP address field of TEST_RSP PDUs?			
MIS/84	— individual address	5.4.1.2.2	M	Yes ☐
MIS/85	— null address	5.4.1.2.2	M	Yes ☐
*MIS/86	Is Duplicate Address Checking supported?	6.9.2	O	Yes ☐ No ☐
MIS/87	—Is the ACK_TIMER function supported?	6.9.2	MIS/86:M	Yes ☐
MIS/88	—ACK_TIMER range (seconds)			Minimum Value = Maximum Value =
MIS/89	—Is the RETRY_COUNTER function supported?	6.9.2	MIS/86:M	Yes ☐
MIS/90	—RETRY_COUNTER range			Minimum Value = Maximum Value =
MIS/91	—Is the XID_R_COUNTER function supported?	6.9.2	MIS/86:M	Yes ☐

A.7 LLC Type 2 operation—Connection-mode

CLS2a *or* CLS4a::

All tables in clause A.7 are to be completed if above predicate evaluates to true.

A.7.1 LLC Type 2—Supported PDU types

Item	Protocol feature/ Supported PDU types	References	Status	Support		
*IP/92a	I PDU supported on transmission	7.4.2	O		Yes ☐	No ☐
*IC/92b	I_CMD supported on transmission	Table 4	IP/92a:O	N/A ☐	Yes ☐	No ☐
IC/93	I_CMD supported on receipt	Table 4	M		Yes ☐	
IR/94	I_RSP supported on transmission	Table 4	IP/92a:M	N/A ☐	Yes ☐	
IR/95	I_RSP supported on receipt	Table 4	M		Yes ☐	
RRC/96	RR_CMD supported on transmission	Table 4	M		Yes ☐	
RRC/97	RR_CMD supported on receipt	Table 4	M		Yes ☐	
RRR/98	RR_RSP supported on transmission	Table 4	M		Yes ☐	
RRR/99	RR_RSP supported on receipt	Table 4	M		Yes ☐	
RNC/100	RNR_CMD supported on transmission	Table 4	M		Yes ☐	
RNC/101	RNR_CMD supported on receipt	Table 4	M		Yes ☐	
RNR/102	RNR_RSP supported on transmission	Table 4	M		Yes ☐	
RNR/103	RNR_RSP supported on receipt	Table 4	M		Yes ☐	
*RJC/104	REJ_CMD supported on transmission	Table 4	O		Yes ☐	No ☐
RJC/105	REJ_CMD supported on receipt	Table 4	M		Yes ☐	
RJR/106	REJ_RSP supported on transmission	Table 4	M		Yes ☐	
RJR/107	REJ_RSP supported on receipt	Table 4	M		Yes ☐	
SAC/108	SABME_CMD supported on transmission	Table 4	M		Yes ☐	
SAC/109	SABME_CMD supported on receipt	Table 4	M		Yes ☐	
DIC/110	DISC_CMD support on transmission	Table 4	M		Yes ☐	
DIC/111	DISC_CMD supported on receipt	Table 4	M		Yes ☐	
UAR/112	UA_RSP supported on transmission	Table 4	M		Yes ☐	
UAR/113	UA_RSP supported on receipt	Table 4	M		Yes ☐	
DMR/114	DM_RSP supported on transmission	Table 4	M		Yes ☐	
DMR/115	DM_RSP supported on receipt	Table 4	M		Yes ☐	
FRR/116	FRMR_RSP supported on transmission	Table 4	M		Yes ☐	
FRR/117	FRMR_RSP supported on receipt	Table 4	M		Yes ☐	

A.7.2 LLC Type 2—Supported parameters in PDUs on transmission

Item	Protocol feature/ Supported parameters on transmission	References	Status	Support
	Do the following PDUs contain a DSAP address, an SSAP address and a Control field as specified in the given referenced clauses?			
PPT/118a	—I_CMD	3.2, 3.3, 5.4	IC/92b:M	N/A ☐ Yes ☐
PPT/118b	—I_RSP	3.2, 3.3, 5.4	IP/92a:M	N/A ☐ Yes ☐
PPT/119	—REJ_CMD	3.2, 3.3, 5.4	RJC/104:M	N/A ☐ Yes ☐
PPT/120	—RR_CMD, RR_RSP, RNR_ CMD, RNR_RSP, REJ_RSP, SABME_CMD, DISC_CMD, UA_RSP, DM_RSP, FRMR_RSP	3.2, 3.3, 5.4	M	Yes ☐
	Do the following PDUs contain an information field?			
PPT/121	—I_CMD, I_RSP, FRMR_RSP	3.2, 3.3, 5.4	M	Yes ☐
	Do the following PDUs contain an information field?			
PPT/122	—REJ_CMD, RR_CMD, RR_RSP, RNR_CMD, RNR_RSP, REJ_RSP, SABME_CMD, DISC_CMD, UA_RSP, DM_RSP	3.2, 5.4	X	No ☐

A.7.3 LLC Type 2—Supported parameters in PDUs on receipt

Item	Protocol feature/ Supported parameters on receipt	References	Status	Support
	Is the receipt of a DSAP, an SSAP and a Control field supported for the following PDUs?			
PPR/123	—I_CMD, I_RSP, RR_CMD, RR_RSP, RNR_CMD, RNR_RSP, REJ_CMD, REJ_RSP, SABME_CMD, DISC_CMD, UA_RSP, DM_RSP, FRMR_RSP	3.2, 3.3, 5.4	M	Yes ☐
	Is the receipt of an Information field supported for the following PDUs?			
PPR/124	—I_CMD, I_RSP, FRMR_RSP	3.2, 3.3, 5.4	M	Yes ☐
	NOTE—Response to receipt of a PDU with an Information field that is not permitted to have an Information field is covered under Frame Reject procedures.			

A.7.4 LLC Type 2—Supported procedures

Item	Protocol feature/ Supported procedures	References	Status	Support
	Support of Connection Establishment			
PRS/125	—as initiator	7.4.1, 7.4.5	O	Yes ☐ No ☐
PRS/126	—as responder	7.4.1, 7.4.5	M	Yes ☐
	Support of Connection Release			
PRS/127	—as initiator (originating release)	7.4.3, 7.4.4, 7.4.5	O	Yes ☐ No ☐
PRS/128	—as initiator (rejecting connection establish-	7.4.3, 7.4.4, 7.4.5	M	Yes ☐
PRS/129	ment) —as responder	7.4.3, 7.4.4, 7.4.5	M	Yes ☐
	Support of Data Transfer			
PRS/130	—as originator	7.4.2, 7.5	O	Yes ☐ No ☐
PRS/131	—as responder	7.4.2, 7.5	M	Yes ☐
PRS/132	Is the Remote Busy procedure supported on receipt of an RNR PDU?	7.5.7	M	Yes ☐
PRS/133	Is the Retransmission procedure supported on receipt of an REJ PDU?	7.5.6	M	Yes ☐
PRS/134	Is the Reject procedure supported on receipt of an I PDU with an unexpected N(S)?	7.4.2, 7.5	M	Yes ☐
PRS/135	Is the Local Busy procedure supported?	7.5.8	M	Yes ☐
PRS/136	Is the Frame Reject procedure supported?	5.4.2.3.5	M	Yes ☐
	Is the Reset procedure supported?			
PRS/137	—as initiator	7.5, 7.6, 7.7	M	Yes ☐
PRS/138	—as responder	7.5, 7.6, 7.7	M	Yes ☐
*PRS/139	Is the LLC Flow Control procedure supported?	Annex B	O	Yes ☐ No ☐

A.7.5 LLC Type 2—Miscellaneous

Item	Protocol feature/ Miscellaneous	References	Status	Support
MIS/140	Do all transmitted PDUs contain an integral number of octets?	3.3	M	Yes ☐
	If the following PDUs are received from the MAC sublayer, are they treated as invalid and ignored?			
MIS/141	—contains a non-integral number of octets	3.3.4	M	Yes ☐
MIS/142	—has a length less than 3 octets (in the case of a one-octet control field)	3.3.4	M	Yes ☐
MIS/143	—has a length less than 4 octets (in the case of a two-octet control field)	3.3.4	M	Yes ☐
MIS/143a	Are duplicate I PDUs recognized and discarded?	7.5.10	O	Yes ☐ No ☐

A.7.6 LLC Type 2—Protocol parameters

Item	Protocol feature/ Protocol parameters	References	Status	Support/Value
PPA/144 PPA/145	Is the ACK_TIMER function implemented? —ACK_TIMER range	7.8.1.1	M	Yes ☐ Minimum Value = Maximum Value =
PPA/146 PPA/147	Is the P_TIMER function implemented? —P_TIMER range	7.8.1.2	M	Yes ☐ Minimum Value = Maximum Value =
PPA/148 PPA/149	Is the REJ_TIMER function implemented? —REJ_TIMER range	7.8.1.3	M	Yes ☐ Minimum Value = Maximum Value =
PPA/150 PPA/151	Is the BUSY_TIMER function implemented? —BUSY_TIMER range	7.8.1.4	M	Yes ☐ Minimum Value = Maximum Value =
PPA/152 PPA/153	Is the N2 (Maximum Number of Transmissions) function implemented? —Number of Transmissions	7.8.2	M	Yes ☐ Minimum Value = Maximum Value =
PPA/154 PPA/155	Is the k (Transmit Window Size) function implemented? —Maximum value of k	7.8.4	M	Yes ☐ Value =
PPA/156 PPA/157a	Is the LLC flow control function implemented? —kstep range	Annex B	PRS/139:M	N/A ☐ Yes ☐ Minimum Value = Maximum Value =
PPA/157b PPA/157c	Is the RW (Receive Window Size) function implemented? —Maximum value of RW	7.8.6	M	Yes ☐ Value =

A.8 LLC Type 3 operation—Acknowledged connectionless-mode

CLS3a *or* CLS4a::

All tables in clause A.8 are to be completed if above predicate evaluates to true.

A.8.1 LLC Type 3—Supported PDU types

Item	Protocol feature/ Supported PDU types	References	Status	Support		
*AnC/158a	Are ACn commands transmitted?	8.5.1	O		Yes ☐	No ☐
A0C/158b	AC0_CMD supported on transmission	8.5.1	AnC/158a:M	N/A ☐	Yes ☐	
A0C/159	AC0_CMD supported on receipt	8.5.2	M		Yes ☐	
A0R/160	AC0_RSP supported on transmission	8.5.3	M		Yes ☐	
A0R/161	AC0_RSP supported on receipt	8.5.4	AnC/158a:M	N/A ☐	Yes ☐	
A1C/162	AC1_CMD supported on transmission	8.5.1	AnC/158a:M	N/A ☐	Yes ☐	
A1C/163	AC1_CMD supported on receipt	8.5.2	M		Yes ☐	
A1R/164	AC1_RSP supported on transmission	8.5.3	M		Yes ☐	
A1R/165	AC1_RSP supported on receipt	8.5.4	AnC/158a:M	N/A ☐	Yes ☐	

A.8.2 LLC Type 3—Supported parameters in PDUs on transmission

Item	Protocol feature/ Supported parameters on transmission	Refer-ences	Status	Support		
A0T/166	AC0_CMD—DSAP address	8.2	AnC/158a:M	N/A ☐	Yes ☐	
A0T/167	AC0_CMD—SSAP address	8.2	AnC/158a:M	N/A ☐	Yes ☐	
	AC0_CMD—					
A0T/168	—P-bit = 1 and non-null Information field	8.3	AnC/158a:O.9	N/A ☐	Yes ☐	No ☐
A0T/169	—P-bit = 1 and null Information field	8.3	AnC/158a:O.9	N/A ☐	Yes ☐	No ☐
	AC0_CMD—					
A0T/170	—P-bit = 0 and non-null Information field	8.3	AnC/158a:O.9	N/A ☐	Yes ☐	No ☐
A0T/171	—P-bit = 0 and null Information field	8.3	AnC/158a:O.9	N/A ☐	Yes ☐	No ☐
A1T/172	AC1_CMD—DSAP address	8.2	AnC/158a:M	N/A ☐	Yes ☐	
A1T/173	AC1_CMD—SSAP address	8.2	AnC/158a:M	N/A ☐	Yes ☐	
	AC1_CMD—					
A1T/174	—P-bit = 1 and non-null Information field	8.3	AnC/158a:O.10	N/A ☐	Yes ☐	No ☐
A1T/175	—P-bit = 1 and null Information field	8.3	AnC/158a:O.10	N/A ☐	Yes ☐	No ☐
	AC1_CMD—					
A1T/176	—P-bit = 0 and non-null Information field	8.3	AnC/158a:O.10	N/A ☐	Yes ☐	No ☐
A1T/177	—P-bit = 0 and null Information field	8.3	AnC/158a:O.10	N/A ☐	Yes ☐	No ☐

Item	Protocol feature/ Supported parameters on transmission	Refer-ences	Status	Support	
(Con'd.)					
A0T/178	AC0_RSP—DSAP address	8.2	M	Yes ☐	
A0T/179	AC0_RSP—SSAP address	8.2	M	Yes ☐	
	AC0_RSP—F-bit = P-bit (= 1)				
A0T/180	—with Status Subfield and non-null LSDU Subfield	8.3	O	Yes ☐	No ☐
A0T/181	—with Status Subfield and null LSDU Subfield	8.3	M	Yes ☐	
	AC0_RSP—F-bit = P-bit (= 0)				
A0T/182	—with Status Subfield and non-null LSDU Subfield	8.3	X		No ☐
A0T/183	—with Status Subfield and null LSDU Subfield	8.3	M	Yes ☐	
A1T/184	AC1_RSP—DSAP address	8.2	M	Yes ☐	
A1T/185	AC1_RSP—SSAP address	8.2	M	Yes ☐	
	AC1_RSP—F-bit = P-bit (= 1)				
A1T/186	—with Status Subfield and non-null LSDU Subfield	8.3	O	Yes ☐	No ☐
A1T/187	—with Status Subfield and null LSDU Subfield	8.3	M	Yes ☐	
A1T/188	AC1_RSP—F-bit = P-bit (= 0)				
	—with Status Subfield and non-null LSDU Subfield	8.3	X		No ☐
A1T/189	—with Status Subfield and null LSDU Subfield	8.3	M	Yes ☐	

A.8.3 LLC Type 3—Supported parameters in PDUs on receipt

Item	Protocol feature/ Supported parameters on receipt	References	Status	Support
A0P/190	AC0_CMD—DSAP address	8.1, 8.2	M	Yes ☐
A0P/191	AC0_CMD—SSAP address	8.1, 8.2	M	Yes ☐
	AC0_CMD—			
A0P/192	—P-bit = 1 and non-null Information field	8.1, 8.3	M	Yes ☐
A0P/193	—P-bit = 1 and null Information field	8.1, 8.3	M	Yes ☐
	AC0_CMD—			
A0P/194	—P-bit = 0 and non-null Information field	8.1, 8.3	M	Yes ☐
A0P/195	—P-bit = 0 and null Information field	8.1, 8.3	M	Yes ☐
A1P/196	AC1_CMD—DSAP address	8.1, 8.2	M	Yes ☐
A1P/197	AC1_CMD—SSAP address	8.1, 8.2	M	Yes ☐
	AC1_CMD—			
A1P/198	—P-bit = 1 and non-null Information field	8.1, 8.3	M	Yes ☐
A1P/199	—P-bit = 1 and null Information field	8.1, 8.3	M	Yes ☐
	AC1_CMD—			
A1P/200	—P-bit = 0 and non-null Information field	8.1, 8.3	M	Yes ☐
A1P/201	—P-bit = 0 and null Information field	8.1, 8.3	M	Yes ☐
A0P/202	AC0_RSP—DSAP address	8.1, 8.2	M	Yes ☐
A0P/203	AC0_RSP—SSAP address	8.1, 8.2	M	Yes ☐
	AC0_RSP—F-bit = P-bit (= 1)			
A0P/204	—with Status Subfield and non-null LSDU Subfield	8.1, 8.3	M	Yes ☐
A0P/205	—with Status Subfield and null LSDU Subfield	8.1, 8.3	M	Yes ☐
	AC0_RSP—F-bit = P-bit (= 0)			
A0P/206	—with Status Subfield and non-null LSDU Subfield	8.1, 8.3, 8.7.2	M	Yes ☐
A0P/207	—with Status Subfield and null LSDU Subfield	8.1, 8.3	M	Yes ☐
A1P/208	AC1_RSP—DSAP address	8.1, 8.2	M	Yes ☐
A1P/209	AC1_RSP—SSAP address	8.1, 8.2	M	Yes ☐
	AC1_RSP—F-bit = P-bit (= 1)			
A1P/210	—with Status Subfield and non-null LSDU Subfield	8.1, 8.3	M	Yes ☐
A1P/211	—with Status Subfield and null LSDU Subfield	8.1, 8.3	M	Yes ☐
	AC1_RSP—F-bit = P-bit (= 0)			
A1P/212	—with Status Subfield and non-null LSDU Subfield	8.1, 8.3, 8.7.2	M	Yes ☐
A1P/213	—with Status Subfield and null LSDU Subfield	8.1, 8.3	M	Yes ☐

A.8.4 LLC Type 3—Supported procedures

Item	Protocol feature/ Supported procedures	References	Status	Support
PRS/214	Is the procedure for Sequence Number Resynchronization supported?	8.4, 8.4.1	AnC/158a:O	N/A ☐ Yes ☐ No ☐
PRS/215a	Is the procedure for Destruction of a Transmit Sequence State Variable V(SI) supported?	8.4, 8.4.2	AnC/158a:O	N/A ☐ Yes ☐ No ☐
PRS/215b	Is the procedure for Destruction of a Receive Sequence State Variable V(RI) and Reception Status State Variable V(RB) supported?	8.4, 8.4.3	O	Yes ☐ No ☐
	Are the procedures for Information Transfer supported?			
PRS/216	—for the general MAC capabilities case	8.5, 8.5.1, 8.5.2.1, 8.5.3, 8.5.4.1	O.11	Yes ☐ No ☐
PRS/217	—for the certain special MAC capabilities case	8.5, 8.5.1, 8.5.2.2, 8.5.3, 8.5.4.2	O.11	Yes ☐ No ☐

A.8.5 LLC Type 3—Miscellaneous

Item	Protocol feature/ Miscellaneous	References	Status	Support
MIS/218	Do all transmitted PDUs contain an integral number of octets?	3.3	M	Yes ☐
	If the following PDUs are received from the MAC sublayer, are they treated as invalid and ignored?			
MIS/219	—contains a non-integral number of octets	3.3.4	M	Yes ☐
MIS/220	—has a length less than 3 octets	3.3.4	M	Yes ☐
	NOTE—The reception of an AC0_RSP or AC1_RSP with bit 8 of the Control Field equal to V(SI) is covered in items PRS/216 and PRS/217.			
	Which of the following addresses are supported in the DSAP address field of AC0_CMD and AC1_CMD PDUs?			
MIS/221	—individual address	5.4.3.1	O.12	Yes ☐ No ☐
MIS/222	—group address	5.4.3.1	O.12	Yes ☐ No ☐
MIS/223	—global address	5.4.3.1	O.12	Yes ☐ No ☐
MIS/224	Is the address in the SSAP address field of AC0_CMD and AC1_CMD PDUs the originator's individual address?	5.4.3.1	M	Yes ☐
	Is the address in the DSAP address field of AC0_RSP and AC1_RSP PDUs?			
MIS/225	—the individual address of the corresponding AC0_CMD or AC1_CMD originator	5.4.3.2	M	Yes ☐
	Is the address in the SSAP address field of AC0_RSP and AC1_RSP PDUs?			
MIS/226	—the individual address of the AC0_RSP or AC1_RSP transmitter	5.4.3.2	M	Yes ☐

A.8.6 LLC Type 3—Protocol parameters

Item	Protocol feature/ Protocol parameters	References	Status	Support/Value
PPA/227 * PPA/228	Is the N4 (Maximum Number of Transmissions) function implemented? —Maximum Number of Transmissions	8.6.1	M	Yes ☐ Value =
PPA/229 PPA/230	Is the T1 (ACK_TIMER) function implemented? —ACK_TIMER range	8.6.4	If PPA/228 > 1:M	N/A ☐ Yes ☐ Minimum Value = Maximum Value =
PPA/231 PPA/232	Is the T2 (RCV_LIFE_TIMER) function implemented? —RCV_LIFE_TIMER range	8.6.5	M	Yes ☐ Minimum Value = Maximum Value =
PPA/233 PPA/234	Is the T3 (TX_LIFE_TIMER) function implemented? —TX_LIFE_TIMER range	8.6.6	M	Yes ☐ Minimum Value = Maximum Value =

A.9 Route determination entity

RDE::

All tables in clause A.9 are to be completed if above predicate evaluates to true.

A.9.1 General characteristics

Item	Protocol feature/ Support of RDE PDU types	References	Status	Support
RQC/1	ROUTE_QUERY_COMMAND supported on transmission	9.3.7.2, 9.4.1	M	Yes ☐
RQC/2	ROUTE_QUERY_COMMAND supported on receipt	9.3.7.2, 9.4.1	M	Yes ☐
RQR/3	ROUTE_QUERY_RESPONSE supported on transmission	9.3.7.2, 9.4.2	M	Yes ☐
RQR/4	ROUTE_QUERY_RESPONSE supported on receipt	9.3.7.2, 9.4.2	M	Yes ☐
RS/5	ROUTE_SELECTED supported on transmission	9.3.7.2, 9.4.3	M	Yes ☐
RS/6	ROUTE_SELECTED supported on receipt	9.3.7.2, 9.4.3	M	Yes ☐

A.9.2 RDE PDU format

Item	Protocol feature/ RDE PDU format	References	Status	Support	
	Do the information fields of all transmitted RDE PDUs contain the following fields as specified in the referenced clauses?				
F/7	—RDE_Ver	9.4		Yes ☐	
F/8	—PType	9.4	M	Yes ☐	
F/9	—Target_MAC	9.4	M	Yes ☐	
F/10	—ORIG_MAC	9.4	M	Yes ☐	
F/11	—Target_SAP	9.4	M	Yes ☐	
F/12	—ORIG_SAP	9.4	M	Yes ☐	
F/13	—Options	9.4	O	Yes ☐	No ☐

A.9.3 Support of Route Control procedures

Item	Protocol feature/Support of Route Control procedures	References	Status	Support	
	Route Control Send EVENT/ACTION Specifications	9.6.3.5			
RCS/14a	—ROUTE_QUERY_COMMAND		M	Yes ☐	
RCS/14b	—ROUTE_QUERY_RESPONSE		M	Yes ☐	
RCS/14c	—ROUTE_SELECTED		M	Yes ☐	
RCS/15	Are all RDE PDUs sent according to the requirements specified in the referenced clauses?	Table 8, 9.6.3.5.1, 9.6.3.5.2	M	Yes ☐	
	Route Control Receive EVENT/ ACTION Specifications	9.6.3.4			
RCR/16a	—ROUTE_QUERY_COMMAND		M	Yes ☐	
RCR/16b	—ROUTE_QUERY_RESPONSE		M	Yes ☐	
RCR/16c	—ROUTE_SELECTED		M	Yes ☐	
RCR/17	Are all RDE PDUs received according to the requirements specified in the referenced clauses?	Table 7, 9.6.3.4.1, 9.6.3.4.2	M	Yes ☐	
	If the following PDUs (i.e., PDUs addressed to the RDE LLC address) are received, are they discarded?	9.4			
RDEP/18a	—it is not a UI PDU		M	Yes ☐	
RDEP/18b	—the SSAP is not the RDE LLC Address		O	Yes ☐	
RDEP/18c	—the ORIG_MAC is not an individual address		M	Yes ☐	
RDEP/18d	—the PType is not recognized		M	Yes ☐	
	Route Control Reset EVENT/ ACTION Specifications	9.6.3.6			
*RRST/19	Are the procedures for Route Reset supported?	9.6.3.6	O	Yes ☐	No ☐
RRST/20	Do RDE reset actions occur according to the requirements specified in the referenced clauses?	Table 9, 9.6.3.6.1, 9.6.3.6.2	RRST/19:M	N/A ☐ Yes ☐	

A.9.4 Support of Route Determination procedures

Item	Protocol feature/ Support of Route Determination procedures	References	Status	Support	
	Route Selection State Machine	9.7.3			
* RSSM/21	Is the REPLACE policy supported?	9.7.3.3, 9.7.1	O	Yes ☐	No ☐
	Are each of the following states (phases) recognized as specified in the referenced clauses?				
		9.7.3.1, 9.7.3.2	M	Yes ☐	
RSSM/22a	—Initial Phase (Reset State)	9.7.3.1, 9.7.3.2	M	Yes ☐	
RSSM/22b	—Query Phase	9.7.3.1, 9.7.3.2	M	N/A ☐ Yes ☐	
RSSM/22c	—Waiting Phase	9.7.3.1, 9.7.3.2	RSSM/21:M	Yes ☐	
RSSM/22d	—New Route Phase	9.7.3.1, 9.7.3.2	M	Yes ☐	
RSSM/22e	—Passive Phase (Route State)				
RSSM/23	Are all state transitions governed by the requirements specified in the referenced clauses?	Table 10, 9.7.3.3	M	Yes ☐	
	Route Administration State Machine	9.7.4			
*RASM/24	Is the HOLD policy supported?	9.7.4.3, 9.7.1	O	Yes ☐	No ☐
	Are each of the following states (phases) recognized as specified in the referenced clauses?				
RASM/25a	—System Empty Phase	9.7.4.1.3	RASM/24:M	N/A ☐	Yes ☐
RASM/25b	—Route Inuse Phase	9.7.4.1.3	RASM/24:M	N/A ☐	Yes ☐
RASM/25c	—New Route Pending Phase	9.7.4.1.3	RASM/24:M	N/A ☐	Yes ☐
RASM/25d	—Holding Phase	9.7.4.1.3	RASM/24:M	N/A ☐	Yes ☐
RASM/26	Are all state transitions governed by the requirements specified in the referenced clauses?	Table 11, 9.7.4.2, 9.7.4.3	RASM/24:M	N/A ☐	Yes ☐

A.9.5 Parameter and parameter values

Item	Protocol feature/ Parameter and parameter values	References	Status	Support/Value
	Route Control Operating Parameters	9.6.1		
STR/27	Is the STR_INDICATOR SUPPORTED? —default value (NSR or STE)	9.6.1.1	M	Yes ☐ Default Value =
AGEE/28	Is the AGING_ENABLED indicator supported?	9.6.1.2	M	Yes ☐
	Route Determination Parameters	9.6.2		
	Are the following RDE policy flags supported on a per SAP basis?			
RDEE/29a RDEE/29b	RDE_ENABLED —default value (allowed values: True or False)	9.6.2.1	M	Yes ☐ Default Value =
RPL/30a RPL/30b	REPLACE —default value (allowed values: True or False)	9.6.2.2	M	Yes ☐ Default Value =
HLD/31a HLD/31b	HOLD —default value (allowed values: True or False)	9.6.2.3	M	Yes ☐ Default Value =
	Are the following route selection parameters supported?			
MXRD/31a MXRD/31b MXRD/31c	MAX_RD —default value —min–max range (2 to 30)	9.6.2.4, 9.7.1	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value =
MNPD/32a MNPD/32b MNPD/32c	MIN_PDU_SIZE —default value (octets) —min–max range (octets) (516)	9.6.2.5, 9.7.1	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value =
MXRT/33a MXRT/33b MXRT/33c MXRT/33d	MAX_RESP_TIME —default value (seconds) —min–max range (seconds) —granularity (seconds)	9.6.2.6, 9.7.1	M	Yes ☐ Default Value = Minimum Value = Maximum Value = Value =
ROT/34a ROT/34b	Is a RESET_ON_TEST parameter supported? —default value (On or Off)	9.6.2.7	O	Yes ☐ No ☐ Default Value =
TRS/35a TRS/35b TRS/35c TRS/35d	Is a Route Selection Timer (TRS) supported? —default value (seconds) — min–max range (seconds) (≥ max. round-trip latency) — granularity (seconds)	9.7.3.1	M	Yes ☐ Default Value = Minimum Value = Maximum Value = Value =
AGEF/36a AGEF/36b AGEF/36c AGEF/36d	Is the Ageing Function supported? —default value (seconds) —min–max range (seconds) —granularity (seconds)	9.7.3.1	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value = Value =

Item	Protocol feature/ Parameter and parameter values	References	Status	Support/Value
(Con'd.) TRR/37a TRR/37b TRR/37c TRR/37d	Is Route Response Timer (TRR) supported? —default value (seconds) —min–max range (seconds) (TRS/2 to TRS) —granularity (seconds)	9.7.4.1.4	M	Yes ☐ Default Value = Minimum Value = Maximum Value = Value =
IRPE/38a IRPE/38b IRPE/38c	Is an invalid RDE frames counter supported? —default value —min–max range	9.6.3.4.2	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value =
ISRF/39a ISRF/39b ISRF/39c	Is a Specifically Routed Frames counter supported? —default value —min–max range	9.6.3.5.2	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value =
IST/40a IST/40b IST/40c	Is a Spanning Tree routed frames counter supported? —default value —min–max range	9.6.3.5.2	O	Yes ☐ No ☐ Default Value = Minimum Value = Maximum Value =

Annex B

(informative)

Relationship between LLC Type 3 and PROWAY (IEC 60955 : 1989)

B.1 Conditions imposed by PROWAY

In order to guarantee acceptable access times for industrial applications, the PROWAY International Standard (IEC 60955 : 1989) imposes specific conditions on the data link layer. Following are those conditions that impact the operation of LLC Type 3:

1) The Medium Access Control sublayer is ISO/IEC 8802-4 with the immediate acknowledgment supported.
2) Only the DL-REPLY service is used to request data. Thus, the ACn command PDU will always contain a null information field when the P bit is set to “1”.
3) The MAC service class of Request-with-Response is used for all Type 3 command PDUs.
4) LLC makes only one attempt to send each Type 3 command PDU. (Logical link parameter N4 is set to one.)
5) The Type 3 acknowledgment timer is eliminated. (Logical link parameter T1 is set to infinity.)
6) The Type 3 transmit variable lifetime timer is eliminated. (Logical link parameter T3 is set to infinity.)

B.2 Simplification of state information for PROWAY

The PROWAY International Standard imposes specific conditions on the operation of the MAC sublayer. These conditions allow simplification of the state information needed by LLC for the Type 3 services. These simplifications are detailed in 8.5.2.2 and 8.5.4.2.

B.3 Mapping of PROWAY services onto LLC services

The service primitives given in the PROWAY International Standard map to those of Type 1 and Type 3 LLC as shown in table B.1.

Table B.1—Mapping of PROWAY service primitives to LLC

PROWAY	LLC
L_DATA.request	DL-DATA request
L_DATA.confirm	(not present)
L_DATA_ACK.request	DL-DATA-ACK request
L_DATA_ACK.indicate	DL-DATA-ACK indication
L_DATA_ACK.confirm	DL-DATA-ACK-STATUS indication
L_REPLY.request	DL-REPLY request
L_REPLY.indicate	DL-REPLY indication
L_REPLY.confirm	DL-REPLY-STATUS indication
L_REPLY_UPDATE.request	DL-REPLY-UPDATE request
L_REPLY_UPDATE.confirm	DL-REPLY-UPDATE-STATUS indication
MA_DATA.request	MA-UNITDATA request
MA_DATA.indicate	MA-UNITDATA indication
MA_DATA.confirm	MA-UNITDATA-STATUS indication

Some parameters differ, as shown in table B.2.

Table B.2—Mapping of PROWAY service primitive parameters to LLC

PROWAY	LLC
SSAP	SSAP portion of source_address
DSAP	DSAP portion of destination_address
local_address	SA portion of source_address
remote_address	DA portion of destination_address
L_sdu	data
service_class	service_class
L_Status	status

B.4 Mapping of PROWAY L_pdu_type field onto LLC control field

For each of the PDUs supported by PROWAY, the PROWAY L_pdu_type field is equal in value to the LLC control field for the corresponding PDU. There are, however, the following differences in the notation used to describe these fields in the two International Standards.

1) LLC specifies the P/F bit separately, while PROWAY includes the P/F bit in the code-point. (In the case of the UI PDU, PROWAY assumes the P/F bit is zero.)

2) For Type 3 operation, PROWAY specifies the sequence bit separately, while LLC includes the sequence bit in the code-point.

Table B.3 shows the mapping of the PROWAY notation for the L_pdu_type field to the LLC notation for the control field.

Table B.3—Relationship between PROWAY L_pdu_type field and LLC control field

PROWAY and LLC field encoding	PROWAY		LLC	
	PDU Type	S bit	PDU Type	P/F bit
11000000	SDN		UI	0
11100110	SDA, SDAR	0	AC0	0
11100111	SDA, SDAR	1	AC1	0
11101110	RDR, RDRR	0	AC0	1
11101111	RDR, RDRR	1	AC1	1

B.5 Simplified Type 3 receiver component state tables

The conditions set by PROWAY (see B.1) permit the Type 3 LLC receiver component state tables to be simplified by eliminating state transitions and actions that cannot occur. These reduced state tables can be shown to be equivalent to those of the PROWAY remote state machine. The state, event, and action descriptions are a subset of those in the main body of this LLC International Standard, except for the REPLY indication primitive action, which has no LSDU parameter because the LSDU is always null.

B.6 Simplified Type 3 sender component state tables

The conditions set by PROWAY (see B.1) permit the Type 3 LLC sender component state tables to be simplified by eliminating state transitions and actions that cannot occur. These reduced state tables can be shown to be equivalent to those of PROWAY local state machine. The state, event, and action descriptions are a subset of those in the main body of this LLC International Standard.

Table B.4—Simplified Type 3 receiver component state transition table

Current state	Event	Action(s)	Next state
READY	REPLY_UPDATE_REQUEST	SAVE:=GIVEN_LSDU REPLY_UPDATE_STATUS_INDICATION	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=O, INFO<>NULL) and RECEIVE_STATUS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=0, C=OK, R=NR, LSDU=NULL) DATA_ACK_INDICATION V(RI):=1-SQC V(RB):=OK	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=0, INFO=NULL) and RECEIVE_STATUS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=0, C=OK, R=NR, LSDU=NULL) V(RI):=1-SQC V(RB):=OK	READY
	RECEIVE_ACn_CMD(SQC=V(RI), P=0) and RECEIVE_STATUS()<>OK	SEND-ACn_RSP(SQR=1-SQC, F=0, C=RECEIVE_STATUS(), R=NR, LSDU=NULL) V(RI):=1-SQC V(RB):=RECEIVE_STATUS()	READY
	RECEIVE_ACn_CMD(P=1) and ACCESS()=OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=OK, R=OK, LSDU=SAVE) REPLY_INDICATION V(RI):=1-SQC V(RB):=OK	READY
	RECEIVE_ACn_CMD(P=1) and ACCESS()<>OK	SEND_ACn_RSP(SQR=1-SQC, F=1, C=OK, R=ACCESS(), LSDU=NULL) V(RI):=1-SQC V(RB):=OK	READY
	RECEIVE_ACn_CMD(SQC<>V(RI), P=0)	SEND_ACn_RSP(SQR=1-SQC, F=0, C=V(RB), R=NR, LSDU=NULL)	READY

Table B.5—Simplified Type 3 sender component state transition table

Current state	Event	Action(s)	Next state
IDLE	MA_UNITDATA_STATUS or RECEIVE_ACn_RSP		IDLE
	DATA_ACK_REQUEST	SEND_ACn_CMD(SQC=V(SI), P=0)	WAIT_A
	REPLY_REQUEST	SEND_ACn_CMD(SQC=V(SI), P=1)	WAIT_R
WAIT_A	RECEIVE_ACn_RSP(LSDU=NULL)	DATA_ACK_STATUS_INDICATION(STATUS=STATUS_SUBFIELD) V(SI):=1-V(SI)	IDLE
	RECEIVE_ACn_RSP (LSDU<>NULL)	DATA_ACK_STATUS_INDICATION(STATUS=PE) V(SI):=1-V(SI) REPORT_STATUS(ILLEGAL_LSDU)	IDLE
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=GOOD		WAIT_A
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=BAD	DATA_ACK_STATUS_INDICATION(STATUS=TRANSMISSION_ STATUS)	IDLE
WAIT_R	RECEIVE_ACn_RSP(R=OK)	REPLY_STATUS_INDICATION(STATUS=STATUS_SUBFIELD, LSDU=GIVEN_LSDU) V(SI):=1-V(SI)	IDLE
	RECEIVE_ACn_RSP(R<>OK)	REPLY_STATUS_INDICATION(STATUS=STATUS_SUBFIELD, LSDU=NULL) V(SI):=1-V(SI)	IDLE
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=GOOD		WAIT_R
	MA_UNITDATA_STATUS and TRANSMISSION_STATUS=BAD	REPLY_STATUS_INDICATION(STATUS=TRANSMISSION_ STATUS, LSDU=NULL)	IDLE

Annex C

(informative)

LLC flow control techniques for bridged LANs

C.1 Overview

This annex describes a technique, called dynamic window flow control, to control the offering of frames to the network by an LLC entity when congestion is detected or suspected. It is most effective in a bridged LAN. The technique is one of recovery from congestion and does not prevent congestion in a bridged LAN. It is not a substitute for proper network sizing.

The method employs the transmit window already permitted by the standard to regulate the flow between two LLCs using the connection-mode service. Congestion in one direction of a logical link connection is treated independently of congestion in the other direction. The technique does not involve communication with the bridges, but rather relies on a simple algorithm implemented by the LLCs. MAC protocols are unaffected.

All actions described in this annex apply to the station transmitting in the direction of the congestion. The receiver does not participate, except through normal LLC procedures, and does not require knowledge of the transmitter's participation. The service interface between the data link layer and the network layer is also unchanged.

C.2 Definitions

C.2.1 k: The transmit window size in use at any given time.

C.2.2 kmax: The maximum transmit window size, which is the maximum value that the transmit window k may have. The value of kmax shall not exceed 127.

C.2.3 kstep: The number of sequentially transmitted, previously unacknowledged I-format PDUs that are successfully received and acknowledged in order to increment k.

C.3 Transmit windows

The dynamic window algorithm consists of modifying the transmitter's transmit window when congestion is first detected and then as the congestion decreases. If the transmitter always uses a transmit window size, k, of 1, this algorithm need not be invoked. If the transmitter uses larger transmit window sizes, then in the absence of congestion the transmitter uses a transmit window size k equal to a maximum possible value, kmax. Thus the transmitter can have as many as kmax frames outstanding, or unacknowledged, at any time. The value of kmax is set to less than or equal to the receive window of the XID sender, and k is initialized to kmax.

C.4 Detection of congestion

Congestion is indicated by the loss of I-format PDUs. (It is assumed that the loss of frames because of random bit errors is small.) Lost I PDUs are detected by their transmitter in one of two ways:

1) The transmitter receives a REJ PDU, which indicates that the receiver detected the loss of an I PDU, or
2) The following sequence of events occurs:
   a) The transmitter's acknowledgment timer expires,
   b) The transmitter sends a RR command PDU with the bit set to "1", and
   c) The transmitter receives an I-format or S-format response PDU in which the F bit is set to "1", but in which the value of N(R) is not equal to the value that the transmitter's send state variable V(S) had when the frame with the P bit set to "1" was sent.

C.5 Operation of the algorithm

When congestion is indicated by one of these two events, the dynamic window algorithm is invoked. Upon invocation, the transmitter sets its transmit window size k to 1. Thus the transmitter waits for an acknowledgment after every I PDU transmission.

Thereafter, when a certain number, kstep, of previously unacknowledged I PDUs are successfully transmitted and acknowledged, k is increased by 1. The value of kstep may be a constant or it may vary; one method of variance is to let kstep = k. The greater the value of kstep, the longer the effect of the flow control.

As I PDUs are successfully transmitted, k will gradually increase until it reaches its maximum value kmax. At this point, the algorithm ends. If another I PDU is lost, either before k reaches kmax or after, the algorithm returns to its starting point, namely the resetting of k to 1.

Annex D

(informative)

Subnetwork access protocol support

This annex presents the preferred procedures for LLC RDE support of subnetwork access protocol (SNAP) addressing.

D.1 Overview

This annex describes the characteristics and options that are recommended in LLC RDE implementations to ensure optimum support for SNAP addressing.

D.2 LLC RDE implementation

LLC RDE implementations supporting SNAP addressing will have the following characteristics:

1) The source_address and destination_address parameters, defined for all RDE primitives, will specify either the LSAP pair or the SNAP pair for which routing information is being requested/indicated. The LSAP address is designated by the combination of the MAC address, the LLC address, and the SNAP address.
2) The SNAP protocol identifier (PID) is specified in option field of the RDE PDU, and encoded as follows:

Tag	Length	Value
00 01	00 05	xx xx xx xx xx

where

XX XX XX XX XX represents a 5-octet SNAP PID. For more information concerning the PID and its representation, see [B1],[1] clause 5.3.

3) The RCC will respond to all valid RDE PDUs that specify a PTYPE of RQC by returning an RQR RDE PDU. The RQR PDU OPTIONS field will reflect the SNAP option only if the SNAP option was present in the invoking RQC.
4) When an RQR PDU without a SNAP option is received, the routing information contained within the PDU will apply to all SNAP data links associated with the source MAC address of the received RQR. When an RQR PDU containing a SNAP option is received, the routing information contained within the PDU will apply to the SNAP data link designated by the MAC, SAP, and SNAP addresses indicated in the RQR.

D.3 Bibliography

[B1] IEEE Std 802-1990, IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture (ANSI).[2]

[1]The numbers in brackets preceded by the letter B correspond to those of the bibliography in D.3.

Annex E

(normative)

Allocation of object identifier values

E.1 Introduction

This annex contains a summary of all object identifier values that have been allocated by this standard, both in this revision and in previous revisions.

Each table shows allocations related to a specific category of information object. The heading of the table identifies the category of information object, and shows the invariant part of the object identifier value allocated to the entries in the table. In cases of discrepancy between an identifier value in an allocation table and in the REGISTERED AS construct of a GDMO template, the template value takes precedence.

The column labeled *Arc* shows the value allocated to the arc subsequent to the invariant part, which completes the object identifier allocated. The column labeled *Purpose* contains a text description of the information object, and, in the case of current allocations, a reference to the location of the definition of the information object in the standard. The column labeled *Status* shows the status of the allocated values, using the following convention:

R *Reserved.* The object identifier value is reserved for future use by this standard.

C *Current.* The object identifier value has been allocated to an information object that is defined within the current revision of the standard.

D *Deprecated.* The object identifier value has been allocated to an information object that was defined in a previous revision of the standard, and whose use is now deprecated.

E.2 Allocation tables

Allocations for Standard-specific extensions.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) standardSpecificExtensions(0)}

ARC	PURPOSE	STATUS
None allocated	N/A	N/A

Allocations for ASN.1 module identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) asn1Module(2)}

ARC	PURPOSE	STATUS
None allocated	N/A	N/A

[2]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

Allocations for Managed object class identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) managedObjectClass(3)}

ARC	PURPOSE	STATUS
llcsap(1)	Single LLC SAP	C
llcconnectionless(2)	Type 1 operation at an LLC SAP	C
llcconnection2(3)	Type 2 connection	C
llcconnection2ivmo(4)	Type 2 connection initial values	C
llcconnectionlessack(5)	Type 3 operation	C
llcconnectionlessAckIVMO(6)	Type 3 operation initial values	C
rdesetup(7)	RDE operating parameters	C
rdepair(8)	RDE communicating LSAP pair	C
llcstation(9)	Entire station	C

Allocations for Package identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) packages(4)}

ARC	PURPOSE	STATUS
llcsapp(1)	LLC SAP type-independent characteristics	C
llcdupaddress(2)	Duplicate address detection option	C
llctype3(3)	Type 3 operation	C
llcbuffer(4)	Buffer management	C
pdusdiscarded(5)	Counts and reasons for PDU discard	C
rdep(6)	RDE configuration and version	C
type1acknowledgementtimertimeouts(7)	ACK timer expirations	C
llcsapcomponent(8)	Connectionless (Type 1) operation	C
llcconnection2p(9)	Connection-oriented (Type 2) operation	C
connectionroute(10)	Connection routing	C
type2disc(11)	DISC PDU counts	C
type2dm(12)	DM PDU counts	C
type2flowcontrol(13)	Flow control parameters	C
type2frmr(14)	FRMR PDU counts	C
type2i(15)	I PDU counts	C
type2pdus(16)	Type 2 PDU discards and retransmissions	C
type2rej(17)	REJ PDU counts	C

Allocations for Package identifiers. (*Continued*)
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) packages(4)}

ARC	PURPOSE	STATUS
type2rnr(18)	RNR PDU counts	C
type2rr(19)	RR PDU counts	C
type2sabme(20)	SABME PDU counts	C
type2ua(21)	UA PDU counts	C
type2violation(22)	Type 2 event notifications	C
llcconnection2ivmop(23)	Initial values for Type 2 operation	C
llcconnectionlessackp(24)	Acknowledged Connectionless (Type 3) operation	C
type3command(25)	Type 3 commands and status	C
type3noresponse(26)	Type 3 delivery failures	C
type3response(27)	Type 3 responses and status	C
type3retransmissions(28)	Type 3 retransmissions	C
type3violation(29)	Type 3 protocol violations	C
llcconnectionlessackivmo(30)	Initial values for Type 2 operation	C
rdesetupp(31)	RDE parameter settings	C
rdepairp(32)	RDE service for an LSAP pair	C
llcstationp(33)	Station type-independent characteristics	C
connectionbusy(34)	Type 2 connection busy conditions	C
connectionReset(35)	Type 2 connection resets	C
type2iack(36)	Type 2 connection acknowledgements	C
type2OptTolIPDU(37)	Toleration of Duplicate I PDUs option	C

Allocations for Parameter identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) parameter(5)}

ARC	PURPOSE	STATUS
None allocated	N/A	N/A

Allocations for Name binding identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) nameBinding(6)}

ARC	PURPOSE	STATUS
llcsapnb(1)	Binding for LLC SAP	C
llcconnectionlessnb(2)	Binding for Type 1 operation	C
llcconnection2nb(3)	Binding for Type 2 connections	C
llcconnection2ivmonb(4)	Binding for Type 2 initial values	C
llcconnectionlessacknb(5)	Binding for Type 3 operation	C
llcconnectionlessackivmonb(6)	Binding for Type 3 initial values	C
rdesetupnb(7)	Binding for RDE operating parameters	C
rdepairnb(8)	Binding for RDE supported LSAP pair	C
resourcetypeid(9)	Binding for the Resource Type ID managed object	C
llcstationnb(10)	Binding for station	C

Allocations for Attribute identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)}

ARC	PURPOSE	STATUS
avgbufferusesize(0)	Average buffer space in use	C
bufferproblems(1)	PDUs discarded due to buffer shortage	C
buffersize(2)	Buffer space available	C
inactivelsap(3)	PDUs discarded due to inactive DSAP	C
llcname(4)	Name of lLCStation managed object	C
maxbufferusesize(5)	Maximum buffer space in use	C
maximumlsapsconfigured(6)	Maximum concurrently supported LSAPs	C
maximumpdun3(7)	Maximum size of a Type 3 command	C
maximumretransmissions4(8)	Type 3 retransmission limit	C
numberofactivelsaps(9)	LLC SAPs currently active	C
pdusdiscarded(10)	Discarded invalid PDUs	C
rde(11)	RDE use at an LSAP	C
receivevariablelifetime(12)	Type 3 receive state variable duration	C
status(13)	Station status	C
strindicator(14)	Spanning tree routing used by RDE	C
supportedservicestypes(15)	Types of LLC operation supported	C
transmitvariablelifetime(16)	Type 3 transmit state variable duration	C

Allocations for Attribute identifiers. (*Continued*)
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)}

ARC	PURPOSE	STATUS
type1AcknowledgeTimeoutValue(17)	Ack timer period	C
type1acknowledgementtimertimeouts(18)	Ack timer expirations	C
type1maximumretrycount(19)	Duplicate address checking retry limit	C
type3acknowledgetimeoutvalue(20)	Type 3 acknowledgement timer period	C
type3retransmissions(21)	Type 3 retransmissions	C
versionnumber(22)	RDE protocol version	C
llcclessname(23)	Name of lLCConnectionless managed object	C
maximumllcinformationfieldsize(24)	Maximum SDU length	C
testreceivedabbresponse(25)	Truncated TEST responses received	C
testreceivedcommand(26)	TEST commands received	C
testreceivedresponse(27)	TEST responses received	C
testsentabbresponse(28)	Truncated TEST responses sent	C
testsentcommand(29)	TEST commands transmitted	C
testsentresponse(30)	TEST responses transmitted	C
uireceived(31)	UI PDUs received and passed to user	C
uisent(32)	UI PDUs transmitted	C
xidreceivedcommand(33)	XID commands received	C
xidreceivedresponse(34)	XID responses receive	C
xidsentcommand(35)	XID commands transmitted	C
xidsentresponse(36)	XID responses transmitted	C
llcconnectionname2(37)	Name of lLCConnection2 managed object	C
kstep(38)	Flow control parameter	C
maximumretransmissions(39)	Maximum retransmissions for connection	C
maxSendWindowSize(40)	Connection maximum send window size	C
pdusdiscarded1(41)	PDU discard due to connection resource shortage	C
pdusdiscarded2(42)	PDU discard due to incorrect sequence number	C
pduretransmissions(43)	PDU retransmissions for connection	C
receivewindowsize(44)	Connection receive window size	C
route(45)	Route used for connection	C
sendwindowsize(46)	Connection current send window size	C
type2acknowledgetimeoutvalue(47)	Connection acknowledgement timer period	C

Allocations for Attribute identifiers. (*Continued*)
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)}

ARC	PURPOSE	STATUS
type2busystatetimeoutvalue(48)	Connection busy timer period	C
type2pbittimeoutvalue(49)	Connection P-bit timer period	C
type2receiveddisc(50)	DISC PDUs received	C
type2receiveddm(51)	DM PDUs received	C
type2receivedfrmr(52)	FRMR PDUs received	C
type2receivedI(53)	I PDUs received	C
type2receivedrej(54)	REJ PDUs received	C
type2receivedrnr(55)	RNR PDUs received	C
type2receivedrr(56)	RR PDUs received	C
type2receivedsabme(57)	SABME PDUs received	C
type2recevedua(58)	UA PDUs received	C
type2rejecttimeoutvalue(59)	Connection reject timer period	C
type2sentdisc(60)	DISC PDUs transmitted	C
type2sentdm(61)	DM PDUs transmitted	C
type2sentfrmr(62)	FRMR PDUs transmitted	C
type2sentI(63)	PDUs transmitted	C
type2sentrej(64)	REJ PDUs transmitted	C
type2sentrnr(65)	RNR PDUs transmitted	C
type2sentrr(66)	RR PDUs transmitted	C
type2sentsabme(67)	SABME PDUs transmitted	C
type2sentua(68)	UA PDUs transmitted	C
type2violation(69)	Connection protocol violations	C
llcconnection2ivmoname(70)	Name of lLCConnection2IVMO managed object	C
llcconnectionlessackname2(71)	Name of lLCConnectionlessAck managed object	C
type3commandip(72)	Type 3 command status IP	C
type3comandit(73)	Type 3 command status IT	C
type3commandok(74)	Type 3 command status OK	C
type3commandpe(75)	Type 3 command status PE	C
type3commandrs(76)	Type 3 command status RS	C
type3commandue(77)	Type 3 command status UE	C
type3commandun(78)	Type 3 command status UN	C

Allocations for Attribute identifiers. (*Continued*)
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)}

ARC	PURPOSE	STATUS
type3noresponse(79)	Type 3 failure to obtain response	C
type3receiveresources(80)	Enable or disable receipt of Type 3 data	C
type3receivedaccommand(81)	Received Type 3 commands	C
type3sentaccommand(82)	Transmitted Type 3 commands	C
type3responseip(83)	Type 3 response status IP	C
type3responseit(84)	Type 3 response status IT	C
type3responsene(85)	Type 3 response status NE	C
type3responsenr(86)	Type 3 response status NR	C
type3responseok(87)	Type 3 response status OK	C
type3responsers(88)	Type 3 response status RS	C
type3responseue(89)	Type 3 response status UE	C
type3responseun(90)	Type 3 response status UN	C
type3retransmissions(91)	Type 3 retransmissions	C
type3violation(92)	Type 3 protocol violations	C
llcconnectionlessackivmoname(93)	Name of lLCConnectionlessAckIVMO managed object	C
agingEnabled(94)	Route ageing enabled	C
agingValue(95)	Route age limit	C
maximumRouteDescriptors(96)	Route length limit	C
maximumResponseTime(97)	RDE response time limit	C
minimumPDUSize(98)	Required PDU length supported by source route	C
rdehold(99)	RDE PDU hold policy during rerouting	C
rdereplace(100)	RDE policy for allowing remote route reselection	C
rdesetupname(101)	Name of rDESetup managed object	C
resetOnTestEnabled(102)	RDE rerouting on receipt of TEST	C
rdepaira(103)	RDE identification of LSAP pair	C
discardcounter(104)	Route changes	C
nsrpducounter(105)	Non-source-routed PDUs	C
nsrselectedcounter(106)	Acceptable source routes not used	C
rif(107)	Routing information field	C
sRFPDUcounter(108)	Source-routed PDUs	C
querycounter(109)	Route Query commands generated	C

Allocations for Attribute identifiers. (*Continued*)
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attribute(7)}

ARC	PURPOSE	STATUS
llcsapname(110)	Name of lLCSAP managed object	C
llcaddress(111)	Individual LLC address of LSAP	C
localBusy(112)	Connection local busy conditions	C
localReset(113)	Connection resets by local user	C
providerReset(114)	Connection resets due to error	C
receivedacks(115)	I PDUs transmitted and acknowledged	C
remoteBusy(116)	Connection remote busy conditions	C
remoteReset(117)	Connection resets by partner	C
sentAcks(118)	I PDUs received and acknowledged	C
optionalTolerationIPDUs(119)	Toleration of I PDUs use	C
dupIPDUsReceived(120)	Duplicate I PDUs received	C

Allocations for Attribute group identifiers.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) attributeGroup(8)}

ARC	PURPOSE	STATUS
None allocated	N/A	N/A

Allocations for Action types.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) action(9)}

ARC	PURPOSE	STATUS
reinitialize(0)	Reinitialize LLC sublayer	C
testsendcommand(2)	Send TEST	C
xidsendcommand(3)	Send XID	C
correlatorexchange(4)	Establish correlator for connection events	C
readcounters(5)	Read RDE counters	C
flushroute(6)	Delete an RDE-maintained route	C

Allocations for Notification types.
Invariant part of object identifier value =
{iso(1) member-body(2) us(840) ieee802-2(10032) notification(10)}

ARC	PURPOSE	STATUS
llcstationevent(0)	Station events	C
communicationalarm(1)	Connection alarm reporting	C
llcconnection2event(2)	Connection events	C
llcclessackevent(3)	Type 3 operation conditions	C