IEEE Std 802.1D™- 2004
(Revision of IEEE Std 802.1D-1998)

IEEE Standards

802.1D™

IEEE Standard for
Local and metropolitan area networks

Media Access Control (MAC) Bridges

IEEE Computer Society

Sponsored by the
LAN/MAN Standards Committee

IEEE
3 Park Avenue, New York, NY 10016-5997, USA

9 June 2004

Print: SH95213
PDF: SS95213

Recognized as an
American National Standard (ANSI)

IEEE Std 802.1D™-2004
(Revision of IEEE Std 802.1D-1998)

IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges

Sponsor
LAN MAN Standards Committee
of the
IEEE Computer Society

Approved 4 June 2004
American National Standard Institute

Approved 9 February 2004
IEEE-SA Standards Board

Abstract: An architecture for the interconnection of IEEE 802® Local Area Networks (LANs) below the MAC Service boundary is defined. MAC Bridges, as specified by this standard, allow communications between end stations attached to separate LANs, each with its own separate MAC, to be transparent to logical link control (LLC) and network layer protocols, just as if the stations were attached to the same LAN.

Keywords: active topology, filtering, GARP, GMRP, LANs, local area networks, MAC Bridges, MAC Service, MANs, metropolitan area networks, multicast registration, transparent bridging, quality of service, RSTP, spanning tree

---

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Print: ISBN 0-7381-3881-5 SH95213
PDF: ISBN 0-7381-3982-3 SS95213

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "AS IS."

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.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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 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 societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE. Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

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

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

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

[This introduction is not part of IEEE Std 802.1D-2004, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges.]

The MAC Bridge standardization activities that resulted in the development of IEEE Std 802.1D-1990 (subsequently republished as IEEE Std 802.1D, 1993 Edition [ISO/IEC 10038:1993] and IEEE Std 802.1D, 1998 Edition [ISO/IEC 15802-3: 1998]) specified an architecture and protocol for the interconnection of IEEE 802 LANs below the MAC Service boundary.

The 2004 revision of this standard incorporates two amendments into the 1998 Edition:

a) IEEE Std 802.1t-2001, technical and editorial corrections to the 1998 Edition; and
b) IEEE Std 802.1w-2001, Rapid Reconfiguration, which specified the Rapid Spanning Tree Algorithm and Protocol (RSTP).

In addition, this revision includes further technical and editorial corrections, and removes the original Spanning Tree Protocol (STP) as a conformance option.

Relationship between IEEE Std 802.1D and IEEE Std 802.1Q

Another IEEE standard, IEEE Std 802.1Q™-2003, extends the concepts of MAC Bridging and filtering services to support the definition and management of Virtual LANs (VLANs).

The capabilities defined in IEEE Std 802.1Q-2003 include the definition of a VLAN frame format that is able to carry VLAN identification and user priority information over LAN technologies, such as CSMA/CD, that have no inherent capability to signal priority information. This information is carried in an additional header field, known as the *Tag Header*, which is inserted immediately following the Destination MAC Address, and Source MAC Address (and Routing Information field, if present) of the original frame. IEEE Std 802.1Q-2003 extends the priority handling aspects of this standard to make use of the ability of the VLAN frame format to carry user priority information end to end across any set of concatenated underlying MACs.

The VLAN Bridging specification contained in IEEE Std 802.1Q-2003 is independent of this standard, in the sense that IEEE Std 802.1Q-2003 contains its own statement of the conformance requirements for VLAN Bridges. However, IEEE Std 802.1Q-2003 makes use of many of the elements of the specification contained in this standard, in particular

a) The Bridge architecture
b) The Internal Sublayer Service, and the specification of its provision by IEEE 802 LAN MACs
c) The major features of the operation of the forwarding process
d) The Rapid Spanning Tree Protocol
e) The Generic Attribute Registration Protocol (GARP)
f) The GARP Multicast Registration Protocol (GMRP)

Since the original Spanning Tree Protocol (STP) has been removed from the 2004 revision of IEEE Std 802.1D, an implementation of RSTP is required for any claim of conformance for an implementation of IEEE Std 802.1Q-2003 that refers to the current revision of IEEE Std 802.1D unless that implementation includes the Multiple Spanning Tree Protocol (MSTP) specified in IEEE Std 802.1Q-2003. MSTP is based on RSTP, extended to provide support for multiple spanning trees.

Notice to users

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

Interpretations

Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

Patents

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 or patent applications for which a license may be required to implement an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Participants

At the time this standard was completed, the 802.1D working group had the following membership:

Tony Jeffree, *Chair and Editor*
Paul Congdon, *Vice-Chair*
Mick Seaman, *Interworking Task Group Chair and Editor*

Les Bell
Paul Bottorff
Jim Burns
Marco Carugi
Dirceu Cavendish
Arjan de Heer
Anush Elangovan
Hesham Elbakoury
David Elie-Dit-Cosaque
Norm Finn
David Frattura
Gerard Goubert
Stephen Haddock
Atsushi Iwata
Neil Jarvis
Manu Kaycee
Hal Keen
Bill Lane
Roger Lapuh
Loren Larsen
Yannick Le Goff
Marcus Leech
Mahalingam Mani
Dinesh Mohan
Bob Moskowitz
Don O Connor
Don Pannell
Glenn Parsons
Ken Patton
Allyn Romanow
Dan Romascanu
Jessy V. Rouyer
Ali Sajassi
Dolors Sala
Muneyoshi Suzuki
Jonathan Thatcher
Michel Thorsen
Dennis Volpano
Karl Weber
Ludwig Winkel
Michael D. Wright

The following members of the balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Butch Anton
Edward Carley
Clint Chaplin
Sunghyun Choi
Keith Chow
Christopher Cooke
Wael Diab
Thomas Dineen
Sourav Dutta
Clint Early
Will Foulds
David Frattura
Anoop Ghanwani
Robert M. Grow
Stephen Haddock

Julian Ho
Atsushi Ito
Peeya Iwagoshi
Tony Jeffree
Stanley Johnson
Stuart Kerry
Shyam Kaluve
Cees Klik
Kshitij Kumar
Pi-Cheng Law
Randolph Little
Ryan Madron
Nikolai Malykh
Kyle Maus
George Miao

Rajesh Moorkath
Charles Ngethe
Satoshi Obara
Stephen Palm
Ashley Peacock
Subbu Ponnuswamy
Vikram Punj
Maximilian Riegel
Floyd Ross
Mick Seaman
Gil Shultz
Adrian Stephens
Scott Valcourt
Dmitri Varsanofiev
Michael D. Wright
Oren Yuen

When the IEEE-SA Standards Board approved this standard on 9 February 2004, it had the following membership:

Don Wright, *Chair*

Chuck Adams,
Stephen Berger
Mark D. Bowman
Joseph Bruder
Bob Davis
Roberto de Boisson
Julian Forster*
Judith Gorman
Arnold M. Greenspan

Mark S. Halpin
Raymond Hapeman
Richard J. Holleman
Richard Hulett
Lowell Johnson
Hermann Koch
Joseph Koepfinger*
Thomas J. McGean
Steve M. Mills

Daleep Mohla
Paul Nikolich
T. W. Olsen
Ronald C. Petersen
Gary S. Robinson
Frank Stone
Malcolm V. Thaden
Doug Topping
Joe D. Watson

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Satish K. Aggarwal, *NRC Representative*
Richard DeBlasio, *DOE Representative*
Alan Cookson, *NIST Representative*

Michelle D. Turner
IEEE Standards Project Editor

Contents

IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges

1. Overview

1.1 Introduction

IEEE 802 Local Area Networks (or LANs; see 3.4) of all types can be connected together using MAC Bridges. The Bridged Local Area Network created allows the interconnection of stations as if they were attached to a single LAN, even if they are attached to separate LANs each with its own independent MAC. A MAC Bridge operates below the MAC Service Boundary, and is transparent to protocols operating above this boundary, in the Logical Link Control (LLC) sublayer or Network Layer (ISO/IEC 7498-1: 1994[1]). The presence of one or more MAC Bridges can lead to differences in the Quality of Service (QoS) provided by the MAC sublayer; it is only because of such differences that MAC Bridge operation is not fully transparent.

A Bridged Local Area Network can provide for

a) The interconnection of stations attached to LANs of different MAC types.
b) An effective increase in the physical extent, the number of permissible attachments, or the total performance of a LAN.
c) Partitioning of the physical LAN for administrative or maintenance reasons.
d) Validation of access to the LAN.
e) Increased availability of the MAC Service in the face of reconfiguration or failure of network components.

1.2 Scope

For the purpose of compatible interconnection of data-processing equipment using the IEEE 802 MAC Service supported by interconnected IEEE 802 LANs (see 3.4) using different or identical Media Access Control methods, this standard specifies a general method for the operation of MAC Bridges. To this end this standard

a) Positions the bridging function within an architectural description of the MAC Sublayer.

b) Defines the principles of operation of the MAC Bridge in terms of the support and preservation of the MAC Service, and the maintenance of QoS.

c) Specifies the MAC Internal Sublayer Service provided by individual LANs to the Media Access Method Independent Functions that provide frame relay in the Bridge.

[1] Information about references can be found in Clause 2.

d) Identifies the functions to be performed by Bridges, and provides an architectural model of the internal operation of a Bridge in terms of Processes and Entities that provide those functions.

e) Establishes the requirements for a protocol between the Bridges in a Bridged Local Area Network to configure the network, and specifies the distributed computation of a Spanning Tree active topology.

f) Establishes the requirements for a protocol between Bridges in a Bridged Local Area Network to configure multicast filtering information, and specifies the means of registering and distributing multicast filtering information by means of the GARP Multicast Registration Protocol (GMRP).

g) Establishes the requirements for Bridge Management in the Bridged Local Area Network, identifying the managed objects and defining the management operations.

h) Specifies performance requirements and recommends default values and applicable ranges for the operational parameters of a Bridge.

i) Specifies the requirements to be satisfied by equipment claiming conformance to this standard.

j) Specifies criteria for the use of MAC-specific bridging methods.

This standard specifies the operation of MAC Bridges that attach directly to IEEE 802 LANs, as specified in the relevant MAC standards for the MAC technology or technologies implemented.

2. References

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

ANSI X3.159-1989, American National Standards for Information Systems—Programming Language—C.[2]

IEEE Std 802®-2001, IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture.[3]

IEEE Std 802.1H™, 1997 Edition [ISO/IEC 11802-5: 1997], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Technical reports and guidelines—Part 5: Media Access Control (MAC) Bridging of Ethernet V2.0 in Local Area Networks.[4]

IEEE Std 802.1Q™-2003, IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks.

IEEE Std 802.1X™-2001, IEEE Standards for Local and Metropolitan Area Networks—Port Based Network Access Control.

IEEE Std 802.2™, 1998 Edition [ISO/IEC 8802-2: 1998], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 2: Logical link control.[5]

IEEE Std 802.3™-2002, IEEE Standard for Local and Metropolitan Area Networks, Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications: Aggregation of Multiple Link Segments.

IEEE Std 802.5, 1998 Edition [ISO/IEC 8802-5: 1998], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 5: Token ring access method and physical layer specifications.

IEEE Std 802.11, 1999 Edition [ISO/IEC 8802-11: 1999], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.

IETF RFC 2236, Fenner, Internet Group Management Protocol (IGMP), Version 2, November 1975.[6]

IETF RFC 1493, Decker, Langille, Rijsinghani and McCloughrie, Definitions of Managed Objects for Bridges, July 1993.

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

[3] IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA. IEEE publications can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org.

[4] The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

[5] ISO [IEEE] and ISO/IEC [IEEE] documents are available from ISO Central Secretariat, 1 rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse; and from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA. ISO [IEEE] and ISO/IEC [IEEE] documents can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org.

[6] Internet RFCs are available from the Internet Engineering Task Force website at http://www.ietf.org/rfc.html.

IETF RFC 2233, The Interfaces Group MIB using SMIv2, McCloghrie, K., Kastenholz, F., November 1997.

IETF RFC 2674, Bell, Smith, Langille, Rijsinghani, McCloghrie, Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering and Virtual LAN Extensions, August 1999.

ISO/IEC 7498-1: 1994, Information processing systems—Open Systems Interconnection—Basic Reference Model—Part 1: The Basic Model.[7]

ISO/IEC 8824-1: 2002, Information technology—Abstract Syntax Notation One (ASN.1): Specification of basic notation.

ISO/IEC 8825-1: 2002, Information technology—ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).

ISO/IEC 9595: 1998, Information technology—Open Systems Interconnection—Common management information service.

ISO 9314-2: 1989, Information processing systems—Fibre Distributed Data Interface (FDDI)—Part 2: Token Ring Media Access Control (MAC).

ISO/IEC 9596-1: 1998, Information technology—Open Systems Interconnection—Common management information protocol—Part 1: Specification.

ISO/IEC TR 11802-1: 1997, Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Technical reports and guidelines—Part 1: The structure and coding of Logical Link Control addresses in Local Area Networks.

ISO/IEC TR 11802-2: 2002, Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Technical reports and guidelines—Part 2: Standard Group MAC addresses.

ISO/IEC 14882: 2003, Information Technology—Programming languages—C++.

ISO/IEC 15802-1: 1995, Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Common specifications—Part 1: Medium Access Control (MAC) service definition.

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

3. Definitions

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

3.1 Bridged Local Area Network

A concatenation of individual IEEE 802 LANs interconnected by MAC Bridges.

NOTE—Unless explicitly specified, the use of the word *network* in this standard refers to a Bridged Local Area Network. The term Bridged Local Area Network is not otherwise abbreviated. The term *Local Area Network* and the abbreviation LAN are used exclusively to refer to an individual LAN specified by a MAC technology without the inclusion of Bridges. This precise use of terminology within this specification allows a Bridged Local Area Network to be distinguished from an individual LAN that has been bridged to other LANs in the network. In more general usage such precise terminology is not required, as it is a explicit goal of this standard that Bridges are transparent to the users of the MAC Service.

3.2 Expedited traffic

Traffic that requires preferential treatment as a consequence of jitter, latency, or throughput constraints, or as a consequence of management policy.

3.3 Group

A Group associates

a) A group MAC address,

b) A set of properties that define membership characteristics, and

c) A set of properties that define the forwarding/filtering behavior of a Bridge with respect to frames destined for members of that group MAC address.

with a set of end stations that all wish to receive information destined for that group MAC address. Members of such a set of end stations are said to be *Group members.*

A Group is said to *exist* if the properties associated with that Group are visible in an entry in the Filtering Database of a Bridge, or in the GARP state machines that characterize the state of the Group; a Group is said to *have members* if the properties of the Group indicate that members of the Group can be reached through specific Ports of the Bridge.

NOTE—An example of the information that Group members might wish to receive is a multicast video data stream.

3.4 IEEE 802 Local Area Network (LAN)

IEEE 802 LANs (also referred to in the text simply as LANs) are LAN technologies that provide a MAC Service equivalent to the MAC Service defined in ISO/IEC 15802-1. IEEE 802 LANs include IEEE Std 802.3 (CSMA/CD), IEEE Std 802.5 (Token Ring), IEEE Std 802.11 (Wireless), and ISO 9314-2 (FDDI) LANs.

4. Abbreviations

The following abbreviations are used in this standard.

BPDU	Bridge Protocol Data Unit
CRC	Cyclic Redundancy Check
FCS	Frame Check Sequence
GARP	Generic Attribute Registration Protocol
GARP PDU	GARP Protocol Data Unit
Gb/s	Gigabit per second (1 Gb/s is equivalent to 1 000 000 000 bits per second)
GID	GARP Information Declaration
GIP	GARP Information Propagation
GMRP	GARP Multicast Registration Protocol
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
kb/s	Kilobit per second (1 kb/s is equivalent to 1000 bits per second)
MAC	Media Access Control
Mb/s	Megabit per second (1 Mb/s is equivalent to 1 000 000 bits per second)
RSTP	Rapid Spanning Tree Algorithm and Protocol
RST BPDU	Rapid Spanning Tree Bridge Protocol Data Unit
STP	Spanning Tree Algorithm and Protocol
Tb/s	Terabit per second (1 Tb/s is equivalent to 1 000 000 Mb/s)
TCN	Topology Change Notification

5. Conformance

5.1 Required capabilities

A MAC Bridge for which conformance to this standard is claimed shall

a) Conform to the relevant MAC standards technologies implemented at its Ports, as specified in 6.4 and 6.5.
b) Conform to IEEE Std 802.2 for the implementation of a class of LLC supporting Type 1 operation as required by 7.3 and 7.1.2.
c) Relay and filter frames as described in 7.1 and specified in 7.5, 7.6, 7.7, and 7.9.
d) Maintain the information required to support Basic Filtering Services, as described in 6.6 and 7.1 and specified in 7.5, 7.8, and 7.9.
e) Conform to the provisions for addressing specified in 7.12.
f) Implement the Rapid Spanning Tree Protocol, as specified in Clause 17.
g) Encode transmitted BPDUs and validate received BPDUs as specified in Clause 9.
h) Specify the following parameters of the implementation
    1) Filtering Database Size, the maximum number of entries.
    2) Permanent Database Size, the maximum number of entries.
i) Specify the following performance characteristics of the implementation
    1) A Guaranteed Port Filtering Rate for each Bridge Port
    2) A Guaranteed Bridge Relaying Rate for the Bridge
    3) The related time intervals T_F and T_R for these parameters as specified in Clause 16.

    Operation of the Bridge within the specified parameters shall not violate any of the other conformance provisions of this standard.

5.2 Optional capabilities

A MAC Bridge for which conformance to this standard is claimed may

a) Support management of the Bridge. Bridges claiming to support management shall support all the management objects and operations for all implemented capabilities as defined in Clause 14.
b) Support the use of a remote management protocol. Bridges claiming to support remote management shall
    1) State which remote management protocol standard(s) or specification(s) are supported.
    2) State which standard(s) or specification(s) for managed object definitions and encodings are supported for use by the remote management protocol.
c) Support multiple Traffic Classes for relaying and filtering frames through one or more outbound Ports, controlling the mapping of the priority of forwarded frames as specified in 6.4, 7.5.1, and 7.7.
d) Support Extended Filtering Services for relaying and filtering frames as described in 7.1 and specified in 7.5, 7.6, 7.7, and 7.9. Bridges claiming to support of Extended Filtering Services shall
    1) Implement the GARP Multicast Registration Protocol (GMRP), as specified in Clause 10.
    2) Implement the Generic Attribute Registration Protocol (GARP), in support of the GMRP Application as specified in Clause 12.

NOTE—The term capability is used to describe a set of related detailed provisions of this standard. Each capability can comprise both mandatory provisions, required if implementation of the capability is to be claimed, and optional provisions. Each detailed provision is specified in one or more of the other clauses of this standard. The Protocol Implementation Conformance Statement (PICS), described in 5.3, provides a useful checklist of these provisions.

5.3 Protocol Implementation Conformance Statement

The supplier of an implementation that is claimed to conform to this standard shall complete a copy of the PICS proforma provided in Annex A and shall provide the information necessary to identify both the supplier and the implementation.

5.4 Recommendations

Support of the relevant managed objects and management operations specified in Clause 14, and support of the use of SNMP to access them, is highly recommended.

5.5 MAC-specific bridging methods

MAC-specific bridging methods may exist. Use of a MAC-specific bridging method and the method specified in this standard on the same LAN shall

a) Not prevent communication between stations in a Bridged Local Area Network.
b) Preserve the MAC Service.
c) Preserve the characteristics of each bridging method within its own domain.
d) Allow both bridging techniques to coexist simultaneously on a LAN without adverse interaction.

6. Support of the MAC Service

MAC Bridges interconnect the separate IEEE 802 LANs that compose a Bridged Local Area Network by relaying and filtering frames between the separate MACs of the bridged LANs. The position of the bridging function within the MAC Sublayer is shown in Figure 6-1.

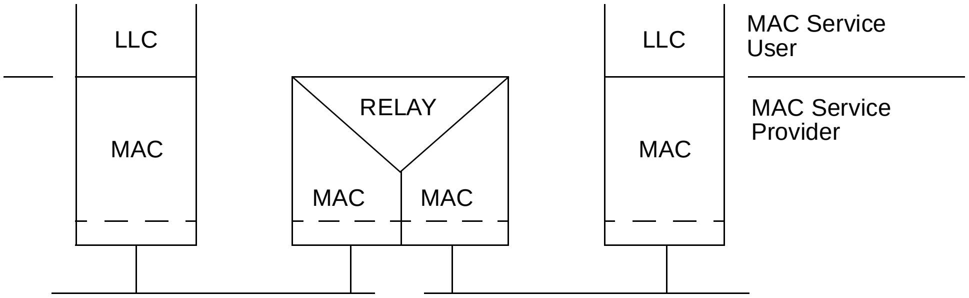


Figure 6-1—Internal organization of the MAC sublayer

This clause discusses the following aspects of service provision:

a) Provision of the MAC Service to end stations.
b) Preservation of the MAC Service.
c) Maintenance of QoS.
d) Provision of the Internal Sublayer Service within the MAC Bridge.
e) Support of the Internal Sublayer Service by specific MAC procedures.
f) Filtering services.

6.1 Support of the MAC Service

The MAC Service provided to end stations attached to a Bridged Local Area Network is the (unconfirmed) connectionless-mode MAC Service defined in ISO/IEC 15802-1. The MAC Service is defined as an abstraction of the features common to a number of specific MAC Services; it describes the transfer of user data between source and destination end stations, via MA-UNITDATA request primitives and corresponding MA-UNITDATA indication primitives issued at MAC service access points. Each MA-UNITDATA request and indication primitive has four parameters: Destination Address, Source Address, MAC Service data unit (MSDU), and Priority.

To improve the availability of the MAC Service to end stations and support network management, MAC Bridges can be configured:

a) To provide redundant paths between end stations to enable the network to continue to provide the MAC Service in the event of component failure (of Bridge or LAN).
b) So that the paths supported between end stations are predictable and configurable given the availability of network components.

A MAC Bridge may restrict the provision of the MAC Service to authenticated and authorized devices. Unauthorized devices can be denied access to a bridged LAN, except as necessary for the purpose of conducting protocol exchanges required by an authentication process.

NOTE—Authentication and authorization to access a LAN may be achieved by administrative or management mechanisms, or by means of an active authorization mechanism, such as is defined in IEEE Std 802.1X.

6.2 Preservation of the MAC Service

The MAC Service provided by a Bridged Local Area Network is similar to that provided by a single LAN (6.3). In consequence,

a) A Bridge is not directly addressed by communicating end stations, except as an end station for management purposes—frames transmitted between end stations carry the MAC Address of the peer-end station in their Destination Address field, not a MAC Address of the Bridge.
b) All MAC Addresses need to be unique within the network.
c) MAC Addresses of end stations are not restricted by the topology and configuration of the network.

6.3 Quality of Service maintenance

Quality of Service comprises the following:

a) Service availability
b) Frame loss
c) Frame misordering
d) Frame duplication
e) Frame transit delay
f) Frame lifetime
g) Undetected frame error rate
h) Maximum service data unit size supported
i) Frame priority
j) Throughput

6.3.1 Service availability

Service availability is measured as a fraction of some total time during which the MAC Service is provided. The operation of a Bridge can increase or lower the service availability.

Service availability can be increased by automatic reconfiguration of the network (see Clause 17) to avoid the use of a failed component (e.g., repeater, cable, or connector) in the data path. Service availability can be lowered by the failure of a Bridge or by a Bridge filtering frames.

A Bridge can discard frames (6.3.2) to preserve other aspects of the MAC Service (6.3.3 and 6.3.4) during reconfiguration, lowering service availability for end stations that do not benefit from the reconfiguration. If an end station moves, it can then be unable to receive frames from other end stations until the filtering information held by the Bridges and used to localize traffic is updated. To minimize service denial, filtering information that has been dynamically learned can be modified when reconfiguration takes place (17.11). However, filtering information that is statically configured cannot be modified in this way.

A Bridge may deny service and discard frames to prevent network access by unauthorized devices.

To maximize the service availability, no loss of service or delay in service provision is caused by Bridges, except as a consequence of a failure, removal, or insertion of a network component, or as a consequence of the movement of an end station, or as a consequence of an attempt to perform unauthorized access. These are regarded as extraordinary events. The operation of any additional protocol necessary to maintain the quality of the MAC Service is thus limited to the configuration of the Bridged Local Area Network, and is independent of individual instances of service provision.

NOTE—This is true only in the absence of admission control mechanisms, i.e., where the Bridges provide a “best effort” service. The specification and applicability of admission controls in Bridges is outside the scope of this standard.

6.3.2 Frame loss

The MAC Service does not guarantee the delivery of Service Data Units. Frames transmitted by a source station arrive, uncorrupted, at the destination station with high probability. The operation of a Bridge introduces minimal additional frame loss.

A frame transmitted by a source station can fail to reach its destination station as a result of the following:

a) Frame corruption during physical layer transmission or reception.
b) Frame discard by a Bridge because
    1) It is unable to transmit the frame within some maximum period of time, and is required to discard the frame to prevent the maximum frame lifetime (6.3.6) from being exceeded.
    2) It is unable to continue to store the frame due to exhaustion of internal buffering capacity as frames continue to arrive at a rate in excess of that at which they can be transmitted.
    3) The size of the service data unit carried by the frame exceeds the maximum supported by the MAC procedures employed on the LAN to which the frame is to be relayed.
    4) Changes in the physical topology of the network necessitate frame discard for a limited period of time to maintain other aspects of QoS (see 17.10).
    5) The device attached to the Port is not authorized for access to the network.
    6) The configuration of Static Filtering Entries in the Filtering Database (7.9.1) disallows the forwarding of frames with particular destination addresses on specific Ports.

NOTE—As Static Filtering Entries are associated with particular Ports or combinations of Ports, there is a possibility that mis-configuration of Static Filtering Entries will lead to unintended frame discard during or following automatic reconfiguration of the Bridged Local Area Network.

6.3.3 Frame misordering

The MAC Service (9.2 of ISO/IEC 15802-1) permits a negligible rate of reordering of frames with a given user priority for a given combination of destination address and source address. MA_UNITDATA.indication service primitives corresponding to MA_UNITDATA.request primitives, with the same requested priority and for the same combination of destination and source addresses, are received in the same order as the request primitives were processed.

NOTE 1—The Forwarding Process in Bridges (7.7) does not misorder or duplicate frames.

Where Bridges are capable of connecting the individual MACs in such a way that multiple paths between any source station–destination station pairs exist, the operation of a protocol is required to ensure that a single path is used.

NOTE 2—Frame misordering and duplication (6.3.4) does not occur during normal operation. When RSTP is configuring or reconfiguring the network (see Clause 17), there is an increased and implementation-dependent probability that frames that are in transit will be misordered or duplicated as network paths change, since a Bridge can buffer frames awaiting transmission through its Ports. Since the probability of duplication or misordering occurring as a result of reconfiguration is small, and the frequency of physical network failures leading to reconfiguration is also generally small, the degradation of the properties of the MAC service is considered to be negligible. Some LAN protocols, for example, LLC Type 2, are particularly sensitive to frame misordering and duplication; to support these the Force Protocol Version parameter (17.13.4) can be used to delay the transition of ports to a Forwarding state and further reduce the probability of such events. A more detailed discussion of misordering and duplication in RSTP can be found in Annex K (informative).

6.3.4 Frame duplication

The MAC Service (9.2 of ISO/IEC 15802-1) permits a negligible rate of duplication of frames. Bridges do not duplicate user data frames.

The potential for frame duplication in a Bridged Local Area Network arises through the possibility of multiple paths between source and destination end stations. Where Bridges can connect individual LANs to provide multiple paths, the operation of a protocol is required to ensure that a single path is used.

6.3.5 Frame transit delay

The MAC Service introduces a variable frame transit delay that is dependent on media types and media access control methods. Frame transit delay is the elapsed time between an MA_UNITDATA.request primitive and the corresponding MA_UNITDATA.indication primitive. Elapsed time values are calculated only on Service Data Units that are successfully transferred.

Since the MAC Service is provided at an abstract interface within an end station, it is not possible to specify the total frame transit delay precisely. It is, however, possible to measure the media access and frame transmission and reception, and the transit delay introduced by an intermediate system, in this case a Bridge.

The minimum additional transit delay introduced by a Bridge is the time taken to receive a frame plus that taken to access the media onto which the frame is to be relayed. The frame is completely received before it is relayed as the Frame Check Sequence (FCS) is to be calculated and the frame discarded if in error.

6.3.6 Frame lifetime

The MAC Service mandates an upper bound to the transit delay experienced for a particular instance of communication. This maximum frame lifetime is necessary to ensure the correct operation of higher layer protocols. The additional transit delay introduced by a Bridge is discussed in 6.3.5.

Since the information provided by the MAC Sublayer to a Bridge does not include the transit delay already experienced by any particular frame, Bridges discard frames to enforce a maximum delay in each Bridge. A recommended and an absolute maximum value are specified in Table 7-3.

6.3.7 Undetected frame error rate

The MAC Service introduces a very low undetected frame error rate in transmitted frames. Undetected errors are protected against by the use of an FCS that is appended to the frame by the MAC Sublayer of the source station prior to transmission, and checked by the destination station on reception.

The FCS calculated for a given service data unit is dependent on the media access control method. It is therefore necessary to recalculate the FCS within a Bridge providing a relay function between IEEE 802 MACs of dissimilar types where there are differences in the method of calculation and/or the coverage of the FCS, or changes to the data that is within the coverage of the FCS. This introduces the possibility of additional undetected errors arising from the operation of a Bridge.

NOTE—Application of the techniques described in Annex F (informative) allow an implementation to achieve an arbitrarily small increase in undetected frame error rate, even in cases where the data that is within the coverage of the FCS is changed.

6.3.8 Maximum Service Data Unit Size

The Maximum Service Data Unit Size that can be supported by an IEEE 802 LAN varies with the MAC method and its associated parameters (speed, electrical characteristics, etc.). It may be constrained by the owner of the LAN. The Maximum Service Data Unit Size supported by a Bridge between two LANs is the smaller of that supported by the LANs. No attempt is made by a Bridge to relay a frame to a LAN that does not support the size of Service Data Unit conveyed by that frame.

6.3.9 Frame priority

The MAC Service includes user priority as a QoS parameter. MA_UNITDATA.requests with a high priority may be given precedence over other request primitives made at the same station, or at other stations attached to the same LAN, and can give rise to earlier MA_UNITDATA.indications.

The MAC Sublayer maps the requested user priorities onto the access priorities supported by the individual media access method. The requested user priority can be conveyed to the destination station with the transmitted frame, using the priority signaling mechanisms inherent in some media access methods. Since not all IEEE 802 LAN MACs can signal the user priority associated with a frame, Bridges regenerate user priority based upon a combination of signaled information and configuration information held in the Bridge.

The transmission delay experienced by a frame in a Bridge comprises:

a) A queuing delay until the frame becomes first in line for transmission on the Port, in accordance with the procedure for selecting frames for transmission described in 7.7.4.

b) The access delay for transmission of the frame.

Queueing delays can be managed using user priority. Access delays can be managed using user priority in media access methods that support more than one access priority.

The Bridge maps user priority onto one or more traffic classes; Bridges that support more than one traffic class are able to support expedited classes of traffic. The Forwarding Process, 7.7, describes the use of user priority and traffic classes in MAC Bridges. Given the constraints placed upon frame misordering in a Bridge, as expressed in 6.3.3, the mappings of priority and traffic class are static.

NOTE 1—The term Traffic Class, as used in this standard, is used only in the context of the operation of the priority handling and queueing functions of the Forwarding Process, as described in 7.7. Any other meanings attached to this term in other contexts do not apply to the use of the term in this standard.

The ability to signal user priority in IEEE 802 LANs allows user priority to be carried with end-to-end significance across a Bridged Local Area Network. This, coupled with a consistent approach to the mapping of user priority to traffic classes and of user priority to access_priority, allows consistent use of priority information, according to the capabilities of the Bridges and MACs in the transmission path.

NOTE 2—IEEE Std 802.1Q™ defines a frame format and procedures that can be used to carry user priority across LAN MAC types that are not able to signal user priority. Use of the IEEE 802.1Q frame format allows the end-to-end significance of user priority to be maintained regardless of the ability of individual LAN MAC types to signal priority.

Under normal circumstances, user priority is not modified in transit through the relay function of a Bridge; however, network management can control how user priority is propagated. Table 7-1 provides the ability to map incoming user priority values on a per-Port basis. By default, the regenerated user priority is identical to the incoming user priority.

6.3.10 Throughput

The total throughput provided by a Bridged Local Area Network can be significantly greater than that provided by an equivalent single LAN. Bridges may localize traffic within the network by filtering frames. Filtering services are described in 6.6.

The throughput between end stations communicating through a Bridge can be lowered by frame discard due to the Bridge’s inability to transmit on the LAN to the destination at the required rate for an extended period.

6.4 Internal Sublayer Service provided within the MAC Bridge

The Internal Sublayer Service provided by a MAC entity to the MAC Relay Entity within a Bridge is that provided by the individual MAC for the LAN Port. This observes the appropriate MAC procedures and protocol for the LAN to which it attaches. No control frames, i.e., frames that do not convey MAC user data, are forwarded on any LAN other than that on which they originated.

The Internal Sublayer Service is derived from the MAC Service defined by ISO/IEC 15802-1 by augmenting that specification with elements necessary to the performance of the relay function. Within an end station, these additional elements are considered to be either below the MAC Service boundary, and pertinent only to the operation of the service provider; or local matters not forming part of the peer-to-peer nature of the MAC Service. Two parameters are added to the list of parameters associated with the MA_UNITDATA.request and MA_UNITDATA.indication primitives defined by ISO/IEC 15802-1. These are frame_type and frame_check_sequence. The definition of the Internal Sublayer Service does not add any new service primitives to those defined by the LAN MAC Service Definition.

6.4.1 Service primitives

The Internal Sublayer Service excludes MAC-specific features and procedures whose operation is confined to that of the individual LANs. The unit-data primitives that describe this service are

```
M_UNITDATA.indication    (
                         frame_type,
                         destination_address,
                         source_address,
                         mac_service_data_unit,
                         user_ priority,
                         frame_check_sequence
                         )
```

Each M_UNITDATA indication corresponds to the receipt of an error-free MAC frame from a LAN.

NOTE—Detailed specifications of error conditions in received frames are contained in the relevant MAC standards; for example, FCS errors, length errors, non-integral number of octets.

The **frame_type** parameter indicates the class of frame. The value of this parameter is one of user_data_ frame, mac_specific_frame, or reserved_frame.

The **destination_address** parameter is the address of an individual MAC entity or a group of MAC entities.

The **source_address** parameter is the individual address of the source MAC entity.

The **mac_service_data_unit** parameter is the service user data.

The **user_priority** parameter is the priority requested by the originating service user. The value of this parameter is in the range 0 through 7.

NOTE—The default user_priority value is 0. Values 1 through 7 form an ordered sequence of user_priorities, with 1 being the lowest value and 7 the highest. See 7.7.3 and Annex G (informative) for further explanation of the use of user_priority values.

The **frame_check_sequence** parameter is explicitly provided as a parameter of the primitive so that it can be used as a parameter to a related request primitive without recalculation.

The identification of the LAN from which particular frames are received is a local matter and is not expressed as a parameter of the service primitive.

```
M_UNITDATA.request      (
                        frame_type,
                        destination_address,
                        source_address,
                        mac_service_data_unit,
                        user_priority,
                        access_priority,
                        frame_check_sequence
                        )
```

A data request primitive is invoked to transmit a frame to an individual LAN.

The **frame_type** parameter indicates the class of frame.

The **destination_address** parameter is the address of an individual MAC entity or a group of MAC entities.

The **source_address** parameter is the individual address of the source MAC entity.

The **mac_service_data_unit** parameter is the service user data.

The **user_priority** parameter is the priority requested by the originating service user. The value of this parameter is in the range 0 (lowest) through 7 (highest).

The **access_priority** parameter is the priority used by the local service provider to convey the request. It can be used to determine the priority attached to the transmission of frames queued by the local MAC Entity, both locally and among other stations attached to the same individual LAN, if the MAC method permits. The value of this parameter, if specified, is in the range 0 (lowest) through 7 (highest).

The **frame_check_sequence** parameter is explicitly provided as a parameter of the primitive so that it can be used without recalculation.

The identification of the LAN to which a frame is to be transmitted is a local matter and is not expressed as a parameter of the service primitive.

6.4.2 MAC status parameters

In addition to the unit-data service primitives, the Internal Sublayer Service makes available a pair of status parameters that permit inspection of, and control over, the administrative and operational state of the MAC entity by the MAC Relay Entity.

MAC_Enabled: The value of this parameter is TRUE if use of the MAC entity is permitted; and is otherwise FALSE. The value of this parameter is determined by the specific MAC procedures, as specified in 6.5.

MAC_Operational: The value of this parameter is TRUE if the MAC entity is in a functioning state and MAC_Enabled is also TRUE; i.e., the MAC entity can be used to transmit and/or receive frames, and its use is permitted by management. Its value is otherwise FALSE. The value of this parameter is determined by the specific MAC procedures, as specified in 6.5.

NOTE—These status parameters provide a common approach across MACs for handling the fact that:

a) A MAC can inherently be working or not (as indicated by MAC_Operational).

b) If the MAC is operational, there may be the need to override its operational state for administrative reasons, preventing any users from making use of its services (by means of MAC_Enabled).

6.4.3 Point-to-Point MAC parameters

In addition to the unit-data service primitives, the Internal Sublayer Service makes available a pair of parameters that permit inspection of, and control over, the administrative and operational state of the point-to-point status of the MAC entity by the MAC Relay Entity.

operPointToPointMAC: This parameter can take two values, as follows:

a) **True.** The MAC is connected to a point-to-point LAN; i.e., there is at most one other system attached to the LAN.

b) **False.** The MAC is connected to a non-point-to-point LAN; i.e., there can be more than one other system attached to the LAN.

adminPointToPointMAC: This parameter can take three values, as follows:

a) **ForceTrue.** The administrator requires the MAC to be treated as if it is connected to a point-to-point LAN, regardless of any indications to the contrary that are generated by the MAC entity.

b) **ForceFalse.** The administrator requires the MAC to be treated as connected to a non-point-to-point LAN, regardless of any indications to the contrary that are generated by the MAC entity.

c) **Auto.** The administrator requires the point-to-point status of the MAC to be determined in accordance with the specific MAC procedures defined in 6.5.

If adminPointToPointMAC is set to ForceTrue, then operPointToPointMAC shall be set True. If adminPointToPointMAC is set to ForceFalse, then operPointToPointMAC shall be set False.

If adminPointToPointMAC is set to Auto, then the value of operPointToPointMAC is determined in accordance with the specific procedures defined for the MAC entity concerned, as defined in 6.5. If these procedures determine that the MAC entity is connected to a point-to-point LAN, then operPointToPointMAC is set TRUE; otherwise it is set FALSE. In the absence of a specific definition of how to determine whether the MAC is connected to a point-to-point LAN or not, the value of operPointToPointMAC shall be FALSE.

The value of operPointToPointMAC is determined dynamically; i.e., it is re-evaluated whenever the value of adminPointToPointMAC changes, and whenever the specific procedures defined for the MAC entity evaluate a change in its point-to-point status.

6.5 Support of the Internal Sublayer Service by specific MAC procedures

This subclause specifies the mapping of the Internal Sublayer Service to the MAC Protocol and Procedures of each individual IEEE 802 MAC type, and the encoding of the parameters of the service in MAC frames. The mapping is specified by reference to the IEEE 802 standards that specify the individual MAC methods. The mapping draws attention to any special responsibilities of Bridges attached to LANs of that MAC type.

6.5.1 Support by IEEE Std 802.3 (CSMA/CD)

The CSMA/CD access method is specified in IEEE Std 802.3. Clause 3 of that standard specifies the MAC frame structure, and Clause 4 specifies the MAC method.

On receipt of an M_UNITDATA.request primitive, the local MAC Entity performs Transmit Data Encapsulation, assembling a frame using the parameters supplied as specified below. It prepends a preamble and a Start Frame Delimiter before handing the frame to the Transmit Media Access Management Component in the MAC Sublayer for transmission (IEEE Std 802.3, 4.2.3).

On receipt of a MAC frame by Receive Media Access Management, the MAC frame is passed to Receive Data Decapsulation, which validates the FCS and disassembles the frame, as specified below, into the parameters that are supplied with an M_UNITDATA.indication primitive (IEEE Std 802.3, 4.2.4).

The **frame_type** parameter takes only the value user_data_frame and is not encoded in MAC frames.

The **destination_address** parameter is encoded in the destination address field (IEEE Std 802.3, 3.2.3).

The **source_address** parameter is encoded in the source address field (IEEE Std 802.3, 3.2.3).

The number of octets of data in the **mac_service_data_unit** parameter is either:

a) Encoded in the Length/Type field of the MAC frame if the frame makes use of the Length interpretation of the Length/Type field (see 3.2.6 in IEEE Std 802.3), or
b) Determined from the length of the received MAC frame, if the frame makes use of the Type interpretation of the Length/Type field (see 3.2.6 in IEEE Std 802.3).

The octets of data are encoded in the data field (see 3.2.7 in IEEE Std 802.3). The Length/Type field forms the initial octets of the mac_service_data_unit parameter.

The **user_priority** parameter provided in a data request primitive is not encoded in MAC frames. The user_priority parameter provided in a data indication primitive shall take the value of the Default User Priority parameter for the Port through which the MAC frame was received. The default value of this parameter is 0, it may be set by management in which case the capability to set it to any of the values 0 through 7 shall be provided.

The **frame_check_sequence** parameter is encoded in the FCS field of the MAC frame (IEEE Std 802.3, 3.2.8). The FCS is computed as a function of the destination address, source address, length, data, and PAD fields. If an M_UNITDATA.request primitive is not accompanied by this parameter, it is calculated in accordance with 3.28 in IEEE Std 802.3.

NOTE 1—Since the PAD field, if present, contributes to the FCS, this parameter needs to include at least the contribution of the PAD field to the FCS in order for the original FCS to be preserved. [See Annex F (informative).]

No special action, above that specified for the support of use of the MAC Service by LLC, is required for the support of the MAC Internal Sublayer Service by the CSMA/CD access method.

NOTE 2—The support by IEEE Std 802.3 is described only in terms of the operation of a Bridge when relaying frames that result from the use of LLC services over an IEEE 802.3 MAC. IEEE Std 802.1H defines the recommended practice for bridging Ethernet V2.0 frames.

The values of the MAC_Enabled and MAC_Operational parameters are determined as follows:

a) For a MAC entity that contains a Link Aggregation sublayer, the value of MAC_Enabled is directly determined by the value of the aAggAdminState attribute (30.7.1.13 in IEEE Std 802.3-2002), and the value of MAC_Operational is directly determined by the value of the aAggOperState attribute (30.7.1.13 in IEEE Std 802.3).
b) Otherwise, for IEEE 802.3 MAC entities that support the MAU managed Object Class (30.5.1 in IEEE Std 802.3):
    1) The value of MAC_Enabled is TRUE.
    2) The value of MAC_Operational is TRUE if the attribute aMediaAvailable carries the value *available*.
    3) The value of MAC_Operational is FALSE if the attribute aMediaAvailable carries any value other than *available*.
c) Otherwise:
    1) The value of MAC_Enabled is TRUE.
    2) The value of MAC_Operational is TRUE.

From the point of view of determining the value of operPointToPointMAC (6.4.3), the MAC is considered to be connected to a point-to-point LAN if any of the following conditions are true:

a) The MAC entity concerned contains a Link Aggregation sublayer, and the set of physical MACs associated with the Aggregator are all aggregatable; or
b) The MAC entity concerned supports auto negotiation (Clause 28 of IEEE Std 802.3), and the auto negotiation function has determined that the LAN is to be operated in full duplex mode; or
c) The MAC entity has been configured by management means for full duplex operation.

Otherwise, the MAC is considered to be connected to a LAN that is not point-to-point.

6.5.2 Support by IEEE Std 802.5 (token-passing ring)

The token-passing ring access method is specified in IEEE Std 802.5. Clause 3 of that standard specifies formats and facilities, and Clause 4 specifies token-passing ring protocols.

On receipt of an M_UNITDATA.request primitive the local MAC Entity composes a frame using the parameters supplied as specified below, appending the frame control, destination address, source address, and FCS fields to the user data, and enqueuing the frame for transmission. On transmission, the starting delimiter, access control field, ending delimiter, and frame status fields are added.

On receipt of a valid MAC frame (IEEE Std 802.5, 4.1.4) that was not transmitted by the Bridge Port's local MAC Entity, with the Routing Information Indicator bit (which occupies the same position in the source address field as does the Group Address bit in the destination address field) set to zero, an M_UNITDATA.indication primitive is generated, with parameters derived from the frame fields as specified in the paragraphs that follow.

The **frame_type** parameter is encoded in the frame_type bits (FF bits) of the frame control field (IEEE Std 802.5, 3.2.3.1). A bit pattern of 0 1 denotes a user_data_frame, a bit pattern of 0 0 denotes a mac_specific_frame, and a bit pattern of 1 0 or 1 1 denotes a reserved_frame.

The **destination_address** parameter is encoded in the destination address field (IEEE Std 802.5, 3.2.4.1).

The **source_address** parameter is encoded in the source address field (IEEE Std 802.5, 3.2.4.2).

The **mac_service_data_unit** parameter is encoded in the information field (IEEE Std 802.5, 3.2.6).

The **user_priority** parameter associated with user_data_frames is encoded in the YYY bits of the frame control field (IEEE Std 802.5, 3.2.3).

The **frame_check_sequence** parameter is encoded in the FCS field of the MAC frame (IEEE Std 802.5, 3.2.7). The FCS is computed as a function of the frame control, destination address, source address, and information fields. If an M_UNITDATA.request primitive is not accompanied by this parameter, it is calculated in accordance with IEEE Std 802.5, 3.2.7.

The Address Recognized (A) bits in the Frame Status field of a frame (IEEE Std 802.5, 3.2.9) may be set to 1 if an M_UNITDATA.indication primitive with a frame_type of user_data_frame is generated, or if such an indication would be generated if buffering had been available; otherwise, the A bits shall not be set except as required by IEEE Std 802.5.

If the A bits are set to 1, the Frame Copied (C) bits (IEEE Std 802.5, 3.2.9) may be set to 1 to reflect the availability of receive buffering; otherwise, the C bits shall not be set.

In order to support the MAC Internal Sublayer Service, a Token Ring Bridge must be capable of recognizing and removing frames transmitted by itself, even though they can carry a source address different from that of the Bridge Port that transmitted them.

The values of the MAC_Enabled and MAC_Operational parameters are determined as follows:

a) For Dedicated Token Ring and High-Speed Token Ring MAC entities:
   1) The value of MAC_Enabled is TRUE.
   2) The value of MAC_Operational is set to TRUE upon invocation of a Mgt_Event_Report.request with an eventRequestType of CPortOperational (see 11.2.2.2 in IEEE Std 802.5).
   3) The value of MAC_Operational is set to FALSE upon invocation of a Mgt_Event_Report.request with an eventRequestType of CPortNonOperational, CPortFailure, or ProtocolError (see 11.2.2.2 in IEEE Std 802.5).
b) For all other IEEE 802.5 MAC entities:
   1) The value of MAC_Enabled is TRUE.
   2) The value of MAC_Operational is set to TRUE upon invocation of a Mgt_Event.indication with an event parameter value of evRingOperational (see 6.1.2 in IEEE Std 802.5).
   3) The value of MAC_Operational is set to FALSE upon invocation of a Mgt_Event.indication with an event parameter value of evRingNonOperational, evRingBeaconing, evStationFailure, or evProtocolError (see 6.1.2 in IEEE Std 802.5).

6.5.3 Support by fibre distributed data interface (FDDI)

The FDDI access method is specified in ISO 9314-2:1989. Clause 6 of that standard specifies Services, and Clauses 7 and 8 specify Facilities and Operation, respectively.

On receipt of a valid frame (ISO 9314-2, 8.3.1) that was not transmitted by the Bridge Port’s local MAC entity, with the first bit of the source address equal to zero, an M_UNITDATA.indication is generated. The associated parameters are derived from the frame fields as specified shortly below.

The Address Recognized (A) indicator in the Frame Status field of the frame (ISO 9314-2, 7.3.8) on the ring from which it was received shall not be set except as required by ISO 9314-2. The Frame Copied (C) indicator (ISO 9314-2, 7.3.8) may be set if an M_UNITDATA.indication primitive with a frame_type of user_data_frame is generated, if the frame is to be forwarded, and if receive buffering is available. Otherwise, the C indicator shall not be altered except as required by ISO 9314-2.

NOTE—This specification of the setting of the C indicator by ISO 9314-2 MAC Bridges enhances that given in ISO 9314-2. A Bridge can be required by ISO 9314-2 to alter the A and/or C indicators when receiving a frame addressed to the Bridge as an FDDI end station, or when receiving a frame associated with the operation of the FDDI MAC.

The parameters associated with the M_UNITDATA.indication generated on receipt of a frame follow:

The **frame_type** parameter is encoded in the frame format bits (CL, FF, and ZZZZ bits) of the Frame Control field (ISO 9314-2, 7.3.3). The bit pattern 0L01rXXX denotes a user_data_frame (asynchronous LLC frame, where L represents the address length and can be 0 or 1, r is reserved and can be received as 0 or 1, and XXX can range from 000 to 111). All other bit patterns yield a frame_type parameter value of not_user_data_frame.

The **destination_address** parameter is enclosed in the destination address field (ISO 9314-2, 7.3.4–7.3.4.1).

The **source_address** parameter is encoded in the source address field (ISO 9314-2, 7.3.4–7.3.4.2).

The **mac_service_data_unit** parameter is encoded in the information field (ISO 9314-2, 7.3.5).

The **user_priority** parameter associated with user_data_frames is encoded in the PPP bits of the frame control field (ISO 9314-2, 7.3.3.4) when the frame is an asynchronously transmitted LLC frame whose frame control field value is 0L010PPP, where L represents the address length (ISO 9314-2, 7.3.3.2).

The **frame_check_sequence** parameter is encoded in the Frame Check Sequence Field of the MAC frame (ISO 9314-2, 7.3.6). The frame_check_sequence is computed as a function of the frame control, destination address, source address, and information fields.

On receipt of an M_UNITDATA.request primitive, the local MAC entity composes a frame using the parameters supplied as specified above, appending the frame control, destination address, source address, and frame check sequence to the user data, and enqueuing the frame for transmission on reception of a suitable token (ISO 9314-2, 8.3.1). On transmission, the preamble, starting delimiter, ending delimiter, and frame status fields are added.

If an M_UNITDATA.request primitive is not accompanied by a frame check sequence, one is calculated in accordance with ISO 9314-2, 7.3.6.

The bit pattern of the frame control field shall be 0L01rPPP, indicating an asynchronous LLC frame with L representing the address length (ISO 9314-2, 7.3.3.2), r being reserved and set to zero, and PPP indicating the frame’s priority (ISO 9314-2, 7.3.3.4).

If the **user_priority** parameter value is specified, it is encoded in the PPP bits and the access priority (ISO 9314-2, 8.1.4) is derived from the token-holding timer (THT) or otherwise by implementor option. If the user_priority parameter value is unspecified, the PPP bits shall be set to zero and the access priority derived from the THT.

In order to support the MAC Internal Layer Service, an FDDI bridge removes frames transmitted by itself as required by ISO 9314-2:1989, even though they can carry a source address different from that of the bridge port that transmitted them.

6.5.4 Support by IEEE Std 802.11 (Wireless LANs)

The wireless LAN access method is specified in IEEE Std 802.11, 1999 Edition. Clause 7 of that standard specifies frame formats, Clause 9 specifies the MAC sublayer function, and Clause 11 specifies the mandatory MAC sublayer management function.

A Bridge to an IEEE 802.11 LAN shall connect to an IEEE 802.11 Portal, which in turn connects to an IEEE 802.11 Distribution System. For the purposes of bridging, the service interface presented at the Portal is identical to the service interface presented at the IEEE 802.11 MAC SAP. An instance of an 8802-11 Distribution System can be implemented from IEEE 802 LAN components. IEEE 802.11 STAs attach to the Distribution System via an IEEE 802.11 Access Point. A bridge shall not connect to an IEEE 802.11 Independent BSS. For a description of the IEEE 802.11 architecture, see Clause 5 of IEEE Std 802.11.

On receipt of an M_UNITDATA.request primitive, the portal constructs a MAC Service Data Unit and passes it to the MAC Data service for transmission (in accordance with the frame formats and procedures specified in IEEE Std 802.11 Clauses 6, 7, 9, and Annex C) using the parameters supplied as specified below.

On receipt of a valid MAC Service Data Unit (see IEEE Std 802.11 Clauses 6, 7, 9, and Annex C), the portal generates an M_UNITDATA.indication primitive with parameter values derived from the frame fields as specified below.

The frame_type parameter only takes the value user_data_frame. When processing MSDU_from_LLC, the frame_type of user_data_frame shall be translated according to parameters specified in 7.1.3.1 of IEEE Std 802.11 and is explicitly encoded in MAC frames.

The destination_address parameter is encoded in MAC frames as the DA described in Table 4 of 7.2.2 of IEEE Std 802.11.

The source_address parameter is encoded in MAC frames as the SA described in Table 4 of 7.2.2 of IEEE Std 802.11.

The mac_service_data_unit parameter is encoded in the Frame Body field (IEEE Std 802.11, 7.1.3.5) of MAC frames. The length of the MSDU shall be ≤ 2304 octets. The length is not encoded in MAC frames; rather, it is conveyed in the PHY headers.

The user_priority parameter is not encoded in MAC frames. The user_priority parameter provided in an M_UNITDATA.indication primitive shall take the value of the Default_User_Priority parameter for the port through which the MAC Service Data Unit was received. The default value of this parameter is 0, it may be set by management, in which case the capability to set it to any of the values 0 through 7 shall be provided.

The Frame Check Sequence (FCS) field of MAC frames is calculated and encoded in accordance with IEEE Std 802.11, 7.1.3.6.

The access_priority parameter is not encoded in MAC frames.

No special action, above that specified in IEEE Std 802.11, is required for the support of the MAC Internal Sublayer Service by the wireless LAN access method.

6.6 Filtering services in Bridged Local Area Networks

MAC Bridges provide filtering services that support aspects of the maintenance of QoS—in particular, transit delay, priority, and throughput. In addition, these services provide a degree of administrative control over the propagation of particular MAC Addresses in the network.

The services described are services in the most general sense; i.e., descriptions of functionality available to a MAC Service user or an administrator to control and access filtering capabilities. The descriptions make no assumptions as to how the service might be realized. There are at least the following possibilities:

a) Use of existing protocols and mechanisms, defined in IEEE 802 standards and elsewhere.
b) Use of management functionality, either locally defined or via remote management protocols.
c) Other means, standardized or otherwise.

6.6.1 Purpose(s) of filtering service provision

Filtering services are provided for the purposes described in 6.6.1.1 and 6.6.1.2.

6.6.1.1 Administrative control

Filtering services provide administrative control over the use of particular source and destination addresses in designated parts of the network. Such control allows network managers and administrators to limit the extent of operation of network layer and other protocols that use individual and group MAC Addresses by establishing administrative boundaries across which specific MAC Addresses are not forwarded.

6.6.1.2 Throughput and end station load

Filtering services increase the overall throughput of the network, and reduce the load placed on end stations caused by the reception of frames that are destined for other end stations, by

a) Limiting frames destined for specific MAC Addresses to parts of the network that, to a high probability, lie along a path between the source MAC Address and the destination MAC Address.
b) Reducing the extent of group addressed frames to those parts of the network that contain end stations that are legitimate recipients of that traffic.

NOTE—Some aspects of the filtering services described in this standard are dependent upon the active participation of end stations. Where such participation is not possible, those aspects of the filtering services will be unavailable.

6.6.2 Goals of filtering service provision

The filtering services provided can be used to

a) Allow the MAC Service provider to dynamically learn where the recipients of frames addressed to individual MAC Addresses are located.
b) Allow end stations that are the potential recipients of MAC frames destined for group MAC Addresses to dynamically indicate to the MAC Service provider which destination MAC Address(es) they wish to receive.
c) Exercise administrative control over the extent of propagation of specific MAC Addresses.

6.6.3 Users of filtering services

The filtering services provided are available to the following users:

a) Network management and administration, for the purposes of applying administrative control. Interactions between administrators of the network and the filtering service provider may be achieved by local means or by means of explicit management mechanisms.
b) End stations, for the purposes of controlling the destination addresses that they will receive. Interactions between end stations and the filtering service provider may be implicit, as is the case with the Learning Process (7.8), or by explicit use of filtering service primitives.

6.6.4 Basis of service

All filtering services in Bridged Local Area Networks rely on the establishment of filtering rules, and subsequent filtering decisions, that are based on the value(s) contained in the Source or Destination MAC Address fields in MAC frames.

NOTE—The filtering services defined by this standard use source address learning and destination address filtering.

6.6.5 Categories of service

Filtering services in Bridged Local Area Networks fall into the following categories:

a) *Basic Filtering Services*. These services are supported by the Forwarding Process (7.7) and by Static Filtering Entries (7.9.1) and Dynamic Filtering Entries (7.9.2) in the Filtering Database. The information contained in the Dynamic Filtering Entries is maintained through the operation of the Learning Process (7.8).
b) *Extended Filtering Services*. These services are supported by the Forwarding Process (7.7), and the Static Filtering Entries (7.9.1) and Group Registration Entries (7.9.3) in the Filtering Database. The information contained in the Group Registration Entries is maintained through the operation of GMRP (10). The categories of Extended Filtering Service are as follows:
    1) Support of dynamic Group forwarding and filtering behavior.
    2) The ability for static filtering information for individual MAC Addresses to specify a subset of Ports for which forwarding or filtering decisions are taken on the basis of dynamic filtering information.

NOTE—Basic Filtering Services as defined in this standard correspond exactly to the filtering capabilities provided by the MAC Bridges standard in IEEE Std 802.1D, 1993 Edition [ISO/IEC 10038: 1993].

All Bridges shall support Basic Filtering Services and may support either or both categories of Extended Filtering Services.

6.6.6 Service configuration

In the absence of explicit information in the Filtering Database, the behavior of the Forwarding Process with respect to the forwarding or filtering of frames destined for group MAC Addresses depends upon the categories of service supported by the Bridge.

Basic Filtering Services support the filtering behavior required for regions of a Bridged Local Area Network in which potential recipients of multicast frames exist, but where either the recipients or the Bridges are either unable to support the dynamic configuration of filtering information for those group MAC Addresses, or the recipients have a requirement to receive all traffic destined for group MAC Addresses.

Extended Filtering Services support the filtering behavior required for regions of a network in which potential recipients of multicast frames exist, and where both the potential recipients of frames and the Bridges are able to support dynamic configuration of filtering information for group MAC Addresses. In order to integrate this extended filtering behavior with the needs of regions of the network that support only Basic Filtering Services, Bridges that support Extended Filtering Services can be statically and dynamically configured to modify their filtering behavior on a per-group MAC Address basis, and also on the basis of the overall filtering service provided by each outbound Port with regard to multicast frames. The latter capability permits configuration of the Port's default forwarding or filtering behavior with regard to group MAC Addresses for which no specific static or dynamic filtering information has been configured.

Service configuration provides the ability to configure the overall filtering for the following cases:

a) Bridges that only implement Basic Filtering Services.
b) Bridges that support Extended Filtering Services in a heterogeneous environment, where some equipment is unable to participate in Dynamic Multicast Filtering, or where some equipment (e.g., routers) have specific needs to see unfiltered traffic.
c) Bridges that support Extended Filtering Services in a homogeneous environment, where all equipment is able to participate in Dynamic Multicast Filtering.

6.6.7 Service definition for Extended Filtering Services

The Filtering Services are described by means of service primitives that define particular types of interaction between MAC Service users and the MAC Service provider across the MAC Service boundary. As these interactions are not defined between peer entities, they are described simply in terms of service requests sent from the MAC Service user to the MAC Service provider.

6.6.7.1 Dynamic registration and de-registration services

These services allow MAC Service users dynamic control over the set of destination Group MAC Addresses that they will receive from the MAC Service provider, by:

a) Registering/de-registering membership of specific Groups associated with those addresses.
b) Registering/de-registering service requirements with regard to the overall forwarding/filtering behavior for Groups.

Provision of these services is achieved by means of GMRP and its associated procedures, as described in Clause 10.

NOTE—The intent of these services is to provide the MAC Service user with dynamic control over access to multicast data streams, for example, multiple video channels made available by a server using a different group MAC Address for each channel. The ability to both register and de-register Group membership, coupled with the filtering action associated with the Group membership, limits the impact of such services on the bandwidth available in the network. These services can be used to control the reception of other categories of multicast traffic, for similar reasons.

REGISTER_GROUP_MEMBER (MAC_ADDRESS)

Indicates to the MAC Service provider that the MAC Service user wishes to receive frames containing the group MAC Address indicated in the MAC_ADDRESS parameter as the destination address. The MAC Addresses that can be carried by this parameter do not include the following:

a) Any individual address.
b) Any of the Reserved Addresses identified in Table 7-10.
c) Any of the GARP Application addresses, as defined in Table 12-1.

DEREGISTER_GROUP_MEMBER (MAC_ADDRESS)

Indicates to the MAC Service provider that the end station no longer wishes to receive frames containing the group MAC Address indicated in the MAC_ADDRESS parameter as the destination address.

REGISTER_SERVICE_REQUIREMENT (REQUIREMENT_SPECIFICATION)

Indicates to the MAC Service provider that the MAC Service user has a requirement for any devices that support Extended Filtering Services to forward frames in the direction of the MAC Service User in accordance with the definition of the service requirement defined by the REQUIREMENT_SPECIFICATION parameter. The values that can be carried by this parameter are as follows:

a) Forward All Groups.
b) Forward Unregistered Groups.

DEREGISTER_SERVICE_REQUIREMENT (REQUIREMENT_SPECIFICATION)

Indicates to the MAC Service provider that the MAC Service user no longer has a requirement for any devices that support Extended Filtering Services to forward frames in the direction of the MAC Service User in accordance with the definition of the service requirement defined by the REQUIREMENT_SPECIFICATION parameter. The values that can be carried by this parameter are as follows:

a) Forward All Groups.
b) Forward Unregistered Groups.

The use of these services can result in the propagation of group MAC Address and service requirement information across the Spanning Tree, affecting the contents of Group Registration Entries (7.9.3) in Bridges and end stations, and thereby affecting the frame forwarding behavior of the Bridges and end stations with regard to multicast frames.

7. Principles of Bridge operation

This clause:

a) Explains the principal elements of Bridge operation and lists the functions that support these.
b) Establishes an architectural model for a Bridge that governs the provision of these functions.
c) Provides a model of Bridge operation in terms of Processes and Entities that support the functions.
d) Details the addressing requirements in a Bridged Local Area Network and specifies the addressing of Entities in a Bridge.

7.1 Bridge operation

The principal elements of Bridge operation are as follows:

a) Relay and filtering of frames.
b) Maintenance of the information required to make frame filtering and relaying decisions.
c) Management of the above.

7.1.1 Relay

A MAC Bridge relays individual MAC user data frames between the separate MACs of the bridged LANs connected to its Ports. The functions that support relaying frames and maintain QoS are as follows:

a) Frame reception.
b) Discard on received frame in error (6.3.2).
c) Frame discard if the frame_type is not user_data_frame (6.4).
d) Regeneration of user priority, if required (6.4).
e) Frame discard to suppress loops in the physical topology of the network.
f) Frame discard to support management control over the physical topology of the network.
g) Frame discard following the application of filtering information.
h) Frame discard on transmittable service data unit size exceeded (6.3.8).
i) Forwarding of received frames to other Bridge Ports.
j) Selection of traffic class, following the application of filtering information.
k) Queuing of frames by traffic class.
l) Frame discard to ensure that a maximum bridge transit delay is not exceeded (6.3.6).
m) Selection of queued frames for transmission.
n) Selection of outbound access priority (6.3.9).
o) Mapping of service data units and recalculation of Frame Check Sequence, if required (6.3.7, 7.7.6).
p) Frame transmission.

7.1.2 Filtering and relaying information

A Bridge filters frames, i.e., does not relay frames received by a Bridge Port to other Ports on that Bridge, in order to prevent frame duplication (6.3.4), and to allow administrative control over network resources. The functions that support the use and maintenance of information for this purpose are as follows:

a) Distributed calculation and configuration of Port State (7.4) for each Bridge Port in the network, to provide a fully, simply, and symmetrically connected spanning tree active topology.
b) Administrative setting of the MAC_Enabled (6.4.2) or Administrative Bridge Port State (14.8.2.2) to exclude a Bridge Port from the active topology.
c) Default or administrative configuration of the Rapid Spanning Tree Protocol parameters (Clause 17) to influence inclusion in the active topology of specific Bridge Ports and the LANs they connect.

A Bridge also filters frames to reduce traffic in parts of the network that do not lie in the path between the source and destination of that traffic. The functions that support the use and maintenance of information for this purpose are as follows:

d) Permanent configuration of reserved addresses.
e) Explicit configuration of static filtering information.
f) Automatic learning of dynamic filtering information for unicast destination addresses through observation of source addresses of network traffic.
g) Ageing out of dynamic filtering information that has been learned.
h) Automatic addition and removal of dynamic filtering information as a result of GMRP exchanges.

A Bridge expedites the transmission of frames generated by critical or time-sensitive services. The function that supports the use and maintenance of information for this purpose is as follows:

i) Explicit configuration of traffic class information associated with the Ports of the Bridge.

7.1.3 Bridge management

The functions that support Bridge Management control and monitor the provision of the above functions are specified in Clause 14.

.

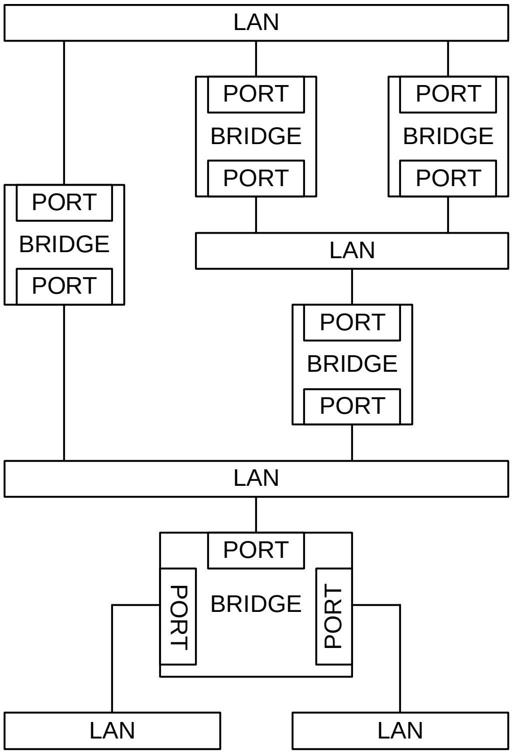


Figure 7-1—A Bridged Local Area Network

7.2 Bridge architecture

A Bridge is modeled as comprising:

a) A MAC Relay Entity that interconnects the Bridge's Ports.
b) At least two Port;.
c) Higher-Layer Entities, including at least a Spanning Tree Protocol Entity.

The MAC Relay Entity handles the Media Access Method Independent Functions of relaying frames between Bridge Ports, filtering frames, and learning filtering information. It uses the Internal Sublayer Service (6.4, 6.5) provided by the separate MAC Entities of each Port.

Each Bridge Port transmits and receives frames to and from the LAN to which it is attached. An individual MAC Entity permanently associated with the Port provides the Internal Sublayer Service (6.4, 6.5) used for frame transmission and reception. The MAC Entity handles all the Media Access Method Dependent Functions (MAC protocol and procedures).

The Spanning Tree Protocol Entity calculates and configures the active topology of the network.

The Spanning Tree Protocol Entity and other higher-layer protocol users, such as Bridge Management (7.1.3) and GARP application entities including GARP Participants (Clause 12), make use of Logical Link Control procedures. These procedures are provided separately for each Port and use the MAC Service provided by the individual MAC Entities.

Figure 7-1 gives an example of the physical topology of a Bridged Local Area Network. LANs are interconnected by MAC Bridges; each Port of a Bridge connects to a single LAN. Figure 7-2 illustrates a Bridge with two Ports, and Figure 7-3 illustrates the architecture of such a Bridge. The term "LLC Entities," used in Figure 7-3 and Figure 7-9, refers to the union of the Link Layer capabilities (which include demultiplexing), provided by LLC (ISO/IEC 8802-2), and the Type interpretation of the Length/Type field specified in IEEE Std 802.3

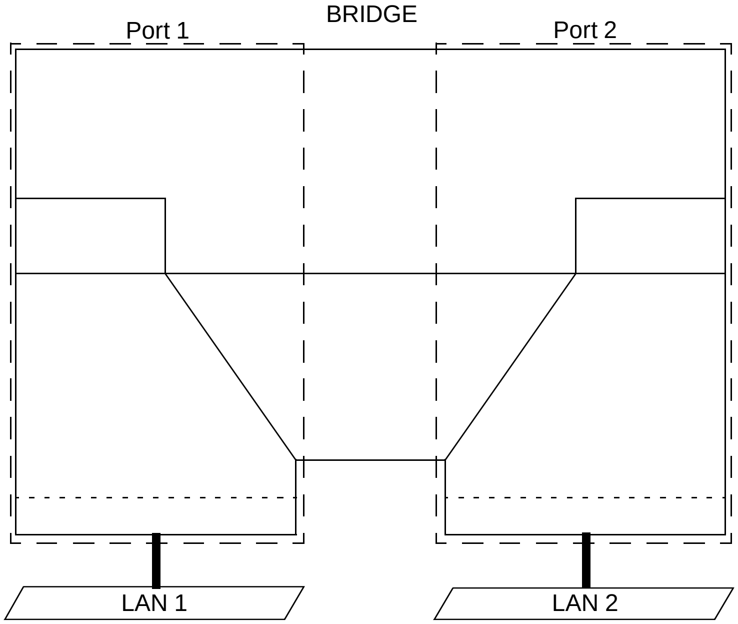


Figure 7-2—Bridge ports

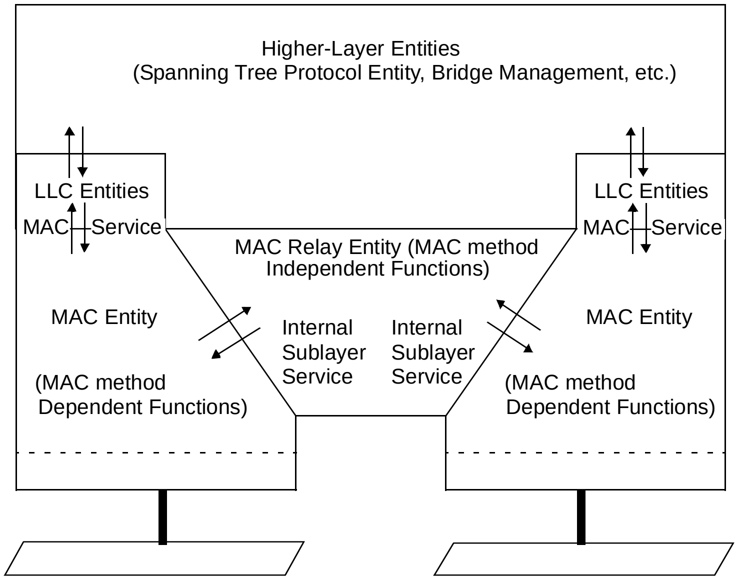

Higher-Layer Entities
(Spanning Tree Protocol Entity, Bridge Management, etc.)
LLC Entities
MAC Service
LLC Entities
MAC Service
MAC Relay Entity (MAC method Independent Functions)
MAC Entity
Internal Sublayer Service
Internal Sublayer Service
MAC Entity
(MAC method Dependent Functions)
(MAC method Dependent Functions)

Figure 7-3—Bridge architecture

7.3 Model of operation

The model of operation is simply a basis for describing the functionality of the MAC Bridge. It is in no way intended to constrain real implementations of a MAC Bridge; these may adopt any internal model of operation compatible with the externally visible behavior that this standard specifies. Conformance of equipment to this standard is purely in respect of observable protocol.

Subclauses 7.5 and 7.6 specify the MAC Relay Entity's use of the Internal Sublayer Service. Port State information (7.4) governs each Port's participation in the Bridged Local Area Network.

Frames are accepted for transmission and delivered on reception to and from Processes and Entities that model the operation of the MAC Relay Entity in a Bridge. These are:

a) The Forwarding Process (7.7), which forwards received frames that are to be relayed to other Bridge Ports, filtering frames on the basis of information contained in the Filtering Database (7.9) and on the state of the Bridge Ports (7.4).
b) The Learning Process (7.8), which by observing the source addresses of frames received on each Port, updates the Filtering Database (7.9), conditionally on the Port state (7.4).
c) The Filtering Database (7.9), which holds filtering information and supports queries by the Forwarding Process as to whether frames with given values of the destination MAC Address field can be forwarded to a given Port.

Each Bridge Port also functions as an end station providing the MAC Service to LLC, which, in turn, supports operation of the Spanning Tree Protocol Entity (7.10) and of other possible users of LLC, such as protocols providing Bridge Management (7.11).

Each Bridge Port shall support the operation of LLC Type 1 procedures in order to support the operation of the Spanning Tree Protocol Entity. Bridge Ports may support other types of LLC procedures, which may be used by other protocols.

Figure 7-4 illustrates a single instance of frame relay between the Ports of a Bridge with two Ports.

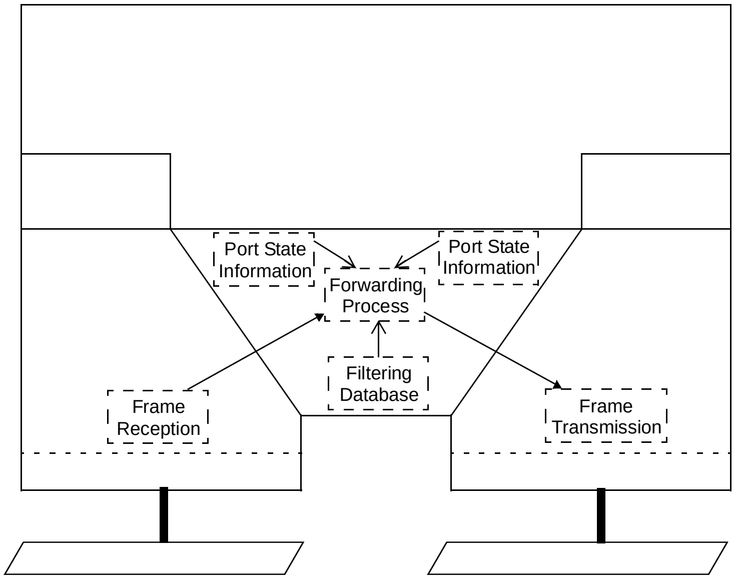


Figure 7-4—Relaying MAC frames

Figure 7-5 illustrates the inclusion of information carried by a single frame, received on one of the Ports of a Bridge with two Ports, in the Filtering Database.

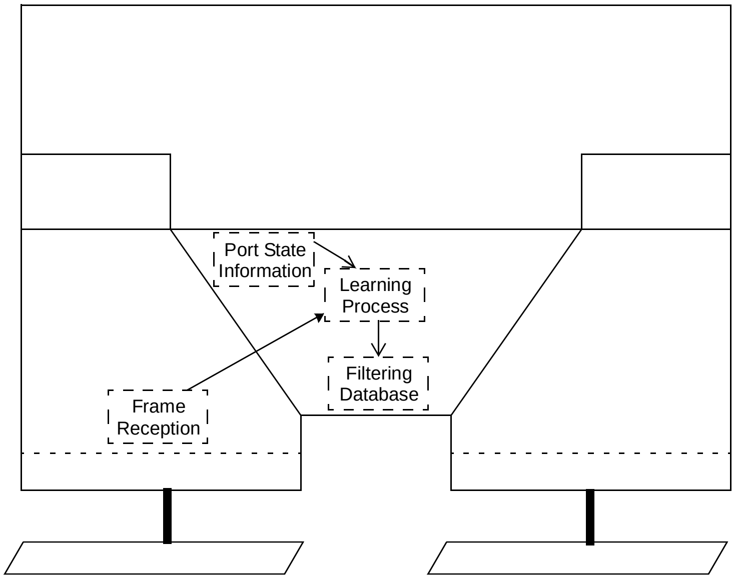


Figure 7-5—Observation of network traffic

Figure 7-6 illustrates the reception and transmission of Bridge Protocol Data Units by the Spanning Tree Protocol Entity.

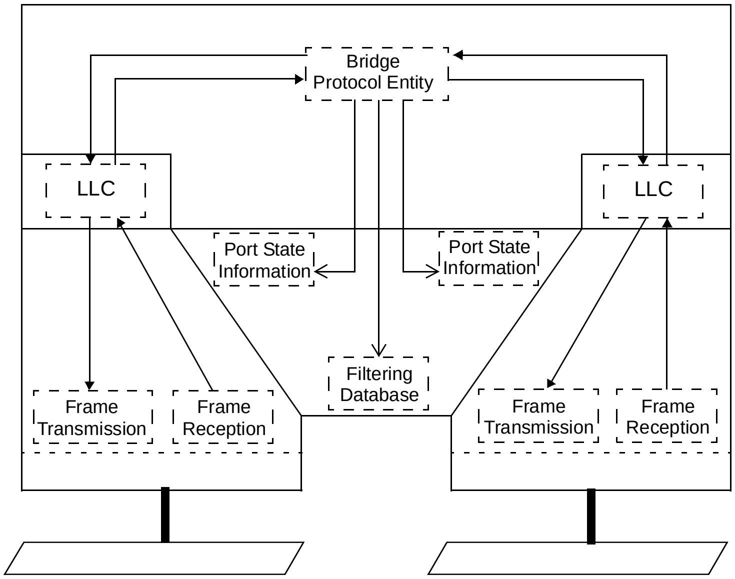


Figure 7-6—Operation of inter-bridge protocol

Figure 7-7 illustrates the reception and transmission of GARP Protocol Data Units by a GARP Entity (7.10).

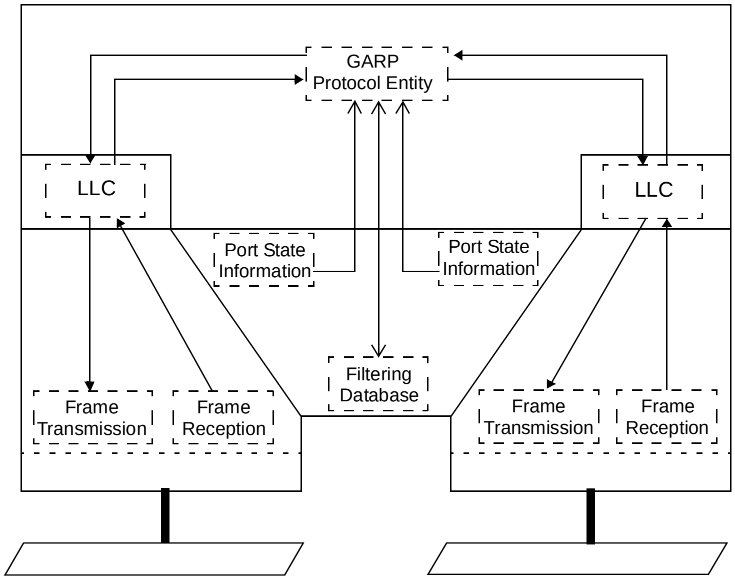


Figure 7-7—Operation of GARP

7.4 Port States and the active topology

Each Bridge Port has an operational Port State that governs whether or not it forwards MAC frames and whether or not it learns from their source addresses.

The *active topology* of a Bridged Local Area Network at any time is the set of communication paths formed by interconnecting the LANs and Bridges by the forwarding Ports. The function of the distributed Spanning Tree algorithm and the protocol (Clause 17) that operates that algorithm is to assign Port States to construct an active topology that is simply connected relative to the forwarding of frames between any given pair of MAC Addresses used to address end stations on the LANs. The *forwarding* and *learning* performed by each Bridge Port is dynamically managed to prevent temporary loops and reduce excessive traffic in the network while minimizing denial of service following any change in the *physical topology* of the network.

Any port that is not enabled, [i.e., has MAC_Operational (6.4.2) False or has been excluded from the active topology by management setting of the Administrative Bridge Port State to Disabled (14.8.2.2)] or has been dynamically excluded from forwarding and learning from MAC frames, is assigned the Port State *Discarding*. Any Port that has learning enabled but forwarding disabled has the Port State *Learning*, and a Port that both learns and forwards frames has the Port State *Forwarding*.

NOTE—The current IETF Bridge MIB (IETF RFC 1493) uses disabled, blocking, listening, learning, forwarding, and broken dot1dStpPortStates. The learning and forwarding states correspond exactly to the Learning and Forwarding Port States specified in this standard. Disabled, blocking, listening, and broken all correspond to the Discarding Port State — while those dot1dStpPortStates serve to distinguish reasons for discarding frames the operation of the Forwarding and Learning processes is the same for all of them. The dot1dStpPortState broken represents the failure or unavailability of the port's MAC as indicated by MAC_Operational FALSE; disabled represents exclusion of the port from the active topology by management setting of the Administrative Port State to Disabled; blocking represents exclusion of the port from the active topology by the spanning tree algorithm [computing an Alternate or Backup Port Role (17.7)]; listening represents a port that the spanning tree algorithm has selected to be part of the active topology (computing a Root Port or Designated Port role) but is temporarily discarding frames to guard against loops or incorrect learning.

Figure 7-6 illustrates the operation of the Spanning Tree Protocol Entity, which operates the Spanning Tree Algorithm and its related protocols, and its modification of Port state information as part of determining the active topology of the Bridged Local Area Network.

Figure 7-4 illustrates the Forwarding Process's use of Port state information: first, for a Port receiving a frame, in order to determine whether the received frame is to be relayed through any other Ports; and second, for another Port in order to determine whether the relayed frame is to be forwarded through that particular Port.

Figure 7-5 illustrates the use of the Port state information for a Port receiving a frame, by the Learning Process, in order to determine whether the station location information is to be incorporated in the Filtering Database.

7.5 Frame reception

The individual MAC Entity of each Bridge Port examines all frames transmitted on the attached LAN.

All error-free received frames give rise to M_UNITDATA indications that are handled as follows.

NOTE—A frame that is in error, as defined by the relevant MAC specification, is discarded by the MAC Entity without giving rise to any M_UNITDATA indication (see 6.4).

Frames with an M_UNITDATA.indication frame_type of user_data_frame (6.4), shall be submitted to the Learning and Forwarding Processes. Frames with other values of frame_type shall not be submitted to the Forwarding Process. They may be submitted to the Learning Process.

Frames with a frame_type of user_data_frame and addressed to the Bridge Port as an end station shall be submitted to LLC. Such frames carry either the individual MAC Address of the Port or a group address associated with the Port (7.12) in the destination address field. Frames submitted to LLC can also be submitted to the Learning and Forwarding Processes, as specified above.

Frames addressed to a Bridge Port as an end station, and relayed to that Bridge Port from other Bridge Ports in the same Bridge by the Forwarding Process, shall also be submitted to LLC. No other frames shall be submitted to LLC.

7.5.1 Regenerating user priority

The user_priority of each received frame is regenerated using the received user_priority and the User Priority Regeneration table for the reception Port. The table specifies the regenerated user_priority value for each of the eight possible values of user_priority (0 through 7) conveyed in received frames. Table 7-1 specifies defaults, these shall be used as the initial values of the entries of the table for each Port.

NOTE 1—IEEE 802 LAN technologies signal a maximum of 8 user_priority values. Annex G (informative) contains further explanation of the use of user_priority values and how they map to traffic classes.

The User Priority Regeneration Table may be modified by management, as described in Clause 14. If so management shall be able to set each of table entries for each reception Port and received user_priority independently, and to any of the values in the ranges specified in Table 7-1.

NOTE 2—The values chosen for the User Priority Regeneration table for a given Bridge Port should be consistent with the user priority to be associated with traffic received through that Port across the rest of the network, and should generate appropriate access priority values for each MAC. The user priority is used

— Via the traffic class table (7.7.3) to determine the traffic class for a given outbound Port, and
— Via fixed, MAC specific mappings (7.7.5) to determine the access priority.

Table 7-1 shows the default values for the regeneration of user priority. Table 7-2 shows the default values for the traffic class table, for all possible numbers of supported traffic classes. Table 7-4 shows the fixed mappings from user priority to access priority that are required for different outbound MAC methods.

Table 7-1—User Priority Regeneration

User Priority	Default Regenerated User Priority	Range
0	0	0–7
1	1	0–7
2	2	0–7
3	3	0–7
4	4	0–7
5	5	0–7
6	6	0–7
7	7	0–7

7.6 Frame transmission

The individual MAC Entity for each Bridge Port transmits frames submitted to it by the MAC Relay Entity.

Relayed frames are submitted for transmission by the Forwarding Process. The M_UNITDATA.request primitive associated with such frames conveys the values of the source and destination address fields received in the corresponding M_UNITDATA.indication primitive.

LLC Protocol Data Units are submitted by LLC as a user of the MAC Service provided by the Bridge Port. Frames transmitted to convey such Protocol Data Units carry the individual MAC Address of the Port in the source address field.

Each frame is transmitted subject to the MAC procedures to be observed for that specific IEEE 802 LAN technology. The frame_type of the corresponding M_UNITDATA.request shall be user_data_frame (6.5).

Frames transmitted following a request by the LLC user of the MAC Service provided by the Bridge Port shall also be submitted to the MAC Relay Entity.

7.7 The Forwarding Process

Frames submitted to the Forwarding Process after being received at any given Bridge Port (7.5) shall be forwarded through the other Bridge Ports subject to the constituent functions of the Forwarding Process. These functions enforce topology restrictions (7.7.1), use filtering database information to filter frames (7.7.2), queue frames (7.7.3), select queued frames for transmission (7.7.4), map priorities (7.7.5), and recalculate FCS, if required (7.7.6).

The Forwarding Process functions are described in 7.7.1–7.7.6 in terms of the action taken for a given frame received on a given Port (termed “the reception Port”). The frame can be forwarded for transmission on some Ports (termed “transmission Ports”), and is discarded without being transmitted at the other Ports.

NOTE—This description of the Forwarding Process is limited to the operation of the relay function of the MAC Bridge, and does not consider what can occur in real implementations once frames are passed to a MAC for transmission. In some MAC implementations, and under some traffic conditions, a degree of indeterminacy can be introduced between passing selected frames to the MAC for transmission and viewing the actual sequence of frames on the LAN medium. An example is the effect of different values for Token Holding Time in FDDI LANs. Such indeterminacy could result in apparent violation of the queuing/de-queueing and prioritization rules described. As a consequence, it is not possible to test conformance to the standard for some implementations simply by relating observed LAN traffic to the described model of the Forwarding Process; conformance tests have to allow for the (permissible) behavior of the MAC implementations as well.

Figure 7-4 illustrates the operation of the Forwarding Process in a single instance of frame relay between the Ports of a Bridge with two Ports. Figure 7-8 illustrates the detailed operation of the Forwarding Process.

.

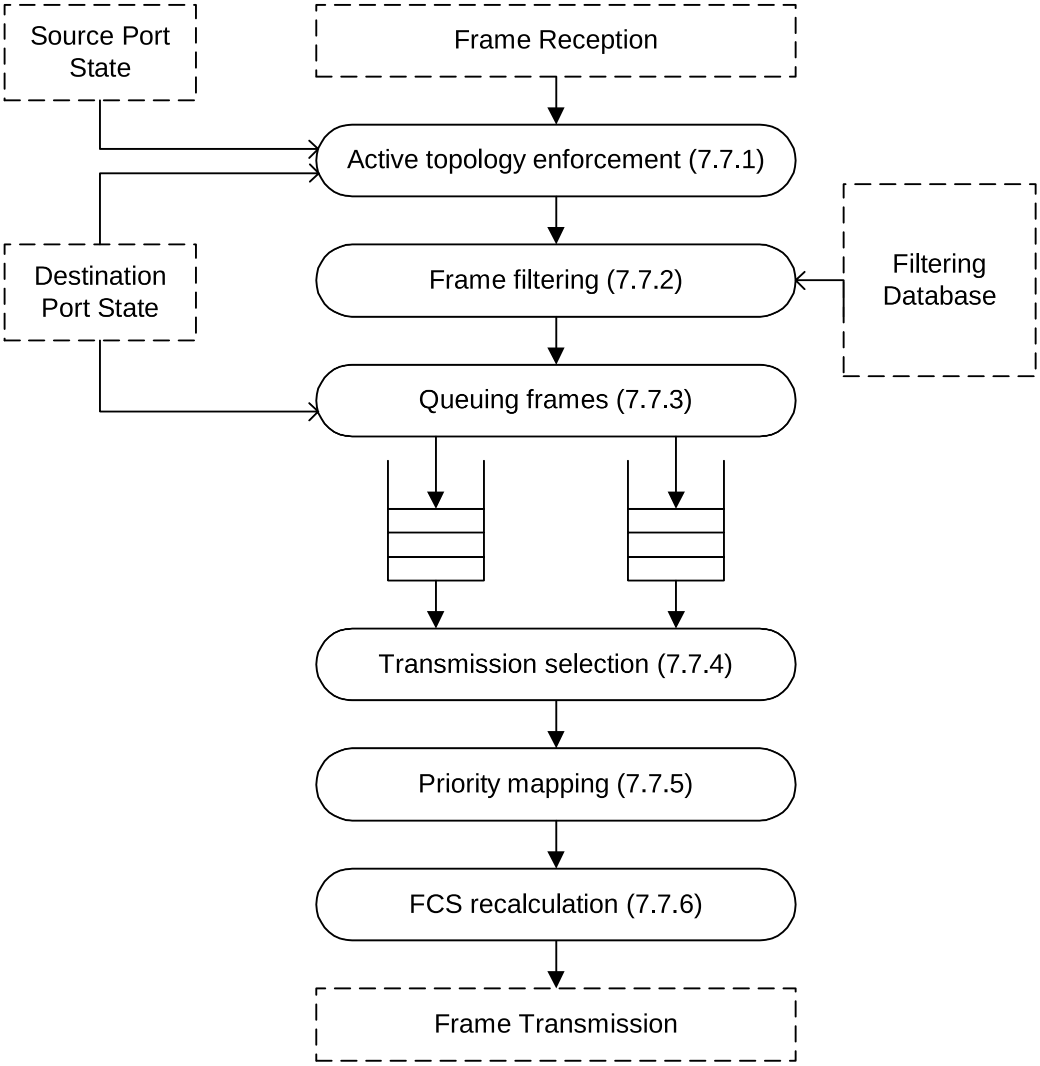


Figure 7-8—Operation of the Forwarding Process

7.7.1 Active topology enforcement

Each Port is selected as a potential transmission Port if, and only if

a) The Port on which the frame was received was in the Forwarding State (7.4), and

b) The Port considered for transmission is in the Forwarding State, and

c) The Port considered for transmission is not the Port on which the frame was received, and

d) The size of the mac_service_data_unit conveyed by the frame does not exceed the maximum size of mac_service_data_unit supported by the LAN attached to the Port considered for transmission.

For each Port not selected as a potential transmission Port, the frame shall be discarded.

7.7.2 Frame filtering

Filtering decisions are taken by the Forwarding Process on the basis of

a) The destination MAC Address carried in a received frame.
b) The information contained in the Filtering Database for that MAC Address and reception Port.
c) The default Group filtering behavior for the potential transmission Port (7.9.4).

For each potential transmission Port (7.7.1), the frame shall be forwarded, or discarded (i.e., filtered) in accordance with the definition of the Filtering Database entry types (7.9.1, 7.9.2, and 7.9.3). The required forwarding and filtering behavior is summarized in 7.9.4, 7.9.5, Table 7-6, Table 7-7, and Table 7-8.

7.7.3 Queuing frames

The Forwarding Process provides storage for queued frames, awaiting an opportunity to submit these to the individual MAC Entity for each Bridge Port for transmission. The order of frames received on the same Bridge Port shall be preserved for:

a) Unicast frames with a given user_priority and destination_address and source_address combination.
b) Multicast frames with a given user_priority and destination_address.

The Forwarding Process may provide more than one transmission queue for a given Bridge Port. Frames are assigned to queue(s) on the basis of their user_priority (7.5.1) using a traffic class table that is part of the state information associated with each Port. Queues correspond one-to-one with traffic classes.

Up to eight traffic classes are supported by the traffic class tables, to allow for separate queues for each level of user_priority. Traffic classes are numbered 0 through N-1, where N is the number of traffic classes for a given outbound Port. Traffic class tables may be managed. Traffic class 0 corresponds to nonexpedited traffic; nonzero traffic classes are expedited classes of traffic.

NOTE—In a given Bridge, different numbers of traffic classes may be implemented for different Ports. Ports with MACs that support a single transmission priority, such as CSMA/CD, can support more than one traffic class.

Where the Forwarding Process does not support expedited classes of traffic for a given Port, i.e., where there is a single traffic class for the Port, all values of user_priority map to traffic class 0. In bridges that support expedited traffic, the recommended mapping of user_priority for the number of traffic classes implemented, is shown in Table 7-2. Each entry in the table is the traffic class assigned to frames of a given user_priority.

A frame queued by the Forwarding Process for transmission on a Port shall be removed from that queue on submission to the individual MAC Entity for that Port. No further attempt shall be made to transmit the frame on that Port even if the transmission is known to have failed.

A frame queued by the Forwarding Process for transmission on a Port can be removed from that queue, and not subsequently transmitted, if the time for which buffering is guaranteed has been exceeded for that frame.

A frame queued for transmission on a Port shall be removed from that queue if that is necessary to ensure that the maximum bridge transit delay (6.3.6, Table 7-3) will not be exceeded at the time at which the frame would subsequently be transmitted.

A frame queued for transmission on a Port shall be removed from that queue if the associated Port leaves the Forwarding State. Removal of a frame from a queue for any particular Port does not of itself imply that it is to be removed from a queue for transmission on any other Port.

Table 7-2—Recommended user priority to traffic class mappings

		Number of available traffic classes							
		1	2	3	4	5	6	7	8
User Priority	0 (Default)	0	0	0	1	1	1	1	2
	1	0	0	0	0	0	0	0	0
	2	0	0	0	0	0	0	0	1
	3	0	0	0	1	1	2	2	3
	4	0	1	1	2	2	3	3	4
	5	0	1	1	2	3	4	4	5
	6	0	1	2	3	4	5	5	6
	7	0	1	2	3	4	5	6	7

NOTE—The rationale for these mappings is discussed in Annex G (informative). Frames with default user priority are given preferential treatment over user priority 1 and 2 in Bridges that implement four or more Traffic Classes.

Table 7-3—Bridge transit delay

Parameter	Recommended value	Absolute maximum
Maximum bridge transit delay	1.0 second	4.0 seconds

7.7.4 Transmission selection

The following shall be supported by as the default algorithm for selecting frames for transmission:

a) For each Port, frames are selected for transmission on the basis of the traffic classes that the Port supports. For a given supported value of traffic class, frames are selected from the corresponding queue for transmission only if all queues corresponding to numerically higher values of traffic class supported by the Port are empty at the time of selection.

b) For a given queue, the order in which frames are selected for transmission shall maintain the ordering requirement specified in 7.7.3.

Additional algorithms that meet the requirements of 7.7.3 may be selected by management.

7.7.5 Priority mapping

The user_priority parameter in an M_UNITDATA.request primitive (6.4) shall be equal to the user_priority parameter in the corresponding data indication.

The mapping of user_priority to outbound access_priority is achieved via fixed, MAC-specific mappings. The access_priority parameter in an M_UNITDATA.request primitive (6.4) shall be determined from the user_priority in accordance Table 7-4. The values shown shall not be changed.

Table 7-4—Outbound access priorities

	Outbound Access Priority per MAC type			
user_priority	**IEEE 802.3**	**IEEE 802.5**	**IEEE 802.11**	**FDDI**
0	0	0	0	0
1	0	1	0	1
2	0	2	0	2
3	0	3	0	3
4	0	4	0	4
5	0	5	0	5
6	0	6	0	6
7	0	6	0	6

7.7.6 FCS recalculation

When a frame is forwarded between two MAC Entities of the same IEEE 802 LAN type, and the data that is within the FCS coverage is not modified, the FCS received in the M_UNITDATA.indication primitive may be supplied in the corresponding M_UNITDATA.request primitive and not recalculated (6.3.7). For frames relayed between LANs of the same MAC type, the Bridge shall not introduce an undetected frame error rate greater than that which would have been achieved by preserving the FCS.

When a frame is forwarded between two MAC Entities of different types, the FCS is recalculated according to the procedures of the transmitting MAC entity if they differ from the FCS calculation procedures of the receiving MAC or the data within the FCS coverage is changed.

NOTE—There are two possibilities for recreating a valid FCS. The first is to generate a new FCS by algorithmically modifying the received FCS, based on knowledge of the FCS algorithm and the transformations that the frame has undergone between reception and transmission. The second is to rely on the normal MAC procedures to recalculate the FCS for the outgoing frame. The former approach can protect against increased levels of undetected frame errors. Annex F (informative) discusses these possibilities in more detail. The frame_check_sequence parameter of the Internal Sublayer Service (6.4) signals the validity, or otherwise, of the FCS; an unspecified value in this parameter in a data request indicates to the transmitting MAC that the FCS is to be recalculated.

7.8 The Learning Process

The Learning Process observes the source addresses of frames received on each Port and updates the Filtering Database conditionally on the state of the receiving Port.

The Learning Process shall create or update a Dynamic Filtering Entry (7.9, 7.9.2) in the Filtering Database, associating the MAC Address in the source address field of the frame with the receiving Port, if and only if

a) The receiving Port is in the Learning State or the Forwarding State (7.4), and
b) The source address field of the frame denotes a specific end station (i.e., is not a group address), and
c) No Static Filtering Entry (7.9, 7.9.1) for the associated MAC Address exists in which the Port Map specifies Forwarding or Filtering for that Port, and

d) The resulting number of entries would not exceed the capacity of the Filtering Database.

If the Filtering Database is already filled to capacity, but a new entry would otherwise be made, then an existing entry may be removed to make room for the new entry.

Figure 7-5 illustrates the operation of the Learning Process in the inclusion of station location information carried by a single frame, received on one of the Ports of a Bridge, in the Filtering Database.

7.9 The Filtering Database

The Filtering Database supports queries by the Forwarding Process, as to whether a frame received by a given Port with a given destination MAC Address is to be forwarded through a given potential transmission Port (7.7.1, 7.7.2). It contains filtering information that is either

a) Static, and explicitly configured by management action; or
b) Dynamic, and automatically entered into the Filtering Database by the normal operation of the Bridge and the protocols it supports.

A single entry type, the Static Filtering Entry, represents all static information in the Filtering Database, for individual and for group MAC Addresses. It allows administrative control of

c) Forwarding of frames with particular destination addresses; and
d) The inclusion in the Filtering Database of dynamic filtering information associated with Extended Filtering Services, and use of this information.

The Filtering Database shall contain entries of the Static Filtering Entry type.

Static filtering information is added to, modified, and removed from the Filtering Database only under explicit management control. It shall not be automatically removed by any ageing mechanism. Static filtering information may be managed using the operations specified in Clause 14.

Two entry types are used to represent dynamic filtering information. Dynamic Filtering Entries are used to specify the ports on which individual addresses have been learned. They are created and updated by the Learning Process (7.8), and are subject to ageing and removal by the Filtering Database. Group Registration Entries support the registration of group MAC Addresses. They are created, updated, and removed by the GMRP protocol in support of Extended Filtering Services (6.6.5, 7.9.3, Clause 10), subject to the state of the *Restricted_Group_Registration* management control (10.3.2.3). If this control is TRUE, the creation of a Group Registration Entry is not permitted unless a Static Filtering Entry exists that permits dynamic registration for the Group concerned. Dynamic filtering information may be read using the remote management capability provided by Bridge Management (7.11) and the operations specified in Clause 14.

Both static and dynamic entries comprise

e) A MAC Address specification.
f) A Port Map, with a control element for each outbound Port for the MAC Address specification.

The Filtering Services supported by a Bridge (Basic and Extended Filtering Services) determine the default behavior of the Bridge with respect to the forwarding of frames destined for group MAC Addresses. In Bridges that support Extended Filtering Services, the default forwarding behavior of each Port for group MAC Addresses can be configured both statically and dynamically by means of Static Filtering Entries and/or Group Registration Entries that can carry the following MAC Address specifications:

g) All Group Addresses, for which no more specific Static Filtering Entry exists.

h) All Unregistered Group Addresses (i.e., all group MAC Addresses for which no Group Registration Entry exists), for which no more specific Static Filtering Entry exists.

NOTE—The All Group Addresses specification [item g)], when used in a Static Filtering Entry with an appropriate control specification, provides the ability to configure a Bridge that supports Extended Filtering Services to behave as a Bridge that supports only Basic Filtering Services on some or all of its Ports. This might be done for the following reasons:
— The Ports serve “legacy” devices that wish to receive multicast traffic, but are unable to register Group membership.
— The Ports serve devices that need to receive all multicast traffic, such as routers or diagnostic devices.

The Filtering Database shall support the creation, updating, and removal of Dynamic Filtering Entries by the Learning Process (7.8). In Bridges that support Extended Filtering Services, the Filtering Database shall support the creation, updating, and removal of Group Registration Entries by GMRP (Clause 10).

Figure 7-4 illustrates the use of the Filtering Database by the Forwarding Process in a single instance of frame relay between the Ports of a Bridge with two Ports.

Figure 7-5 illustrates the creation or update of a dynamic entry in the Filtering Database by the Learning Process.

Figure 7-6 illustrates the operation of the Spanning Tree Protocol Entity (7.10), and its notification of the Filtering Database of changes in active topology signaled by a spanning tree protocol.

7.9.1 Static Filtering Entries

A Static Filtering Entry specifies

a) A MAC Address specification, comprising
   1) An Individual MAC Address; or
   2) A Group MAC Address; or
   3) All Group Addresses, for which no more specific Static Filtering Entry exists; or
   4) All Unregistered Group Addresses, for which no more specific Static Filtering Entry exists.
b) A Port Map, containing a control element for each outbound Port, specifying that a frame with a destination MAC Address that meets this specification is to be
   1) Forwarded, independently of any dynamic filtering information held by the Filtering Database; or
   2) Filtered, independently of any dynamic filtering information; or
   3) Forwarded or filtered on the basis of dynamic filtering information, or on the basis of the default Group filtering behavior for the outbound Port (7.9.4) if no dynamic filtering information is present specifically for the MAC Address.

All Bridges shall have the capability to support the first two values for the MAC Address specification, and the first two values for each control element for all Static Filtering Entries [i.e., shall have the capability to support items a1), a2), b1), and b2) above].

A Bridge that supports Extended Filtering Services shall have the capability to support all four values for the MAC Address specification and all three control element values for Static Filtering Entries that specify group MAC Addresses, and may have the capability to support all three control element values for Static Filtering Entries that specify individual MAC Addresses [i.e., shall have the capability to support items a1) through a4), and may have the capability to support item b3), in addition to support of items b1) and b2)].

For a given MAC Address specification, a separate Static Filtering Entry with a distinct Port Map may be created for each inbound Port from which frames are received by the Forwarding Process.

In addition to controlling the forwarding of frames, Static Filtering Entries for group MAC Addresses provide the Registrar Administrative Control values for the GMRP protocol (Clause 10, Clause 12, 12.8.1). Static configuration of forwarding of specific group addressed frames to an outbound port indicates Registration Fixed on that port—a desire to receive frames addressed to that Group even in the absence of dynamic information. Static configuration of filtering of frames that might otherwise be sent to an outbound port indicates Registration Forbidden. The absence of a Static Filtering Entry for the group address, or the configuration of forwarding or filtering on the basis of dynamic filtering information, indicates Normal Registration.

NOTE—The possibility of configuring a number of Static Filtering Entries, each for a different inbound port or ports, can appear to complicate registration controls. Group registration is propagated across the Bridge to the inbound Port, and that Port acts as a GMRP Applicant if any part of the propagated information indicates a desire to receive frames for the Group. Such frames will be filtered from Ports that do not wish to receive them, and correct operation is maintained.

7.9.2 Dynamic Filtering Entries

A Dynamic Filtering Entry specifies

a) An individual MAC Address.
b) A Port Map consisting of a control element that specifies forwarding of frames destined for that MAC Address to a single Port.

NOTE 1—This is equivalent to specifying a single port number; hence, this specification is directly equivalent to the specification of dynamic entries in IEEE Std 802.1D, 1993 Edition.

Dynamic Filtering Entries are created and updated by the Learning Process (7.8). They shall be automatically removed after a specified time, the Ageing Time (Table 7-5), has elapsed since the entry was created or last updated. No more than one Dynamic Filtering Entry shall be created in the Filtering Database for a given MAC Address.

A Dynamic Filtering Entry shall not be created or updated by the Learning Process if any Static Filtering Entry already exists for this MAC Address with a control element specification, for the outbound Port specified by the Learning Process, that specifies forwarding or filtering irrespective of dynamic filtering information.

NOTE 2—For Bridges that do not permit the (optional) ability of Static Filtering Entries to specify forwarding or filtering on the basis of dynamic filtering information (see 7.9.1), including Bridges that conform to IEEE Std 802.1D, 1993 Edition, this effectively prevents the creation of Dynamic Filtering Entries where a Static Filtering Entry exists for the same MAC Address. This in turn ensures that these Bridges continue to conform to their own specification, which prohibits creation of a Dynamic Filtering Entry if a Static Filtering Entry already exists.

For Bridges that do permit the ability of Static Filtering Entries to specify forwarding or filtering on the basis of dynamic filtering information, it is possible for Dynamic and Static Filtering Entries to exist for the same MAC Address, as long as the address is not learned on a Port for which there is a Static Filtering Entry that specifies “forwarding or filtering independently of any dynamic filtering information.”

The facility provided by this updated specification allows source address learning to be confined to a subset of Ports.

Dynamic Filtering Entries cannot be created or updated by management.

If a Dynamic Filtering Entry exists for a given MAC Address, creation or updating of a Static Filtering Entry for the same address causes removal of any conflicting information that may be contained in the Dynamic Filtering Entry. If removal of such conflicting information would result in a Port Map that does not specify Forwarding on any Port, then that Dynamic Filtering Entry is removed from the Filtering Database.

The ageing out of Dynamic Filtering Entries ensures that end stations that have been moved to a different part of the network are not permanently prevented from receiving frames. It also takes account of changes in the active topology of the Bridged Local Area Network that can cause end stations to appear to move from the point of view of the bridge; i.e., the path to those end stations subsequently lies through a different Bridge Port.

The Ageing Time may be set by management (Clause 14). A range of applicable values and a recommended default is specified in Table 7-5; the default value is suggested to remove the need for explicit configuration in most cases. If the value of Ageing Time can be set by management, the Bridge shall have the capability to use values in the range specified with a granularity of 1 s.

Table 7-5—Ageing Time parameter value

Parameter	Recommended default value	Range
Ageing Time	300.0 s	10.0–1 000 000.0 s

NOTE 3—The granularity is specified here in order to establish a common basis for the granularity expressed in the management operations defined in Clause 14, not to constrain the granularity of the actual timer supported by a conformant implementation. If the implementation supports a granularity other than 1 second, then it is possible that the value read back by management following a Set operation will not match the actual value expressed in the Set.

The Rapid Spanning Tree Algorithm and Protocol specified in Clause 17 includes a procedure for notifying all Bridges in the Bridged Local Area Network of a change in the active topology, so that Dynamic Filtering Entries can be removed from the Filtering Database. While the topology is not changing, this procedure allows normal ageing to accommodate extended periods during which addressed end stations do not generate frames themselves, perhaps through being powered down, without sacrificing the ability of the Bridged Local Area Network to continue to provide service after automatic configuration.

7.9.3 Group Registration Entries

A Group Registration Entry specifies

a) A MAC Address specification, comprising
   1) A Group MAC Address; or
   2) All Group Addresses, for which no more specific Static Filtering Entry exists; or
   3) All Unregistered Group Addresses, for which no more specific Static Filtering Entry exists.

b) A Port Map consisting of a control element for each outbound Port that specifies forwarding or filtering of frames destined to the MAC Address.

Group Registration Entries are created, modified, and deleted by the operation of GMRP (Clause 10). No more than one Group Registration Entry shall be created in the Filtering Database for a given MAC Address specification.

NOTE—It is possible to have a Static Filtering Entry that has values of Forward or Filter on some or all Ports that mask the dynamic values held in a corresponding Group Registration Entry. The values in the Group Registration Entry will continue to be updated by GMRP; hence, subsequent modification of that entry to allow the use of dynamic filtering information on one or more Ports immediately activates the true GMRP registration state that was hitherto masked by the static information.

7.9.4 Default Group filtering behavior

Forwarding and filtering of group addressed frames on each outbound Port by a Bridge that supports Extended Filtering Services, can be managed by creating a Static Filtering Entry that specifies a default for All Group Addresses and or a Static Filtering Entry that specifies a default for All Unregistered Group Addresses (7.9.1). The behavior of each of these defaults, as modified by more explicit Filtering Database entries applicable to a given frame's MAC Address, reception Port, and outbound Port, is as follows:

NOTE 1—As stated in 7.9.1, a Bridge may support creation of separate Static Filtering Entries with a distinct Port Map for each reception Port. If this capability is not provided, then a given Static Filtering Entry applies to all reception Ports.

a) *Forward All Groups*. The frame is forwarded, unless an explicit Static Filtering Entry specifies filtering independent of any dynamic filtering information.
b) *Forward Unregistered Groups*. The frame is forwarded, unless
    1) An explicit Static Filtering Entry specifies filtering independent of any dynamic filtering information; or
    2) An explicit Static Filtering Entry specifies forwarding or filtering on the basis of dynamic filtering information, and an applicable explicit Group Registration Entry exists specifying filtering; or
    3) An applicable explicit Static Filtering Entry does not exist, but an applicable Group Registration entry specifies filtering.
c) *Filter Unregistered Groups*. The frame is filtered unless
    1) An explicit Static Filtering Entry specifies forwarding independent of any dynamic filtering information; or
    2) An explicit Static Filtering Entry specifies forwarding or filtering on the basis of dynamic filtering information, and an applicable explicit Group Registration Entry exists specifying forwarding; or
    3) An applicable explicit Static Filtering Entry does not exist, but an applicable Group Registration entry specifies forwarding.

In Bridges that support only Basic Filtering Services, the default Group filtering behavior is Forward All Groups for all Ports of the Bridge.

NOTE 2—Forward All Groups corresponds directly to the behavior specified in IEEE Std 802.1D, 1993 Edition when forwarding group MAC Addressed frames for which no static filtering information exists in the Filtering Database. Forward All Groups makes use of information contained in Static Filtering Entries for specific group MAC Addresses, but overrides any information contained in Group Registration Entries. Forward Unregistered Groups is analogous to the forwarding behavior of a Bridge with respect to individual MAC Addresses; if there is no static or dynamic information for a specific group MAC Address, then the frame is forwarded; otherwise, the frame is forwarded in accordance with the statically configured or dynamically learned information.

NOTE 3—The result is that the default Group filtering behavior can be configured for each Port of the Bridge via Static Filtering Entries, which are determined dynamically via Group Registration Entries created/updated by GMRP (Clause 10), or both. For example, in the absence of any static or dynamic information in the Filtering Database for All Group Addresses or All Unregistered Group Addresses, the default Group filtering behavior will be Filter Unregistered Groups on all Ports. Subsequently, the creation of a Dynamic Group Registration Entry for All Unregistered Group Addresses indicating "Registered" on a given Port would cause that Port to exhibit Forward Unregistered Groups behavior. Similarly, creating a Static Filtering Entry for All Group Addresses indicating "Registration Fixed" on a given Port would cause that Port to exhibit Forward All Groups behavior.
Hence, by using appropriate combinations of "Registration Fixed," "Registration Forbidden," and "Normal Registration" in the Port Maps of Static Filtering Entries for the All Group Addresses and All Unregistered Group Addresses address specifications, it is possible, for a given Port, to
— Fix the default Group filtering behavior to be just one of the three behaviors described above; or
— Restrict the choice of behaviors to a subset, and allow GMRP registrations to determine the final choice; or
— Allow any one of the three behaviors to be adopted, in accordance with any registrations received via GMRP.

7.9.5 Querying the Filtering Database

Each entry in the Filtering Database comprises:

a) A MAC Address specification;
b) A Port Map, with a control element for each outbound Port.

A given individual MAC Address can be in a Static Filtering Entry, a Dynamic Filtering Entry, both, or neither. Table 7-6 combines Static Filtering Entry and Dynamic Filtering Entry information to specify forwarding, or filtering, of a frame with an individual destination MAC Address through an outbound Port.

Table 7-6—Combining Static and Dynamic Filtering Entries for an individual MAC Address

Filtering Information	Static Filtering Entry Control Element for this individual MAC Address and Port specifies:				
	Forward	Filter	Use Dynamic Filtering Information, or no Static Filtering Entry present. Dynamic Filtering Entry Control Element for this individual MAC Address and Port specifies:		
			Forward	Filter	No Dynamic Filtering Entry present
Result	Forward	Filter	Forward	Filter	Forward

Table 7-7 specifies the result, Registered or Not Registered, of combining a Static Filtering Entry and a Group Registration Entry for the “All Group Addresses” address specification, and for the “All Unregistered Group Addresses” address specification.

Table 7-7—Combining Static Filtering and Group Registration Entries

Filtering Information	Static Filtering Entry Control Element for this address specification and Port specifies:				
	Registration Fixed (Forward)	Registration Forbidden (Filter)	Use Group Registration Information, or no Static Filtering Entry present. Group Registration Entry Control Element for this address specification and Port specifies:		
			Registered (Forward)	Not Registered (Filter)	No Group Registration Entry present
Result	Registered	Not Registered	Registered	Not Registered	Not Registered

Table 7-8 combines Static Filtering Entry and Group Registration Entry information for a specific group MAC Address with the Table 7-7 results for All Group Addresses and All Unregistered Group Addresses to specify forwarding, or filtering, of a frame with that destination group MAC Address through an outbound Port.

Table 7-8—Forwarding or Filtering for specific group MAC Addresses

				Static Filtering Entry Control Element for this group MAC Address and Port specifies:				
				Registration Fixed (Forward)	Registration Forbidden (Filter)	Use Group Registration Information, or no Static Filtering Entry present. Group Registration Entry Control Element for this group MAC Address and Port specifies:		
						Registered (Forward)	Not Registered (Filter)	No Group Registration Entry present
All Group Addresses control elements for this Port specify (Table 7-7):	Not Registered	All Unregistered Group Addresses control elements for this Port specify (Table 7-7):	Not Registered	Forward	Filter	Forward	Filter	Filter (Filter Unregistered Groups)
			Registered	Forward	Filter	Forward	Filter	Forward (Forward Unregistered Groups)
	Registered			Forward	Filter	Forward (Forward All Groups)	Forward (Forward All Groups)	Forward (Forward All Groups)

7.9.6 Permanent Database

The Permanent Database provides fixed storage for a number of Static Filtering Entries. The Filtering Database shall be initialized with the Filtering Database Entries contained in this fixed data store.

Entries can be added to and removed from the Permanent Database under explicit management control, using the management functionality defined in Clause 14. Changes to the contents of Static Filtering Entries in the Permanent Database do not affect forwarding and filtering decisions taken by the Forwarding Process until such a time as the Filtering Database is reinitialized.

NOTE 1—This aspect of the Permanent Database can be viewed as providing a “boot image” for the Filtering Database, defining the contents of all initial entries, before any dynamic filtering information is added.

NOTE 2—10.3.2.3 defines an initial state for the contents of the Permanent Database, required for the purposes of GMRP operation.

7.10 Spanning Tree Protocol Entity and GARP Entities

The Spanning Tree Protocol Entity operates the Rapid Spanning Tree Protocol. The Spanning Tree Protocol Entities of Bridges attached to a given individual LAN in a Bridged Local Area Network communicate by exchanging Bridge Protocol Data Units (BPDUs).

Figure 7-6 illustrates the operation of the Spanning Tree Protocol Entity including the reception and transmission of frames containing BPDUs, the modification of the state information associated with individual Bridge Ports, and notification of the Filtering Database of changes in active topology.

The GARP Entities operate the Algorithms and Protocols of the GARP Applications supported by the Bridge, and consist of the set of GARP Participants for those GARP Applications (12.2, Clause 10). The GARP Entities of Bridges attached to a given individual LAN communicate by exchanging GARP Protocol Data Units (GARP PDUs).

Figure 7-7 illustrates the operation of a GARP Entity including the reception and transmission of frames containing GARP PDUs, the use of control information contained in the Filtering Database, and notification of the Filtering Database of changes in filtering information.

7.11 Bridge management

Remote management facilities may be provided by the Bridge, and are modeled as being performed by the Bridge Management Entity. The facilities provided and supporting operations are specified in Clause 14. Bridge Management protocols use the Service provided by the operation of LLC Procedures, which use the MAC Service provided by the Bridged Local Area Network.

7.12 Addressing

All MAC Entities communicating across a Bridged Local Area Network use 48-bit addresses. These can be Universally Administered Addresses or a combination of Universally Administered and Locally Administered Addresses.

7.12.1 End stations

Frames transmitted between end stations using the MAC Service provided by a Bridged Local Area Network carry the MAC Address of the source and destination peer end stations in the source and destination address fields of the frames, respectively. The address, or other means of identification, of a Bridge is not carried in frames transmitted between peer users for the purpose of frame relay in the network.

The broadcast address and other group MAC Addresses apply to the use of the MAC Service provided by a Bridged Local Area Network as a whole. In the absence of explicit filters configured via management as Static Filtering Entries, or via GMRP as Group Registration Entries (Clause 14, Clause 10, 7.9), frames with such destination addresses are relayed throughout the network.

7.12.2 Bridge Ports

The individual MAC Entity associated with each Bridge Port shall have a separate individual MAC Address. This address is used for any MAC procedures required by the particular MAC.

Frames received from or relayed to the LAN to which a Port is attached and which carry a MAC Address for the Port in the destination MAC address field shall be submitted to the MAC Service User (LLC), and to the LLC Service User for the LSAP identified by the destination LLC Address exactly as for an end station.

7.12.3 Spanning Tree Protocol Entities and GARP Entities

Spanning Tree Protocol Entities only receive and transmit BPDUs. These are only received and transmitted from other Spanning Tree Protocol Entities (or where two Bridge Ports are connected to the same LAN, to and from themselves).

GARP Entities only receive and transmit GARP PDUs (12.10) that are formatted according to the requirements of the GARP Applications they support. These are only received and transmitted from other GARP Entities.

A Spanning Tree Protocol Entity or a GARP Entity uses the DL_UNITDATA.request primitive (see IEEE Std 802.2) provided by the individual LLC Entities associated with each active Bridge Port to transmit BPDUs or GARP PDUs. Each PDU is transmitted on one selected Bridge Port. PDUs are received through corresponding DL_UNITDATA.indication primitives. The source_address and destination_address parameters of the DL_UNITDATA.request primitive shall both denote the standard LLC address assigned to the Bridge Spanning Tree Protocols. This identifies the Spanning Tree Protocol Entity and the GARP Entity among other users of LLC.

Each DL_UNITDATA.request primitive gives rise to the transmission of an LLC UI command PDU, which conveys the BPDU or GARP PDU in its information field. The source and destination LLC address fields are set to the values supplied in the request primitive.

The value assigned to the Bridge Spanning Tree Protocol LLC address is given in Table 7-9.[8]

Table 7-9—Standard LLC address assignment

Assignment	Value
Bridge Spanning Tree Protocol	01000010

Code Representation: The least significant bit of the value shown is the right-most. The bits increase in significance from right to left. It should be noted that the code representation used here has been chosen in order to maintain consistency with the representation used elsewhere in this standard; however, it differs from the representation used in ISO/IEC TR 11802-1: 1997.

This standard defines a Protocol Identifier field, present in all BPDUs (Clause 9) and GARP PDUs (12.10), which serves to identify different protocols supported by Spanning Tree Protocol Entities and GARP Entities, within the scope of the LLC address assignment. This standard specifies a single value of the Protocol Identifier in Clause 9 for use in BPDUs. This value serves to identify BPDUs exchanged between Spanning Tree Protocol Entities operating the Rapid Spanning Tree Algorithm and Protocol specified in Clause 17. A second value of this protocol identifier for use in GARP PDUs is defined in 12.10. This value serves to identify GARP PDUs exchanged between GARP Participants operating the protocol specified in Clause 12. Further values of this field are reserved for future standardization.

A Spanning Tree Protocol Entity or GARP Entity that receives a BPDU or a GARP PDU with an unknown Protocol Identifier shall discard that PDU.

A Spanning Tree Protocol Entity that operates the Rapid Spanning Tree Algorithm and Protocol specified in Clause 17 always transmits BPDUs addressed to all other Spanning Tree Protocol Entities attached to the LAN on which the frame containing the BPDU is transmitted. A 48-bit Universal Address, known as the

[8] ISO/IEC TR 11802-1: 1997 contains the full list of standard LLC address assignments, and documents the criteria for assignment.

Bridge Group Address, has been assigned for this purpose and shall be used in the destination address field of all MAC frames conveying BPDUs. Its value is specified in Table 7-10. This group address shall be configured in the Permanent Database (7.12.6) in order to confine BPDUs to the individual LAN on which they are transmitted.

Table 7-10—Reserved addresses

Assignment	Value
Bridge Group Address	01-80-C2-00-00-00
IEEE Std 802.3x Full Duplex PAUSE operation	01-80-C2-00-00-01
IEEE Std 802.3ad Slow_Protocols_Multicast address	01-80-C2-00-00-02
IEEE P802.1X PAE address	01-80-C2-00-00-03
Reserved for future standardization	01-80-C2-00-00-04
Reserved for future standardization	01-80-C2-00-00-05
Reserved for future standardization	01-80-C2-00-00-06
Reserved for future standardization	01-80-C2-00-00-07
Reserved for future standardization	01-80-C2-00-00-08
Reserved for future standardization	01-80-C2-00-00-09
Reserved for future standardization	01-80-C2-00-00-0A
Reserved for future standardization	01-80-C2-00-00-0B
Reserved for future standardization	01-80-C2-00-00-0C
Reserved for future standardization	01-80-C2-00-00-0D
Reserved for future standardization	01-80-C2-00-00-0E
Reserved for future standardization	01-80-C2-00-00-0F

A GARP Entity that

a) Operates the GARP as specified in Clause 12; and
b) Supports a given GARP Application

always transmits GARP PDUs addressed to all other GARP Entities that

c) Implement the same GARP Application; and
d) Are attached to the LAN on which the frame containing the GARP PDU is transmitted.

A group MAC Address, specific to the GARP Application concerned, shall be used as the destination MAC Address field to address this group of GARP Entities. A set of 48-bit Universal Addresses, known as GARP Application addresses, has been assigned for that purpose. The values of the GARP Application addresses are defined in Table 12-1. These group MAC Addresses are reserved for assignment to standard protocols, according to the criteria for such assignments (Clause 5.5 of ISO/IEC TR 11802-2) and are configured as Static Filtering Entries in the Filtering Database (7.9.1) and Permanent Database (7.9.6) as follows:

e) GARP Application addresses assigned to GARP Applications that are supported by the Bridge are configured to confine GARP PDUs for that GARP Application to the individual LAN on which they are transmitted;
f) GARP Application addresses assigned to GARP Applications that are not supported by the Bridge are not configured in the Filtering Database or Permanent Database.

Management shall not provide the capability to delete or modify entries in the Permanent or Filtering Databases for supported GARP application address or to create entries for unsupported GARP application addresses.

The source address field of MAC frames conveying BPDUs or GARP PDUs for GARP Applications supported by the Bridge shall convey the individual MAC Address for the Bridge Port through which the PDU is transmitted (7.12.2).

7.12.4 Bridge Management Entities

A Bridge Management Entity transmits and receives protocol data units using the Service provided by an individual LLC Entity associated with a Bridge Port. Each LLC entity uses the MAC Service provided by the individual MAC Entity of that Port and supported by the Bridged Local Area Network as a whole.

As a user of the MAC Service provided by a Bridged Local Area Network, the Bridge Management Entity can be attached to any point in the network. Frames addressed to the Bridge Management Entity will be relayed by Bridges if necessary to reach the LAN to which it is attached.

In order to ensure that received frames are not duplicated a unique address is associated with each point of attachment. A Bridge Management Entity for a specific Bridge is addressed by one or more individual MAC Addresses in conjunction with the higher-layer protocol identifier and addressing information. It may share one or more points of attachment to the Bridged Local Area Network with the Ports of the Bridge with which it is associated.

This standard specifies a standard group address for public use that serves to convey management requests to the Bridge Management Entities associated with all Bridge Ports that are attached to a Bridged Local Area Network. A management request that is conveyed in a MAC frame carrying this address value in the destination address field will generally elicit multiple responses from a single Bridge. This address is known as the All LANs Bridge Management Group Address and takes the value specified in Table 7-11.

Table 7-11—Addressing bridge management

Assignment	Value
All LANs Bridge Management Group Address	01-80-C2-00-00-10

7.12.5 Unique identification of a bridge

A unique 48-bit Universally Administered MAC Address, termed the Bridge Address, shall be assigned to each Bridge. The Bridge Address may be the individual MAC Address of a Bridge Port, in which case, use of the address of the lowest numbered Bridge Port (Port 1) is recommended.

7.12.6 Reserved addresses

Frames containing any of the group MAC Addresses specified in Table 7-10 in their destination address field shall not be relayed by the Bridge. They are configured in the Permanent Database. Management shall not provide the capability to modify or remove these entries from the Permanent or the Filtering Databases. These group MAC Addresses are reserved for assignment to standard protocols, according to the criteria for such assignments (Clause 5.5 of ISO/IEC TR 11802-2).

7.12.7 Points of attachment and connectivity for Higher-Layer Entities

The Higher-Layer Entities in a Bridge, such as the Spanning Tree Protocol Entity (7.10), GARP Entity (7.10), and Bridge Management (7.11), are modeled as attaching directly to one or more individual LANs connected by the Bridge's Ports, in the same way that any distinct end station is attached to the network. While these entities and the relay function of the Bridge use the same individual MAC entities to transmit and receive frames, the addressing and connectivity to and from these entities is the same as if they were attached as separate end stations "outside" the Port or Ports where they are actually attached. Figure 7-9 is functionally equivalent to Figure 7-3, but illustrates this logical separation between the points of attachment used by the Higher-Layer Entities and those used by the MAC Relay Entity.

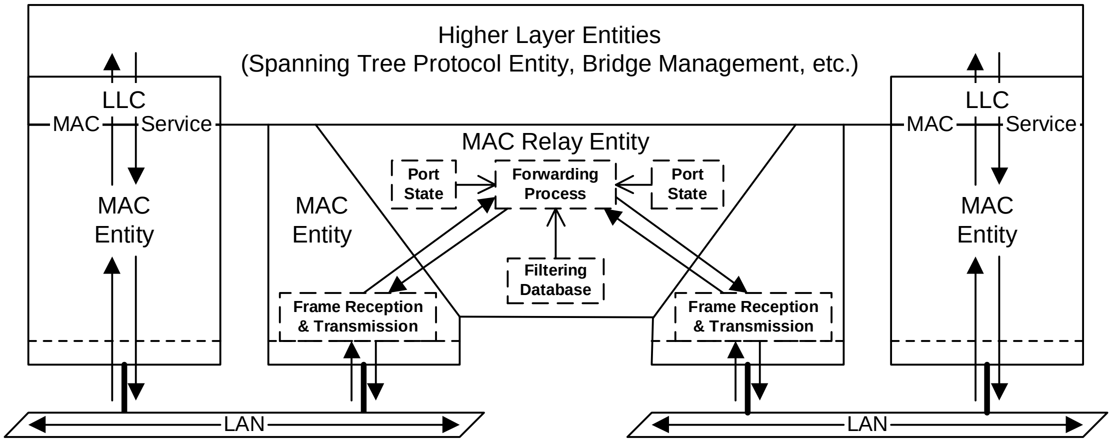


Figure 7-9—Logical points of attachment of the Higher-Layer and Relay Entities

Figure 7-10 depicts the information used to control the forwarding of frames from one Bridge Port to another (the Port States and the content of the Filtering Database) as a series of switches (shown in the open, disconnected state) inserted in the path provided by the MAC Relay Entity. For the Bridge to forward a given frame between two Ports, all three switches must be in the closed state. While showing Higher-Layer Entities sharing the point of attachment to each LAN used by each Bridge Port to forward frames, this figure further illustrates a point made by Figure 7-9—controls placed in the forwarding path have no effect upon the ability of a Higher-Layer Entity to transmit and receive frames to or from a given LAN using a direct attachment to that LAN (e.g., from entity A to LAN A), they only affect the path taken by any indirect transmission or reception (e.g., from entity A to or from LAN B).

The functions provided by Higher-Layer Entities can be categorized as requiring either

a) A single point of attachment to the Bridged Local Area Network, providing connectivity to stations attached to the network at any point (subject to administrative control), as does Bridge Management; or
b) A distinct point of attachment to each individual LAN attached by a Bridge Port, providing connectivity only to peer entities connected directly to that LAN, as do the Spanning Tree Protocol Entity and the GARP Entity.

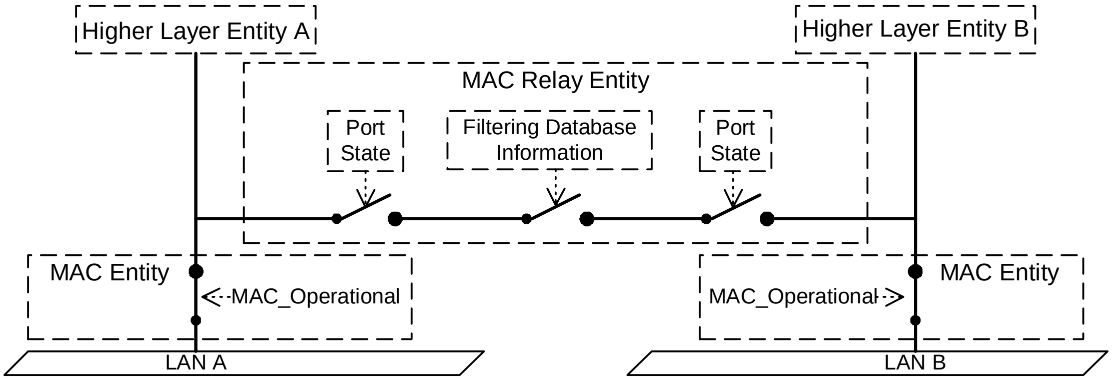


Figure 7-10—Effect of control information on the forwarding path

In the latter case it is essential that the function associate each received and transmitted frame with a point of attachment. Frames transmitted or received via one point of attachment are not to be relayed to and from other Ports and attached LANs, so the MAC Addresses (7.12.3, 7.12.6, Table 7-10) used to reach these functions are permanently configured in the Filtering Database (7.9.6).

NOTE 1 —Addresses used to reach functions with distinct points of attachment are generally group MAC Addresses.

NOTE 2—A single Higher-Layer entity can incorporate both a function requiring a single point of attachment and a function requiring distinct points of attachment. The two functions are reached using different MAC addresses.

Figure 7-11 illustrates forwarding path connectivity for frames destined for Higher-Layer Entities requiring per-Port points of attachment. Configuration of the Permanent Database in all Bridges to prevent relay of frames addressed to these entities means that they receive frames only via their direct points of attachment (i.e., from LAN A to entity A, and from LAN B to entity B), regardless of Port states.

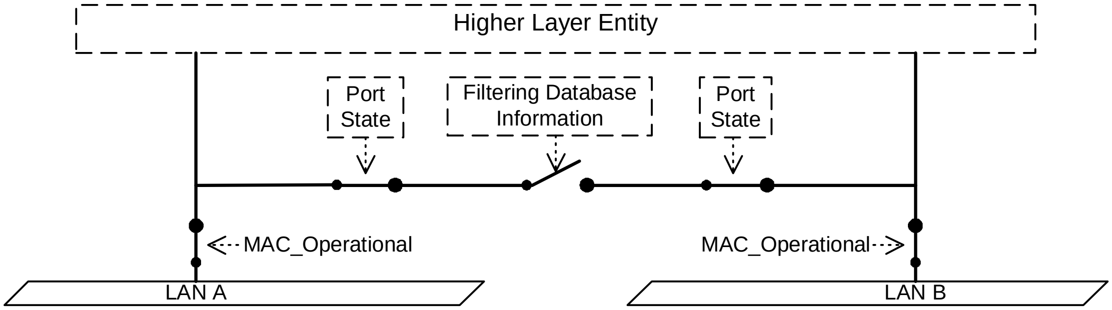


Figure 7-11—Per-Port points of attachment

Figure 7-12 and Figure 7-13 illustrate forwarding path connectivity for frames destined for a Higher-Layer Entity requiring a single point of attachment. In both figures, the Filtering Database permits relay of frames, as do the Port states in Figure 7-12 where frames received from LAN B are relayed by the Bridge to the entity and to LAN A.

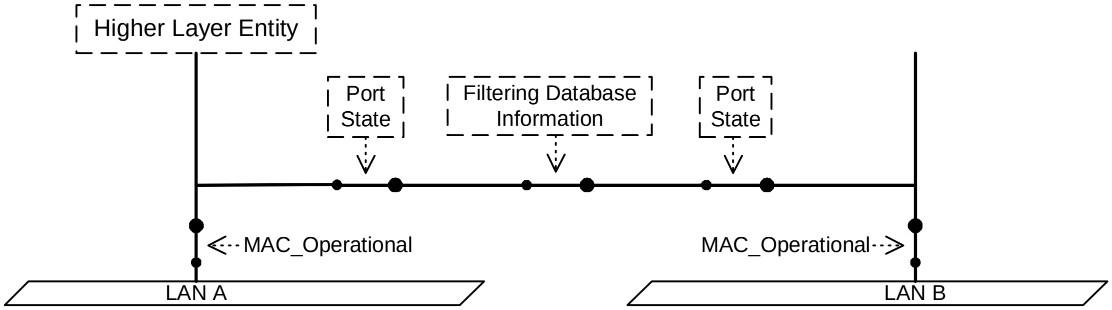


Figure 7-12—Single point of attachment—relay permitted

In Figure 7-13 frames received from LAN A are received by the entity directly, but frames received from LAN B are not relayed by the Bridge, and will only be received by the entity if another forwarding path is provided between LANs A and B. If the Discarding Port state shown resulted from spanning tree computation (and not from disabling the Administrative Bridge Port State), such a path will exist via one or more Bridges. If there is no active Spanning Tree path from B to A, the network has partitioned into two separate Bridged Local Area Networks and the Higher-Layer Entity shown is reachable only via LAN A.

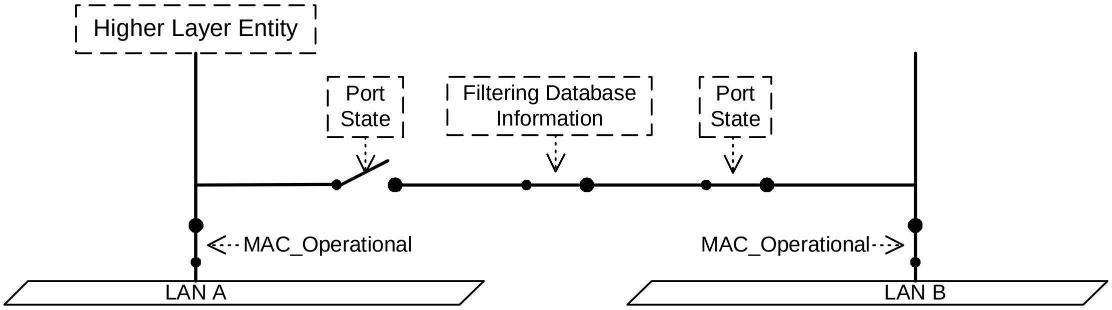


Figure 7-13—Single point of attachment—relay not permitted

Specific Higher-Layer Entities can take notice of the Administrative Bridge Port State, as required by their specification. The Spanning Tree Protocol Entity is one such example—BPDUs are never transmitted or received on Ports with an Administrative Bridge Port State of Disabled.

If a Bridge Port's MAC Entity is not operational, a Higher-Layer Entity directly attached at the Port will not be reachable, as Figure 7-14 illustrates. The Spanning Tree Protocol Entity ensures that the Port State is Discarding if the MAC_Operational (6.4.2) is FALSE, even if the Administrative Bridge Port State is Enabled.

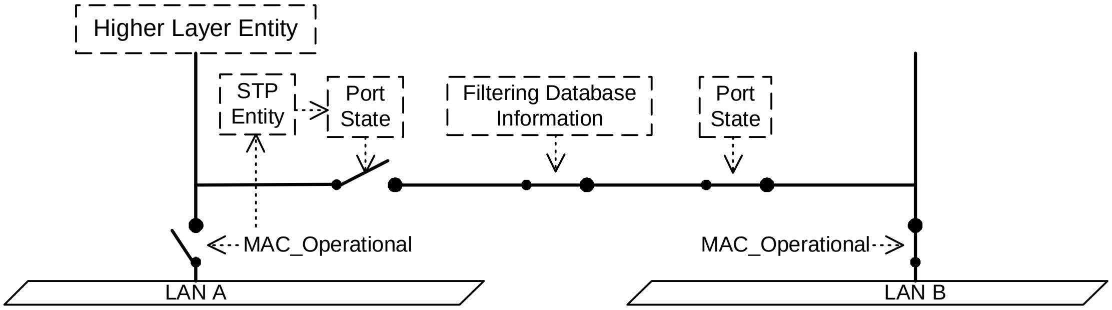


Figure 7-14—Effect of Port State

The connectivity provided to Higher-Layer Entities and to the LANs that compose a Bridged Local Area Network can be further controlled by a Bridge Port operating as a network access port (IEEE Std 802.1X). The operation of Port-based access control has the effect of creating two distinct points of access to the LAN. One, the *uncontrolled Port*, allows transmission and reception of frames to and from the attached LAN regardless of the authorization state; the other, the *controlled Port*, only allows transmission following authorization. If the port is not authorized, the Spanning Tree Protocol Entity, which uses the controlled port (as does the MAC Relay Entity), will be unable to exchange BPDUs with other Bridges attached to LAN A, and will set the Bridge Port State to Discarding.

NOTE—If the Spanning Tree Protocol Entity was not aware of the Unauthorized state of the Port, and believed that it was transmitting and receiving BPDUs it might assign a Bridge Port State of Forwarding. Following authorization a temporary loop in network connectivity might then be created.

Figure 7-15 illustrates the connectivity provided to Higher-Layer Entities if the MAC entity is physically capable of transmitting and receiving frames, i.e., MAC_Operational is TRUE, but AuthControlledPortStatus is Unauthorized. Higher-Layer Entity A and the PAE (the port access entity that operates the authorization protocol) are connected to the uncontrolled port and can transmit and receive frames using the MAC entity associated with the Port, which Higher-Layer Entity B cannot. None of the three entities can transmit or receive to or from LAN B.

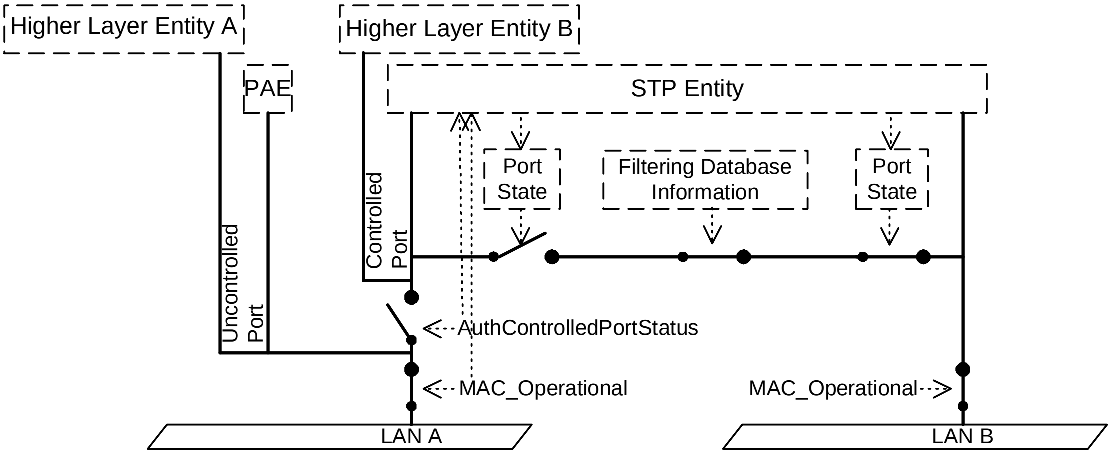


Figure 7-15—Effect of authorization

NOTE—The administrative and operational state values associated with the MAC, the Port's authorization state, and the Bridge Port State, equate to the ifAdminStatus and ifOperStatus parameters associated with the corresponding interface definitions; see IETF RFC 2233.

8. Spanning tree algorithm and protocol

In IEEE Std 802.1D, 1998 Edition, and prior editions of this standard, this clause specified the spanning tree algorithm and protocol (STP).[9] STP has now been superseded by the Rapid Spanning Tree Protocol (RSTP) specified in Clause 17 of this standard. Implementation of RSTP and an appropriate claim of conformance shall be substituted for any implementation and claim of conformance to STP required by any other standard.

[9] To facilitate support of legacy equipment, prior revisions of this standard are electronically available.
Consult http://grouper.ieee.org/groups/802/1 and http://standards.ieee.org for information on the Get802 program.

9. Encoding of bridge protocol data units

This clause specifies the structure and encoding of the Bridge Protocol Data Units exchanged between Spanning Tree Protocol Entities for the Rapid Spanning Tree Protocol (Clause 17).

9.1 Structure

9.1.1 Transmission and representation of octets

All BPDUs shall contain an integral number of octets. The octets in a BPDU are numbered starting from 1 and increasing in the order they are put into a Data Link Service Data Unit (DLSDU). The bits in an octet are numbered from 1 to 8, where 1 is the low-order bit.

When consecutive bits within an octet are used to represent a binary number, the higher bit number has the most significant value. When consecutive octets are used to represent a binary number, the lower octet number has the most significant value. All Bridge Protocol Entities respect these bit and octet ordering conventions, thus allowing communications to take place.

9.1.2 Components

A Protocol Identifier is encoded in the initial octets of all BPDUs. This standard specifies a single Protocol Identifier value for use in BPDUs. All other Protocol Identifier values are reserved for future standard use. This standard places no further restriction on the structure, encoding, or use of BPDUs with different values of the Protocol Identifier field, should these exist, by other standard protocols.

9.2 Encoding of parameter types

9.2.1 Encoding of protocol identifiers

A Protocol Identifier shall be encoded in two octets.

9.2.2 Encoding of protocol version identifiers

A Protocol Version Identifier shall be encoded in one octet. If two Protocol Version Identifiers are interpreted as unsigned binary numbers, the greater number identifies the more recently defined Protocol Version.

9.2.3 Encoding of BPDU types

The type of the BPDU shall be encoded as a single octet. The bit pattern contained in the octet merely serves to distinguish the type; no ordering relationship between BPDUs of different types is implied.

9.2.4 Encoding of flags

A flag shall be encoded as a bit in a single octet. A flag is set if the bit takes the value 1. A number of flags may be encoded in a single octet. Bits in the octet that do not correspond to flags defined for the BPDU’s type are reset, i.e., shall take the value 0. No additional flags will be defined for a BPDU of given protocol version and type.

9.2.5 Encoding of Bridge Identifiers

A Bridge Identifier shall be encoded as eight octets, taken to represent an unsigned binary number. Two Bridge Identifiers may be numerically compared and the lesser shall denote the Bridge of the better priority.

NOTE 1—Use of the terms “higher” and “lower” to describe both the relative numerical values and the relative priority of Spanning Tree priority information can cause confusion, as lesser numbers convey better priorities. In this clause and in Clause 17 (Rapid Spanning Tree), relative numeric values are described as “least,” “lesser,” “equal,” and “greater,” and their comparisons as “less than,” “equal to,” or “greater than,” while relative Spanning Tree priorities are described as “best,” “better,” “the same,” “different,” and “worse” and their comparisons as “better than,” “the same as,” “different from,” and “worse than.” The terms “superior” and “inferior” describe comparisons not simply based on strict ordered comparison of priority components.

The four most significant bits of the most significant octet of a Bridge Identifier comprise a settable priority component that permits the relative priority of Bridges to be managed (17.13.7 and Clause 14). The next most significant twelve bits of a Bridge Identifier (the four least significant bits of the most significant octet, plus the second most significant octet) comprise a locally assigned system ID extension. The six least significant octets ensure the uniqueness of the Bridge Identifier; they shall be derived from the globally unique Bridge Address (7.12.5) according to the following procedure.

NOTE 2—The number of bits that are considered to be part of the system ID (60 bits) differs in this version of the standard from the 1998 and prior versions (formerly, the priority component was 16 bits and the system ID component 48 bits). This change was made in order to allow implementations of Multiple Spanning Trees (IEEE Std 802.1Q) to make use of the 12-bit system ID extension as a means of generating a distinct Bridge Identifier per VLAN, rather than forcing such implementations to allocate up to 4094 MAC addresses for use as Bridge Identifiers. To maintain management compatibility with older implementations, the priority component is still considered, for management purposes, to be a 16-bit value, but the values that it can be set to are restricted to only those values where the least significant 12 bits are zero (i.e., only the most significant 4 bits are settable).

The third most significant octet is derived from the initial octet of the MAC Address; the least significant bit of the octet (Bit 1) is assigned the value of the first bit of the Bridge Address, the next most significant bit is assigned the value of the second bit of the Bridge Address, and so on. The fourth through eighth octets are similarly assigned the values of the second to the sixth octets of the Bridge Address.

9.2.6 Encoding of Root Path Cost

Root Path Cost shall be encoded as four octets, taken to represent an unsigned binary number, a multiple of arbitrary cost units. Subclause 17.14 contains recommendations as to the increment to the Root Path Cost, in order that some common value can be placed on this parameter without requiring a management installation practice for Bridges in a Bridged Local Area Network.

9.2.7 Encoding of Port Identifiers

A Port Identifier shall be encoded as two octets, taken to represent an unsigned binary number. If two Port Identifiers are numerically compared, the lesser number denotes the Port of better priority. The more significant octet of a Port Identifier is a settable priority component that permits the relative priority of Ports on the same Bridge to be managed (17.13.7 and Clause 14). The less significant twelve bits is the Port Number expressed as an unsigned binary number. The value 0 is not used as a Port Number.

NOTE—The number of bits that are considered to be part of the Port Number (12 bits) differs from the 1998 and prior versions of this standard (formerly, the priority component was 8 bits and the Port Number component also 8 bits). This change acknowledged that modern switched LAN infrastructures call for increasingly large numbers of Ports to be supported in a single Bridge. To maintain management compatibility with older implementations, the priority component is still considered, for management purposes, to be an 8-bit value, but the values that it can be set to are restricted to those where the least significant 4 bits are zero (i.e., only the most significant 4 bits are settable).

9.2.8 Encoding of Timer Values

Timer Values shall be encoded in two octets, taken to represent an unsigned binary number multiplied by a unit of time of 1/256 of a second. This permits times in the range 0 to, but not including, 256 s to be represented.

9.2.9 Encoding of Port Role values

Port Role values shall be encoded in two consecutive flag bits, taken to represent an unsigned integer, as follows:

a) A value of 0 indicates Unknown.
b) A value of 1 indicates Alternate or Backup.
c) A value of 2 indicates Root.
d) A value of 3 indicates Designated.

The Unknown value of Port Role cannot be generated by a valid implementation; however, this value is accepted on receipt.

NOTE—If the Unknown value of the Port Role parameter is received, the state machines will effectively treat the RST BPDU as if it were a Configuration BPDU.

9.2.10 Encoding of Length Values

Version 1 Length Values shall be encoded in one octet, taken to represent an unsigned binary number.

9.3 BPDU formats and parameters

9.3.1 Configuration BPDUs

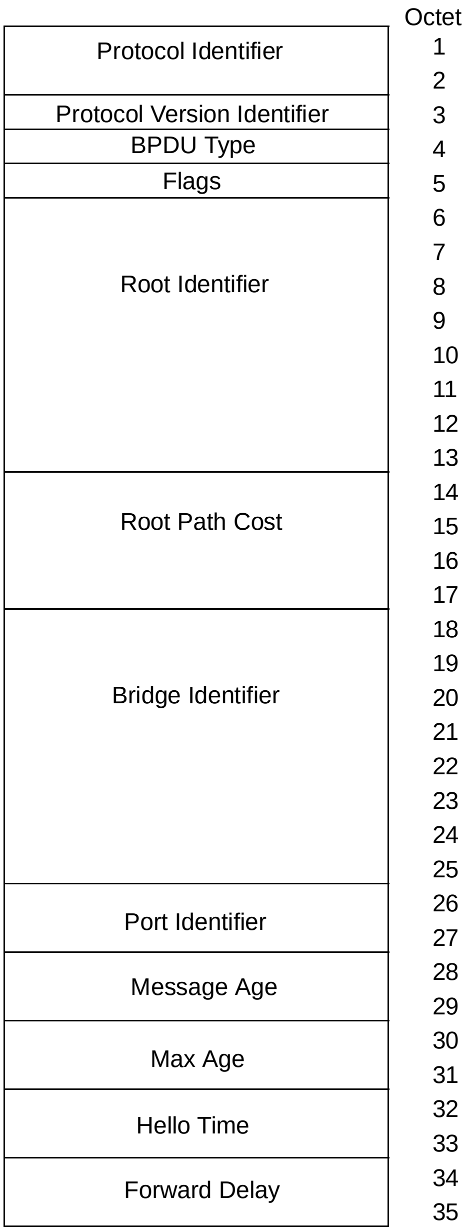
The format of the Configuration BPDUs is shown in Figure 9-1. Each transmitted Configuration BPDU shall contain the following parameters and no others. Where a specific parameter value is indicated in this subclause, that parameter value shall be encoded in all transmitted Configuration BPDUs:

a) The Protocol Identifier is encoded in Octets 1 and 2. It takes the value 0000 0000 0000 0000, which identifies the Rapid Spanning Tree Protocol as specified in Clause 17.

NOTE—This value of the Protocol Identifier also identifies the Spanning Tree Algorithm and Protocol specified in previous editions of this standard.

b) The Protocol Version Identifier is encoded in Octet 3. It takes the value 0000 0000.
c) The BPDU Type is encoded in Octet 4. This field takes the value 0000 0000.
This denotes a Configuration BPDU.
d) The Topology Change Acknowledgment flag is encoded in Bit 8 of Octet 5.
e) The Topology Change flag is encoded in Bit 1 of Octet 5.
f) The remaining flags, Bits 2 through 7 of Octet 5, are unused and take the value 0.
g) The Root Identifier is encoded in Octets 6 through 13.
h) The Root Path Cost is encoded in Octets 14 through 17.
i) The Bridge Identifier is encoded in Octets 18 through 25.
j) The Port Identifier is encoded in Octets 26 and 27.
k) The Message Age timer value is encoded in Octets 28 and 29.
l) The Max Age timer value is encoded in Octets 30 and 31.
m) The Hello Time timer value is encoded in Octets 32 and 33.
n) The Forward Delay timer value is encoded in Octets 34 and 35.

The Message Age (Octets 28 and 29) shall be less than Max Age (Octets 30 and 31).



	Octet
Protocol Identifier	1 2
Protocol Version Identifier	3
BPDU Type	4
Flags	5
Root Identifier	6 7 8 9 10 11 12 13
Root Path Cost	14 15 16 17
Bridge Identifier	18 19 20 21 22 23 24 25
Port Identifier	26 27
Message Age	28 29
Max Age	30 31
Hello Time	32 33
Forward Delay	34 35

Figure 9-1—Configuration BPDU parameters and format

9.3.2 Topology Change Notification BPDUs

The format of the Topology Change Notification BPDUs is shown in Figure 9-2. Each transmitted Topology Change Notification BPDU shall contain the following parameters and no others. Where a specific parameter value is indicated in this subclause, that parameter value shall be encoded in all transmitted Topology Change Notification BPDUs:

a) The Protocol Identifier is encoded in Octets 1 and 2 . It takes the value 0000 0000 0000 0000.

b) The Protocol Version Identifier is encoded in Octet 3 . It takes the value 0000 0000.

c) The BPDU Type is encoded in Octet 4 . This field takes the value 1000 0000 (where bit 8 is shown at the left of the sequence). This denotes a Topology Change Notification BPDU.

	Octet
Protocol Identifier	1
	2
Protocol Version Identifier	3
BPDU Type	4

Figure 9-2—Topology change notification BPDU parameters and format

9.3.3 Rapid Spanning Tree BPDUs (RST BPDUs)

The format of the RST BPDUs is shown in Figure 9-3. Each transmitted RST BPDU shall contain the following parameters and no others. Where a specific parameter value is indicated in this subclause, that parameter value shall be encoded in all transmitted RST BPDUs.

a) The Protocol Identifier is encoded in Octets 1 and 2. It takes the value 0000 0000 0000 0000.
b) The Protocol Version Identifier is encoded in Octet 3. It takes the value 0000 0010.
c) The BPDU Type is encoded in Octet 4. This field takes the value 0000 0010. This denotes a Rapid Spanning Tree BPDU.
d) The Topology Change flag is encoded in Bit 1 of Octet 5 (see 17.21.20).
e) The Proposal flag is encoded in Bit 2 of Octet 5 (see 17.21.20).
f) The Port Role is encoded in Bits 3 and 4 of Octet 5 (see 17.21.20).
g) The Learning flag is encoded in Bit 5 of Octet 5 (see 17.21.20).
h) The Forwarding flag is encoded in Bit 6 of Octet 5 (see 17.21.20).
i) The Agreement flag is encoded in Bit 7 of Octet 5 (see 17.21.20).
j) The Topology Change Acknowledgment flag is encoded in Bit 8 of Octet 5 as zero (see 17.21.20).
k) The Root Identifier is encoded in Octets 6 through 13 (see 17.19.21, 17.21.20).
l) The Root Path Cost is encoded in Octets 14 through 17 (see 17.19.21, 17.21.20).
m) The Bridge Identifier is encoded in Octets 18 through 25. (see 17.19.21, 17.21.20)
n) The Port Identifier is encoded in Octets 26 and 27 (see 17.19.21, 17.21.20).
o) The Message Age timer value is encoded in Octets 28 and 29 (see 17.19.21, 17.21.20).
p) The Max Age timer value is encoded in Octets 30 and 31 (see 17.19.21, 17.21.20).
q) The Hello Time timer value is encoded in Octets 32 and 33 (see 17.19.21, 17.21.20).
r) The Forward Delay timer value is encoded in Octets 34 and 35 (see 17.19.21, 17.21.20).
s) The Version 1 Length value is encoded in Octet 36. It takes the value 0000 0000, which indicates that there is no Version 1 protocol information present.

NOTE—The presence of a Version 1 Length value of 0, indicating that no version 1 information is present, is required in Version 2 BPDUs in order to make it possible to define subsequent versions of the protocol that can carry additional parameters other than those defined for Version 1 of the protocol (previously defined in IEEE Std 802.1G).

The Message Age (Octets 28 and 29) shall be less than Max Age (Octets 30 and 31).

9.3.4 Validation of received BPDUs

A Spanning Tree Protocol Entity shall process a received BPDU as specified in 17.15 if and only if the BPDU contains at least four octets and the Protocol Identifier has the value specified for BPDUs (9.3.2), and

a) The BPDU Type denotes a Configuration BPDU and the BPDU contains at least 35 octets, and the BPDU's Message Age is less than its Max Age parameter, and the Bridge Identifier and Port Identifier parameters from the received BPDU do not match the values that would be transmitted in a BPDU from this port; or

NOTE 1—If the Bridge Identifier and Port Identifier both match the values that would be transmitted in a Configuration BPDU, the BPDU is discarded to prevent processing of the Port's own BPDUs; for example, if they are received by the Port as a result of a loopback condition. If a loopback condition exists, there will be other undesirable effects caused by

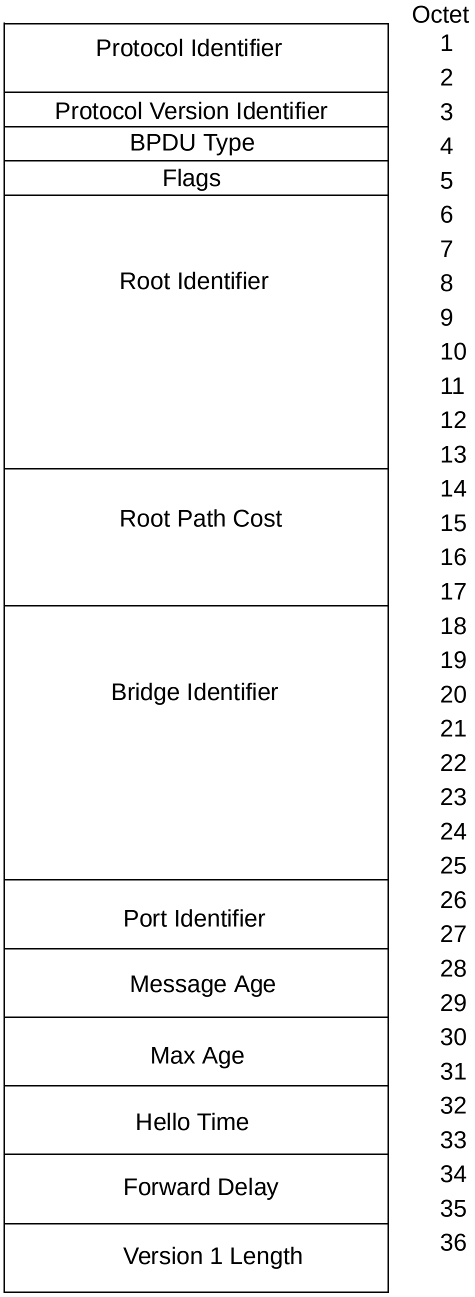


Figure 9-3—RST BPDU parameters and format

the looping back of data frames relayed through the Port. When transmitting RST BPDUs, the Rapid Spanning Tree Protocol implements a more sophisticated check, so this test is not applied to RST BPDUs (see below).

b) The BPDU Type denotes a Topology Change Notification BPDU; or
c) The BPDU Type denotes a Rapid Spanning Tree BPDU and the BPDU contains at least 36 octets.

NOTE 2—The RSTP Port Information state machine (see 17.27) checks that the BPDU's Message Age is less its Max Age, and if not, will immediately age out the received information.

The following rules apply to the validation and interpretation of BPDUs, to ensure that backwards compatibility is maintained between versions of this protocol.

For an implementation that supports version A of the protocol, a received BPDU of a given type that carries a protocol version number B is interpreted as follows:

d) Where B is greater than or equal to A, the BPDU shall be interpreted as if it carried the supported version number, A. Specifically:
   1) All BPDU types, parameters, and flags that are defined in version A shall be interpreted in the manner specified for version A of the protocol for the given BPDU type.
   2) All BPDU types, parameters, and flags that are undefined in version A for the given BPDU type shall be ignored.
   3) All octets that appear in the BPDU beyond the largest numbered octet defined for version A for the given BPDU type shall be ignored.
e) Where B is less than A, the BPDU shall be interpreted as specified for the version number, B, carried in the BPDU. Specifically:
   1) All BPDU parameters and flags shall be interpreted in the manner specified for version B of the protocol for the given BPDU type.
   2) All BPDU parameters and flags that are undefined in version B for the given BPDU type shall be ignored.
   3) All octets that appear in the BPDU beyond the largest numbered octet defined for version B for the given BPDU type shall be ignored.

NOTE 3—In other words, if the protocol version implemented differs from the protocol version number carried in the BPDU, then only those BPDU types, parameters, and flags that are specified within the lesser numbered protocol version are interpreted by the implementation (in accordance with the lesser numbered protocol version’s specification), and no attempt is made to interpret any additional BPDU types, parameters, and flags that may be specified within the greater numbered protocol version. In the specific case of STP (version 0) and RSTP (version 2), as there is only a single RST BPDU type defined in version 2, and as the RST BPDU type is undefined in version 0, a version 0 implementation will ignore all RST BPDUs. Version 2 implementations, however, recognize and process both version 0 and version 2 BPDUs. As version 2 makes no changes to the BPDU types defined for version 0 (and always transmits such BPDU types with 0 as the version identifier), version 0 BPDUs are always interpreted by version 2 implementations according to their version 0 definition.

10. GARP Multicast Registration Protocol (GMRP)

10.1 Purpose

The GARP (Generic Attribute Registration Protocol) Multicast Registration Protocol (GMRP) provides a mechanism that allows end stations and MAC Bridges to dynamically register (and subsequently, deregister) Group membership information with the Bridges attached to the same LAN, and disseminates that information across all the Bridges that support Extended Filtering Services in the Bridged Local Area Network. The operation of GMRP relies upon the services provided by GARP, defined in Clause 12.

The information registered, de-registered, and disseminated via GMRP is in the following forms:

a) *Group membership information*. This indicates the presence of GMRP participants that are members of a particular Group (or Groups), and carries the group MAC Address(es) associated with the Group(s). The exchange of specific Group membership information can result in the creation or updating of Group Registration Entries in the Filtering Database to indicate the Port(s) on which members of the Group(s) have been registered. The structure of these entries is described in 7.9.3.

b) *Group service requirement information*. This indicates that one or more GMRP participants require Forward All Groups or Forward Unregistered Groups to be the default Group filtering behavior (see 6.6.7 and 7.9.4).

Registration of Group membership information makes Bridges aware that frames destined for the group MAC Address concerned should only be forwarded in the direction of the registered members of the Group. Therefore, forwarding of frames destined for the address associated with that Group occurs only on Ports on which such membership registration has been received.

Registration of Group service requirement information makes the Bridges aware that Ports that can forward frames in the direction from which the information has been received should modify their default Group forwarding behavior in accordance with the service requirement expressed.

NOTE—Modification of default Group forwarding behavior allows Bridge Ports to accommodate GMRP-unaware devices in the Bridged Local Area Network by forwarding frames destined for unregistered group MAC Addresses.

The operation of GMRP can result in

c) The propagation of Group membership information and Group service requirement information, and consequent creation, updating, or deletion of Group Registration Entries in the Filtering Databases of all Bridges in the network that support Extended Filtering Services.

d) Consequent changes to the Group filtering behavior of such Bridges.

10.2 Model of operation

GMRP defines a *GARP Application* (12.2, 12.2.1, and 10.3) that provides the extended filtering services defined in 6.6.5 and 6.6.7. To this end, GMRP makes use of

a) The declaration and propagation services offered by *GARP Information Distribution* (GID; 12.2 and 12.2.2) and *GARP Information Propagation* (GIP; 12.2 and 12.2.3) to declare and propagate Group membership and Group service requirement information within the Bridged Local Area Network.

b) The registration services offered by GID (12.2 and 12.2.2) to allow Group membership and Group service requirement information to control the frame filtering behavior of participating devices.

10.2.1 Propagation of Group Membership information

The Forwarding Process uses the Group Registration Entries in the Filtering Databases to ensure that frames are transmitted only through those Bridge Ports necessary to reach LANs to which Group members are attached. Figure 10-1 illustrates the Group Registration Entries created by GMRP for a single Group.

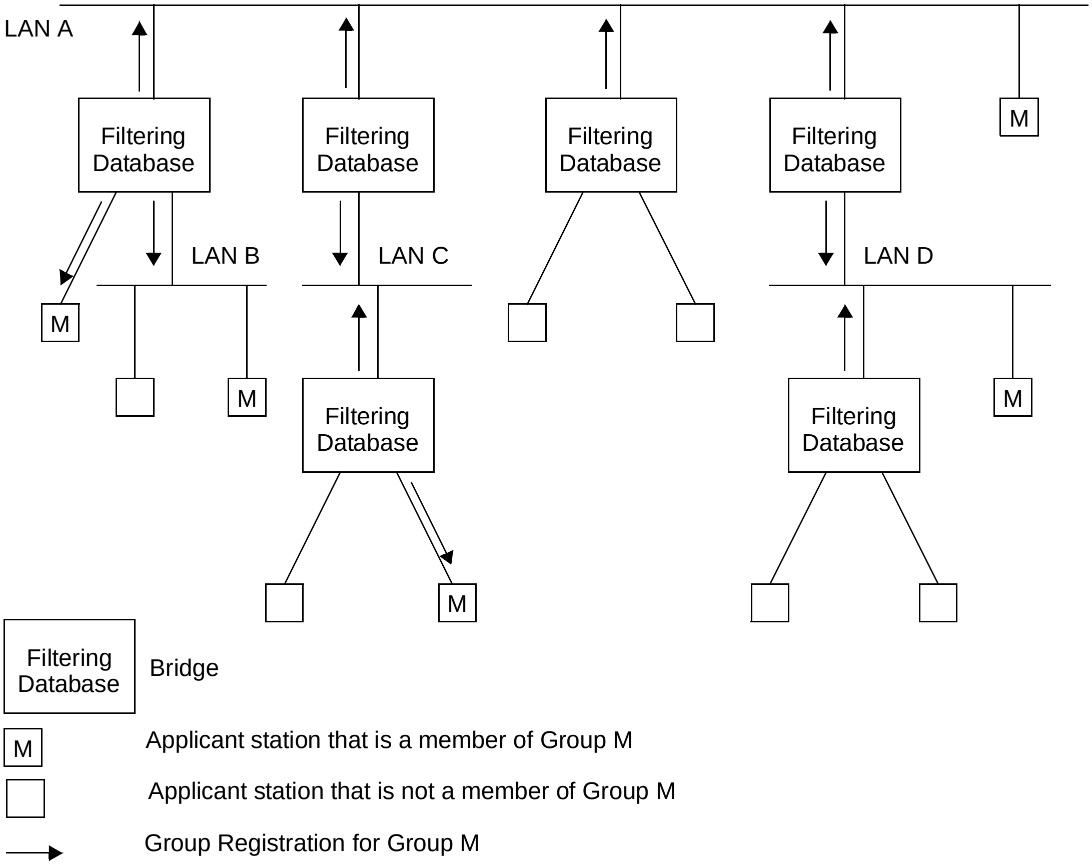


Figure 10-1—Example Directed Graph

By receiving frames from all Ports and forwarding only through Ports for which GMRP has created Group Registration Entries, Bridges facilitate Group distribution mechanisms based on the concept of an Open Host Group. Any GMRP Participants (12.2) that wish to receive frames transmitted to a particular Group or Groups request membership of the Group(s) concerned. Any MAC Service user that wishes to send frames to a particular Group can do so from any point of attachment to the Bridged Local Area Network. These frames can be received on all LANs to which registered GMRP Participants are attached, but the filtering applied by Bridges ensures that frames are not transmitted on LANs that are not part of the active topology between the sources of the frames and the registered Group members. GMRP and the Group Registration Entries thus restrict the frames to pruned subsets of the overall loop free active topology.

NOTE—The term “Open Host Group” comes from the terminology introduced in the definition of the Internet Group Membership Protocol (IGMP) defined by the IETF.

MAC Service users that are sources of MAC frames destined for the Group do not have to register as members of the Group themselves unless they also wish to receive frames transmitted to the Group address by other sources.

10.2.2 Propagation of Group service requirement information

GMRP propagates Group service requirement information in the same manner as for Group Registration information. If any Port in a given Bridge has a registered Group service requirement of All Groups or All Unregistered Groups (expressed in terms of the control information in Static Filtering Entries and/or Dynamic Filtering Entries with a MAC Address specification of All Groups or All Unregistered Groups), this fact is propagated on all other Ports of the Bridge, resulting in the registration of that information on Ports of adjacent Bridges. As a consequence of that registration, the default Group filtering behavior of those Ports can change in order to maintain compatibility with the service requirements expressed by the registered information, as defined in 7.9.4. This ensures that connectivity can be maintained in LANs where the service requirements of different regions of the Bridged Local Area Network differ.

NOTE—In a Bridged Local Area Network where the default Group filtering behavior is not the same for all “edge” Ports, service requirement propagation will tend to result in all “backbone” Ports switching to the highest precedence Group filtering behavior in use in the network. The precedence rules are defined in 7.9.4.

10.2.3 Source pruning

As described in 10.2.1, the operation of GMRP defines a subtree of the Spanning Tree as a result of the creation of Group Registration Entries in the Filtering Databases of the Bridges. End stations are also able to make use of the Group Membership information registered via GMRP to allow them to keep track of the set of Groups for which active members currently exist and the service requirements of upstream devices. This allows end stations that are sources of frames destined for a Group to suppress the transmission of such frames, if their registered Group membership and Group service requirement information indicates that there are no valid recipients of those frames reachable via the LANs to which they are attached.

NOTE—In effect, for the purposes of frame transmission, the end station can be viewed as if it operates as a single Port Bridge, with its own default Group filtering behavior and “Filtering Database” entries updated via GMRP that tell it whether or not multicast frames that it has generated should be forwarded onto the attached LAN. In order to achieve this, it is necessary for the end station to implement both the Registrar and the Applicant functionality of GARP, as described in 12.6.3, 12.7.1, and 12.7.2. The Applicant Only and Simple-Applicant Participants described in 12.6.7 and 12.6.8 do not contain the Registrar functionality that would be required for source pruning.

This end system behavior is known as *source pruning*. Source pruning allows MAC Service users that are sources of MAC frames destined for a number of Groups, such as server stations or routers, to avoid unnecessary flooding of traffic on their local LANs in circumstances where there are no current Group members in the network that wish to receive such traffic.

10.2.4 Use of Group service requirement registration by end stations

The ability to propagate Group service requirement information is described in this standard primarily as a means of propagating the requirements of the Bridges themselves. However, this mechanism can also be used by end stations that have requirements involving some aspect of promiscuous reception, such as Routers or network monitors. A GMRP-aware end station wishing to receive all multicast traffic can declare membership of All Groups on the LAN to which it is attached; similarly, an end station that wishes to receive unregistered multicast traffic can do so by declaring membership of All Unregistered Groups. The circumstances under which these facilities might be used are further discussed in I.2.

10.3 Definition of the GMRP Application

10.3.1 Definition of GARP protocol elements

10.3.1.1 Use of GIP Contexts by GMRP

GMRP, as defined in this standard, operates within the Base Spanning Tree Context, as defined in 12.2.4. A GIP Context identifier value of 0 shall be used to identify this context. The use of GMRP in any other GIP Context is outside the scope of this standard.

10.3.1.2 GMRP Application address

The group MAC Address used as the destination address for GARP PDUs destined for GMRP Participants shall be the GMRP Address identified in Table 12-1.

10.3.1.3 Encoding of GMRP Attribute Types

GMRP defines two Attribute Types (12.10.2.2) that are carried in GARP protocol exchanges, as follows:

a) The Group Attribute Type.
b) The Service Requirement Attribute Type.

The Group Attribute Type is used to identify values of group MAC Addresses. The value of the Group Attribute Type carried in GARP PDUs (12.10.2.2) shall be 1.

The Service Requirement Attribute Type is used to identify values of Group service requirements. The value of the Service Requirement Attribute Type carried in GARP PDUs (12.10.2.2) shall be 2.

10.3.1.4 Encoding of GMRP Attribute Values

Values of instances of the Group Attribute Type shall be encoded as Attribute Values in GARP PDUs (12.10.2.6) as six octets, each taken to represent an unsigned binary number. The octets are derived from the Hexadecimal Representation of a 48-bit MAC Address (defined in IEEE Std 802) as follows:

a) Each two-digit hexadecimal numeral in the Hexadecimal Representation is taken to represent an unsigned hexadecimal value, in the normal way, i.e., the rightmost digit of each numeral represents the least significant digit of the value, the leftmost digit is the most significant.
b) The first octet of the attribute value encoding is derived from the left-most hexadecimal value in the Hexadecimal Representation of the MAC Address. The least significant bit of the octet (bit 1) is assigned the least significant bit of the hexadecimal value, the next most significant bit is assigned the value of the second significant bit of the hexadecimal value, and so on.
c) The second through sixth octets of the encoding are similarly assigned the value of the second through sixth hexadecimal values in the Hexadecimal Representation of the MAC Address.

Values of this Attribute Type shall not include individual MAC Addresses.

Values of instances of the Service Requirement Attribute Type shall be encoded as Attribute Values in GARP PDUs (12.10.2.6) as a single octet, taken to represent an unsigned binary number. Only two values of this type are defined:

a) All Groups shall be encoded as the value 0.
b) All Unregistered Groups shall be encoded as the value 1.

The remaining possible values (2 through 255) are undefined.

10.3.2 Provision and support of Extended Filtering Services

10.3.2.1 End system registration and de-registration

The GMRP Application element of a GARP Participant provides the dynamic registration and de-registration services defined in 6.6.7.1, as follows:

On receipt of a REGISTER_GROUP_MEMBER service primitive, the GMRP Participant issues a GID_Join.request. The attribute_type parameter of the request carries the value of the Group Attribute Type (10.3.1.3) and the attribute_value parameter carries the value of the MAC_ADDRESS parameter of the service primitive.

On receipt of a DEREGISTER_GROUP_MEMBER service primitive, the GMRP Participant issues a GID_Leave.request. The attribute_type parameter of the request carries the value of the Group Attribute Type (10.3.1.3) and the attribute_value parameter carries the value of the MAC_ADDRESS parameter of the service primitive.

On receipt of a REGISTER_SERVICE_REQUIREMENT service primitive, the GMRP Participant issues a GID_Join.request. The attribute_type parameter of the request carries the value of the Service Requirement Attribute Type (10.3.1.3) and the attribute_value parameter carries the value of the REQUIREMENT_SPECIFICATION parameter of the service primitive.

On receipt of a DEREGISTER_SERVICE_REQUIREMENT service primitive, the GMRP Participant issues a GID_Leave.request. The attribute_type parameter of the request carries the value of the Service Requirement Attribute Type (10.3.1.3) and the attribute_value parameter carries the value of the REQUIREMENT_SPECIFICATION parameter of the service primitive.

10.3.2.2 Registration and de-registration events

The GMRP Application element of a GARP Participant responds to registration and de-registration events signalled by GID as follows:

On receipt of a GID_Join.indication whose attribute_type is equal to the value of the Group Attribute Type or the Service Requirement Attribute Type (10.3.1.3), the GMRP Application element specifies the Port associated with the GMRP Participant as Forwarding in the Port Map field of the Group Registration Entry (7.9.3) for the MAC Address specification carried in the attribute_value parameter. If such a Group Registration Entry does not exist in the Filtering Database, a new Group Registration Entry is created.

Creation of new Group Registration Entries may be restricted by the Restricted_Group_Registration control (10.3.2.3). If this control is TRUE, creation of a new dynamic entry is permitted only if there is a Static Filtering Entry for the Group Address with a Registrar Administrative Control value of Normal Registration.

NOTE—Both Group Membership and Group service requirement information is recorded in the Filtering Database by means of Group Registration Entries (see 7.9.3). In the case of Group Membership Information, the MAC Address specification in the Group Registration Entry is a Group MAC Address. In the case of Group service requirement information, the MAC Address specification is either “All Group Addresses” or “All Unregistered Group Addresses.”

On receipt of a GID_Leave.indication whose attribute_type is equal to the value of the Group Attribute Type or the Service Requirement Attribute Type (10.3.1.3), the GMRP Application element specifies the Port associated with the GMRP Participant as Filtering in the Port Map field of the Group Registration Entry (7.9.3) for the MAC Address specification carried in the attribute_value parameter. If such a Filtering Database entry does not exist in the Filtering Database, then the indication is ignored. If setting that Port to Filtering results in there being no Ports in the Port Map specified as Forwarding (i.e., all Group members are de-registered), then that Group Registration Entry is removed from the Filtering Database.

10.3.2.3 Administrative controls

The provision of static control over the registration state of the state machines associated with the GMRP Application is achieved by means of the Registrar Administrative Control parameters associated with the operation of GARP (12.8.1). These parameters are represented in the Filtering Database by the information held in Static Filtering Entries (7.9.1). If no Static Filtering Entry exists for a given MAC Address specification, the value of the Registrar Administrative Control parameter for the corresponding attribute value is Normal Registration for all Ports of the Bridge.

The initial state of the Permanent Database (i.e., the state of the Permanent Database in a Bridge that has not been otherwise configured by management action) includes a Static Filtering Entry with a MAC Address specification of All Groups, in which the Port Map indicates Registration Fixed. This Static Filtering Entry will have the effect of determining the default Group filtering behavior of all Ports of the Bridge to be Forward All Groups. This Permanent Database entry may be deleted or updated by management action.

NOTE—This specification of an initial Static Filtering Entry means that operation using Forward Unregistered Groups or Filter Unregistered Groups requires a conscious action on the part of the network manager or administrator.

Where management capability is implemented, the Registrar Administrative Control parameters can be applied and modified by means of the management functionality defined in 14.7.

The provision of static control over the ability of Applicant state machines to participate in protocol exchanges is achieved by means of the Applicant Administrative Control parameters associated with the operation of GARP (12.8.2). Where management capability is implemented, the Applicant Administrative Control parameters can be applied and modified by means of the management functionality defined in 14.9.

Further administrative control over dynamic Group registration may be achieved, if supported, by means of a per-Port Restricted_Group_Registration control parameter. If the value of this control is TRUE for a given Port, the creation or modification of Dynamic Group Registration Entries as a result of GMRP exchanges on that Port shall be restricted only to those MAC addresses for which Static Filtering Entries exist in which the Registrar Administrative Control value is Normal Registration. If the value of the Restricted_Group_Registration control is FALSE, dynamic Group registration is not so restricted. Where management capability is implemented, the value of the Restricted_Group_Registration control can be manipulated by means of the management functionality defined in 14.10.1. If management of this parameter is not supported, the value of this parameter shall be FALSE for all Ports.

10.4 Conformance to GMRP

This subclause defines the conformance requirements for implementations claiming conformance to GMRP. Two cases are covered: implementation of GMRP in MAC Bridges and implementation of GMRP in end stations. Although this standard is principally concerned with defining the requirements for MAC Bridges, the conformance requirements for end station implementations of GMRP are included in order to give guidance to such implementations. The PICS Proforma defined in Annex A is concerned only with conformance claims with respect to MAC Bridges.

10.4.1 Conformance to GMRP in MAC Bridges

A MAC Bridge for which conformance to GMRP is claimed shall

a) Conform to the operation of the GARP Applicant and Registrar state machines, and the LeaveAll generation mechanism, as defined in 12.7.1, 12.7.2, and 12.7.3.

b) Exchange GARP PDUs as required by those state machines, formatted in accordance with the generic PDU format described in 12.10, and able to carry application-specific information as defined in 10.3.1, using the GMRP Address as defined in Table 12-1.
c) Propagate registration information in accordance with the operation of GIP for the Base Spanning Tree Context, as defined in 12.2.3 and 12.2.4.
d) Implement the GMRP Application component as defined in 10.3.

10.4.2 Conformance to GMRP in end stations

An end station for which conformance to GMRP is claimed shall

a) Conform to the operation of one of the following:
   1) the Applicant state machine, as defined in 12.7.1; or
   2) the Applicant Only state machine, as defined in 12.7.5; or
   3) the Simple Applicant state machine, as defined in 12.7.6.
b) Exchange GARP PDUs as required by the GARP state machine(s) implemented, formatted in accordance with the generic PDU format described in 12.10, and able to carry application-specific information as defined in 10.3.1, using the GMRP Application address as defined in Table 12-1.
c) Support the provision of end system registration and de-registration as defined in 10.3.2.1.
d) Discard MAC frames carrying any GARP Application address as the destination MAC Address in accordance with the requirements of 7.12.3.

An end station for which conformance to the operation of the Applicant state machine (12.7.1) is claimed shall also

e) Conform to the operation of the GARP Registrar state machine and the LeaveAll generation mechanism, as defined in 12.7.2 and 12.7.3; and
f) Support the provision of Group and Group service requirement registration and de-registration as defined in 10.3.2.2; and
g) Filter outgoing frames destined for group MAC Addresses in accordance with registered Group and Group service requirement information, in a manner consistent with the operation of the filtering function of the forwarding process described in 7.7.2.

It is recommended that only those end stations that require the ability to perform source pruning (10.2.3) conform to the operation of the Applicant state machine (12.7.1).

For the reasons stated in 12.6.9, it is recommended that end stations that do not require the ability to perform source pruning implement the Applicant Only state machine (12.7.5), in preference to the Simple Applicant state machine (12.7.6).

NOTE—End stations that implement only item a)2) and items b) through d) are equivalent to the description of the Applicant-Only Participant (12.6.7); those that implement only item a)3) and items b) through d) are equivalent to the description of the Simple-Applicant Participant (12.6.8). Such end stations require only the ability to register membership of one or more Groups, and revoke that membership at some later point in time; for this reason, there is no requirement to support the operation of the Registrar or Leave All state machines.

End stations that implement item a)1) and items b) through g) are able to perform “source pruning” as described in 10.2.3, i.e., to suppress the transmission of frames destined for Groups that currently have no membership. Consequently, such end stations need to support the full Applicant state machine, in combination with the Registrar and Leave All state machines.

11. Example “C” code implementation of GMRP

In IEEE Std 802.1D, 1998 Edition this clause provided an example implementation of the GMRP application (Clause 10). It contained no normative provisions and has been superseded by other ways of sharing implementation information.

12. Generic Attribute Registration Protocol (GARP)

The Generic Attribute Registration Protocol allows participants in a GARP Application to register attributes with other participants in a Bridged Local Area Network. The definition of attribute types, their values, and the semantics associated with values when registered, are specific to each GARP Application.

NOTE—Clause 10 defines a GARP Application, GMRP, which registers attributes of two types—group MAC Addresses and Group service requirements. Values of these attributes control group address filtering by GMRP participants.

12.1 GARP overview

GARP allows a participant in a given GARP Application to make or withdraw *declarations* of *attributes*, and for those declarations (or withdrawals) to result in the *registration* (or removal of registrations) of those attributes with the other GARP Participants for that Application.

A declaration by a GARP Participant for an end station or Bridge Port is recorded by an Applicant state machine for the declared attribute and Port. Changes in the Applicant state machine's variables trigger the transmission of a GARP PDU to communicate the declaration (or withdrawal).

A registration is recorded by a Registrar state machine for the attribute at each participating end station and Bridge Port that receives the GARP PDU. Removal of a given attribute registration occurs only if all the other participants connected to the same LAN withdraw the declaration.

Attributes registered on Bridge Ports that are part of the applicable *active topology* (7.4, 12.2.4) are declared on all the other Bridge Ports that are also part of that active topology. Hence, a given declaration is propagated to all application participants, and registered in each Bridge on those Ports that are "nearest" to the source or sources of the declaration.

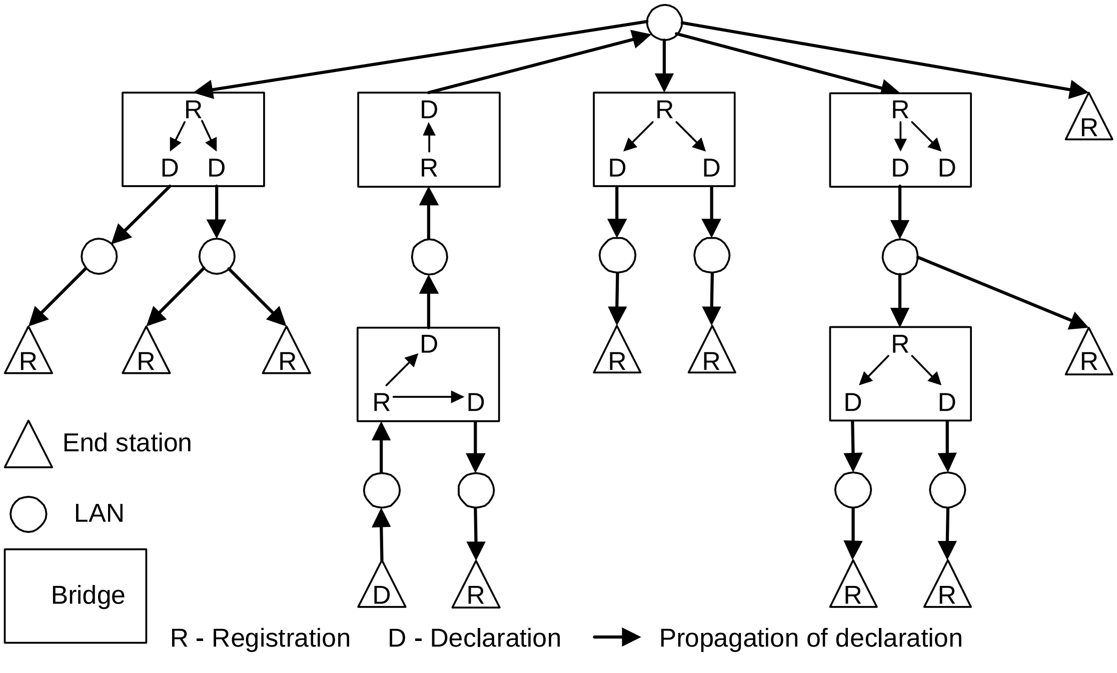


Figure 12-1—Example—Attribute value propagation from one station

NOTE—Unless otherwise stated, the following description assumes operation within the Base Spanning Tree Context. While registration can occur on any Bridge Port, regardless of Port State (7.4), propagation follows the spanning tree active topology. All the Bridge Ports shown in Figure 12-1, Figure 12-2, and Figure 12-3 are in the Forwarding Port State.

Figure 12-1 illustrates the result of a single end station making a declaration, and shows the Bridge Ports that also make declarations to propagate the attribute. The attribute is propagated to all LANs in the Bridged Local Area Network, but the directional nature of the propagation results in registration only on Bridge Ports that receive (as opposed to transmit) declarations.

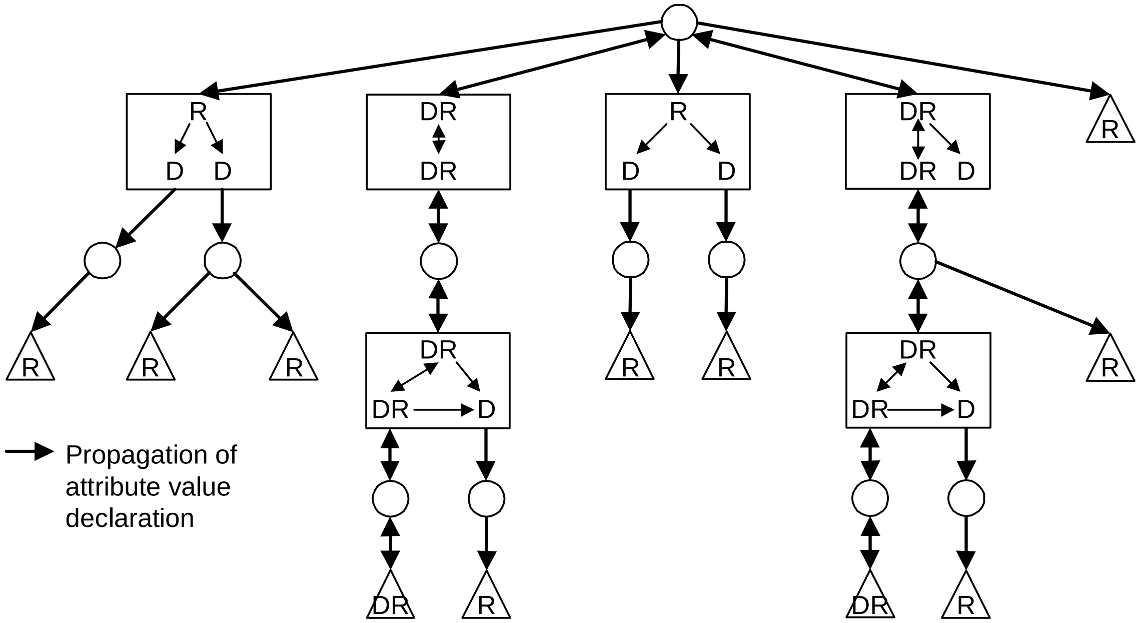


Figure 12-2—Example —Attribute value propagation from two stations

Figure 12-2 illustrates the result of different end stations declaring the same attribute on different LANs. All end stations register the attribute, and some Bridges register it on more than one Port.

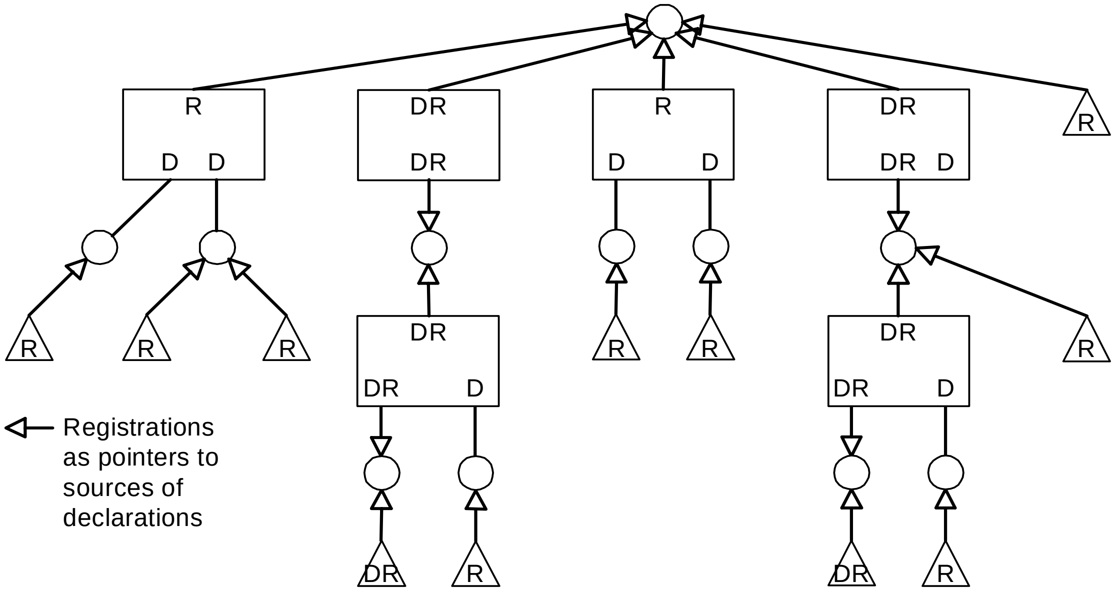


Figure 12-3—Example — Registrations as pointers to the sources of declarations

The set of Bridge Ports and end stations that both declare and register a given attribute defines the subset of the active topology that contains all the participants declaring that attribute. A registration can be regarded as a pointer to participants that have declared that attribute, as illustrated in Figure 12-3 for the declarations and registrations in Figure 12-2.

Applications where it is desirable to form “reachability” trees are generally good candidates for the use of GARP. For example, if the attribute in Figure 12-3 is a Group MAC Address that carries the semantics “I wish to receive details of the final score in the Superbowl,” and it is deemed desirable for those results to be sent only to the subset of the active topology that contains end stations that have declared that attribute, then an end station that has these results available could use the presence or absence of a registration as an indication of whether or not to send the results on the LAN to which it is attached, and any Bridge receiving the results could determine on which Ports the results should be forwarded.

In MAC Bridges, GARP operates only when the Bridge Filtering Mode is set to Extended Filtering Mode. Bridges that are unable to operate in Extended Filtering Mode, or have been set to operate in Basic Filtering Mode, are transparent with respect to GARP protocol exchanges, and forward GARP PDUs on all Ports that are in Forwarding. Similarly, Bridges that do not implement a given GARP Application are transparent to GARP protocol exchanges destined for that Application.

GARP operates only on Ports that are MAC_Operational (6.4.2). If the Port is operating as a network access port (IEEE Std 802.1X), GARP uses the controlled port (7.12.7). On any Port whose MAC_Operational parameter is FALSE or whose AuthControlledPortStatus is not Authorized, any GARP entity shall not transmit GARP PDUs, and shall discard, without processing, any received GARP PDUs.

12.2 GARP architecture

Figure 12-4 illustrates the components of GARP Participants in a two-Port Bridge and an end station.

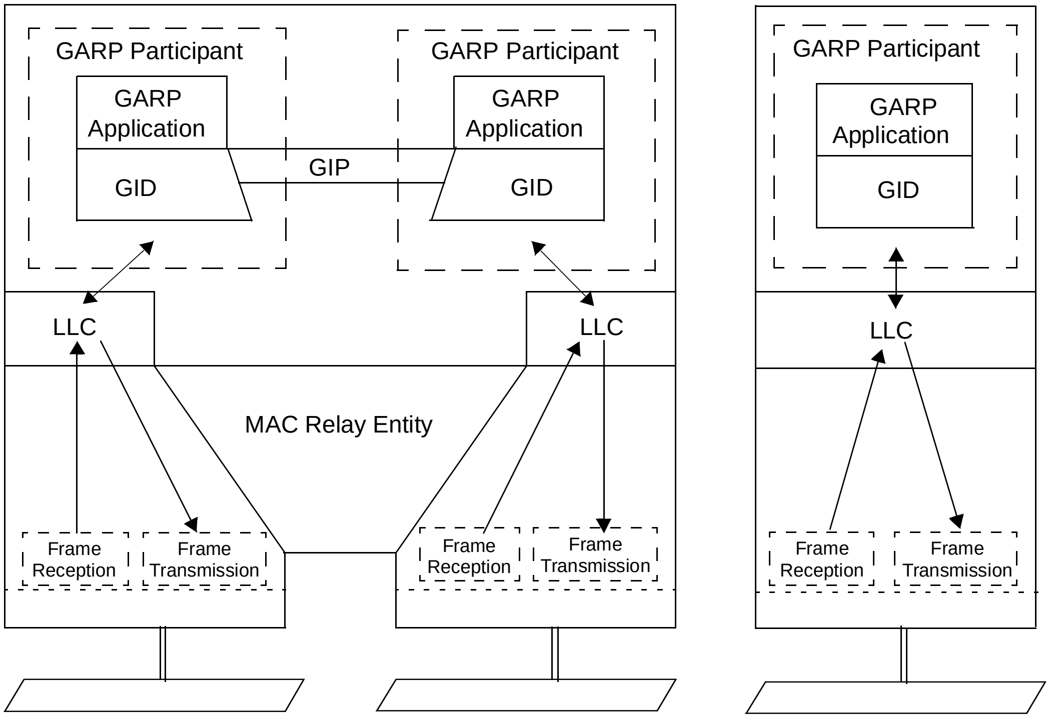


Figure 12-4—GARP architecture

A *GARP Participant* in a Bridge or an end station consists of a *GARP Application* component, and a *GARP Information Declaration (GID)* component associated with each Port of the Bridge. One such GARP Participant exists per Port, per GARP Application. The propagation of information between GARP

Participants for the same Application in a Bridge is carried out by the *GARP Information Propagation (GIP)* component. Protocol exchanges take place between GARP Participants by means of LLC Type 1 services, using the group MAC Address and PDU format defined for the GARP Application concerned.

12.2.1 GARP Applications

For each GARP Application, the following are defined:

a) A set of Attribute types used by the Application.
b) The Attribute values permitted for each Attribute type.
c) The semantics associated with each Attribute type and value.
d) The group MAC Address used to exchange GARP PDUs between Application Participants.
e) The structure and encoding of the Attribute types and values in GARP PDUs.
f) The requirements for GARP state machine support in end stations and Bridges.

The GARP Application component of the GARP Participant is responsible for defining the semantics associated with parameter values and operators received in GARP PDUs, and for generating GARP PDUs for transmission. The Application component makes use of the GID component, and the state machines associated with GID's operation, to control its protocol interactions. The service offered to the Application component by the GID component is defined by the set of primitives described in 12.2.2.

12.2.2 GID

An instance of GID consists of the set of state machines that define the current registration and declaration state of all attribute values associated with the GARP Participant of which the GID instance is a component. Figure 12-5 illustrates the set of state machines associated with a GID instance.

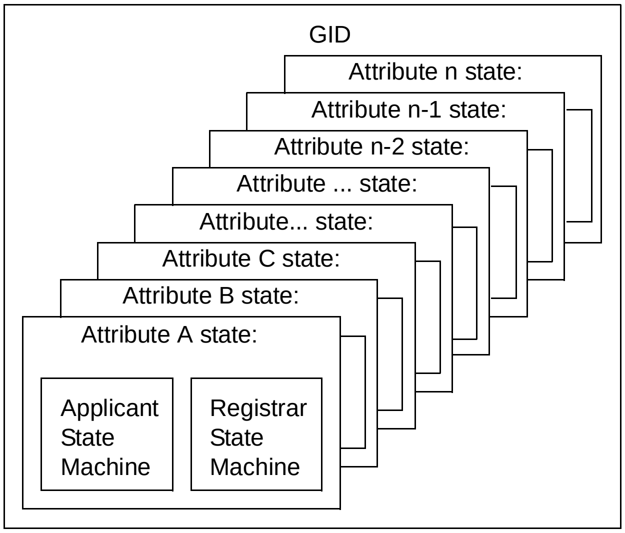


Figure 12-5—GID architecture

The operation of GID is defined by the following:

a) The Applicant State Transition Table (Table 12-3).
b) The Registrar State Transition Table (Table 12-4).
c) The Applicant and Registrar state machines that represent the declaration and registration state for each attribute value.
d) The service primitives available to users of GID.

12.2.2.1 Declarations

The following two primitives allow applications to request GID to make (Join) or withdraw (Leave) Attribute declarations:

GID_Join.request (attribute_type, attribute_value)

GID_Leave.request (attribute_type, attribute_value)

GID uses the current Applicant state for the attribute, and the action and subsequent state defined for that state and event in the Applicant State Transition Table (Table 12-3), to determine the action taken on receipt of a primitive and the subsequent Applicant state.

GID requests can be generated by both the Application and the Information Propagation components.

12.2.2.2 Registration

The following two primitives are defined to allow GID to notify an Application that a given Attribute has been registered (Join) or de-registered (Leave) on a given Port, as a result of protocol on the attached LAN:

GID_Join.indication (attribute_type, attribute_value)

GID_Leave.indication (attribute_type, attribute_value)

GID uses the current Registrar state for the attribute, and the action and subsequent state defined for that state and event in the Registrar State Transition Table (Table 12-4), to determine the action taken on receipt of registration and de-registration information contained in GARP PDUs and the subsequent Registrar state.

GID indications are received by both the Application and the Information Propagation components.

12.2.3 GIP

The GARP Information Propagation (GIP) function operates in the same way for all GARP applications, and enables propagation of attributes registered on Bridge Ports across the network to other participants.

For a given GARP Application and GIP Context (12.2.4), and for the set of Ports that are in a Forwarding state as defined by that GIP Context

a) Any GID_Join.indication received by GIP from a given Port in the set is propagated as a GID_Join.request to the instance(s) of GID associated with each other Port in the set.
b) Any GID_Leave.indication received by GIP from a given Port in the set is propagated as a GID_Leave.request to the instance(s) of GID associated with each other Port in the set (Port P, say) if and only if no registration now exists for that Attribute on any other Port in the set excluding P.

These rules propagate attribute registrations through any given Port if any other Port has seen a registration for that attribute, and propagate de-registrations if all other Ports are now de-registered.

As the set of Ports that are in a Forwarding state for a given GIP Context can change dynamically, for example as a result of Spanning Tree reconfiguration, GIP operates as follows after such a change:

c) If a Port is added to the set, and that Port has registered an attribute (i.e., a GID_Join.indication has occurred more recently than any GID_Leave.indication for the attribute), then GID_Join.requests are propagated to the GID instances for each of the other Ports in the set.

d) If a Port is removed from the set, and that Port has registered an attribute and no other Port has, then GID_Leave.requests are propagated to the GID instances for each of the other Ports in the set.

12.2.4 GARP Information Propagation Context

For a given Port of a GARP-aware Bridge and GARP Application supported by that Bridge, an instance of a GARP Participant can exist for each *GARP Information Propagation Context (GIP Context)* understood by the Bridge. A GIP Context identifies the set of Bridge Ports that form the applicable active topology (7.4).

An example is the active topology formed by the operation of RSTP (Clause 17). This GIP Context provides the same connectivity as the Spanning Tree in each Bridge and is known as the *Base Spanning Tree Context*.

NOTE—This standard uses only the Base Spanning Tree Context to define the operation of GMRP; however, other GIP Contexts may be used for other Applications, or for extending GMRP functionality. In particular, this flexibility allows the use of GARP Applications where an active topology can be defined by a multiple Spanning Trees instance or a Virtual LAN (VLAN). VLANs are defined in IEEE Std 802.1Q.

GARP protocol exchanges can occur on all of the Ports of a Bridge; however propagation across a Bridged Local Area Network of attribute registrations for a given application uses only the Bridge Ports in the active topology identified by the GIP Context. Each GARP Application specification identifies the contexts it can operate within, specifies rules for selecting forwarding Ports, and assigns GIP Context identifiers for use in conjunction with the operation of GARP and its administrative controls (12.8).

A GIP Context identifier of 0 identifies the *Base Spanning Tree Context*. The GARP Application specifies how the GIP Context of each GARP PDUs is identified if any other context is used.

12.3 Requirements to be met by GARP

GARP establishes, maintains, withdraws, and disseminates attribute declarations and registrations amongst the GARP Participants attached to a Bridged Local Area Network. The protocol meets the following requirements for Applicant and Registrar behavior, error recovery, performance, scalability, compatibility with non-GARP aware devices, and the load imposed on Bridges, end stations, and the network:

a) Participants can issue declarations for GARP application attributes (12.2.2, 12.7.1, and 12.9).
b) Participants can withdraw declarations for attributes (12.2.2, 12.7.1, and 12.9).
c) Each Bridge propagates declarations to GARP Participants (12.2.3).
d) GARP Participants can track the current state of declaration and registration of attributes on each Port of the participant device (12.7.1, and 12.7.2).
e) GARP Participants can remove state information relating to attributes that are no longer active within part or all of the network, e.g., as a result of the failure of a participant (12.7.2, and 12.7.3).
f) The latency involved in issuing, propagating, or revoking attribute declarations, is small (i.e., comparable to the frame propagation delay) and increases linearly as a function of the diameter of the network (12.7.1, 12.9, 12.7.2, and 12.7.3).
g) GARP is resilient in the face of the failure of GARP Participants (H.4).
h) GARP is resilient in the face of single packet loss.
i) GARP will operate correctly in networks where:
    1) All Bridges support both Basic and Extended Filtering Services; or where
    2) Some Bridges support only Basic Filtering Services and some both Basic and Extended Filtering Services (12.4, 12.7.1, 12.9, 12.7.2, 12.7.3, and I.1).
j) The communications bandwidth consumed on any particular LAN by Applicants and Registrars in exchanging GARP PDUs will be a small percentage of the total available bandwidth, and independent of the total traffic supported by the network. The bandwidth consumed will be a function of the number of attributes registered.

12.4 Requirements for interoperability between GARP Participants

T o ensure the interoperability of GARP, the following are required:

a) Each GARP Application uses a unique group MAC Address as the destination address of GARP PDUs. Table 12-1 specifies group MAC Addresses allocated to existing applications and reserved for future standardized applications. A Bridge that implements the application corresponding to a given entry in that table shall not forward frames destined for that MAC Address; a Bridge that does not implement the application shall forward frames destined for that MAC Address received on any Port that is part of the active topology to all other Ports that are part of the active topology.
b) The transmission and reception of GARP PDUs between GARP Participants, formatted as defined for the application using the generic PDU format defined in 12.10, shall use LLC Type 1 procedures. The standard Bridge Spanning Tree Protocol LLC Address in Table 7-9 shall be used as the source and destination LLC address.
c) GARP PDUs, i.e., frames with the destination MAC and LLC addresses specified in item a) and item b), that are not well formed (i.e., are not structured and encoded as defined in 12.10 and with attribute types and values encoded as defined by the GARP Application) shall be discarded on receipt.

Table 12-1—GARP Application addresses

Assignment	Value
GMRP Address (See Clause 10)	01-80-C2-00-00-20
GVRP Address (See IEEE Std 802.1Q)	01-80-C2-00-00-21
Reserved	01-80-C2-00-00-22
Reserved	01-80-C2-00-00-23
Reserved	01-80-C2-00-00-24
Reserved	01-80-C2-00-00-25
Reserved	01-80-C2-00-00-26
Reserved	01-80-C2-00-00-27
Reserved	01-80-C2-00-00-28
Reserved	01-80-C2-00-00-29
Reserved	01-80-C2-00-00-2A
Reserved	01-80-C2-00-00-2B
Reserved	01-80-C2-00-00-2C
Reserved	01-80-C2-00-00-2D
Reserved	01-80-C2-00-00-2E
Reserved	01-80-C2-00-00-2F

NOTE—GARP uses the same LLC Address as RSTP. Distinct MAC Addresses and protocol identifiers ensure that received PDUs can be delivered to the appropriate protocol entities. PDUs with a destination MAC Address equal to the Bridge Group Address identified in Table 7-10, the LLC address assigned in Table 7-9, and the Spanning Tree protocol identifier defined in Clause 9, are processed by RSTP (see 7.12.3). PDUs with a destination MAC Address equal to any of the GARP Application addresses in Table 12-1, the LLC address assigned in Table 7-9, and the GARP protocol identifier as defined in 12.10, are handled as described in item a) through item c) above.

12.5 Conformance to GARP Applications

The specification of GARP in this subclause defines behavior generic to GARP Applications. Conformance to GARP is defined with reference to a specific application or applications. Each application specifies the

GARP functionality required to claim conformance to that application, as stated in 12.2.1. A conformant implementation of GARP is therefore an implementation that supports at least the GARP functionality required to claim conformance to the set of GARP Applications supported by that implementation.

12.6 Protocol Operation

This clause provides an informal introduction. The definitive description of GARP is contained in the State Machine Descriptions (12.7), Procedures (12.9), and Encoding of GARP Protocol Data Units (12.10).

12.6.1 Basic notions

The basic notions behind the operation of GARP are that

a) A simple, fully distributed many-to-many protocol is possible. There is no need for an additional election protocol to change the problem to allow a many-to-one design.
b) The protocol should be resilient against the loss of a single message, in a set of related messages, but does not need to be stronger.
c) A Participant that wishes to make a declaration (an Applicant) sends Join messages.
d) If an Applicant sees other Participants sending two Join messages, it does not need to send a Join message itself in order to participate in the declaration concerned.
e) An Applicant that wishes to withdraw a declaration need only send a single Leave message; it can then forget all about the registration concerned. There is no need for it to confirm that de-registration has taken place, as other Participants may wish to maintain the registration themselves.
f) Missing or spuriously continued registrations that arise from lost messages are cleared up by a periodic mechanism that sends LeaveAll messages. A LeaveAll message declares that all registrations will shortly be terminated unless one or more Participants declares a continuing interest in specific registrations by issuing further Joins.

To guard against the possibility of a participant missing a Leave message, causing another participant’s Registrar to think there are no remaining declarations for an attribute value, one additional mechanism is necessary. If a participant receives a Leave message, and no subsequent Joins, it sends a further message to prompt rejoining before revoking the registration.

12.6.2 GARP Messages

The description so far introduces three basic message types used in GARP—Join, Leave, and LeaveAll. However, making do with only these three message types would add to the eventual complexity of the protocol.

Consider two GARP Participants attached to the same LAN. The fact that one (Andy, for example) sends the other (Bill) two Join messages says nothing about Andy’s knowledge of Bill’s wish to make that declaration. Would-be Applicants can also need to know if there are other Applicants making the same declaration, as do Bridge Ports. Attempts to work around the problem by setting join timers such that no single Participant can send two messages in an interval within which two or more Participants can be expected to send messages results in timer dependencies that make determining correctness of the protocol operation difficult.

Consider the sending of a second Leave by a Registrar to prompt a rejoin. If a Join is just being sent, the protocol now depends on the second Join not being lost. The protocol depends on the relative values of the Registrar’s leave timer and other Participants’ join timers.

The protocol is therefore based on the general design principle that participants communicate their current state, rather than send directions. The following four attribute value-specific message types are used:

a) Empty—I am not trying to declare this Attribute value. I have not registered this Attribute value, but I care if there are any Participants that wish to declare it.
b) JoinEmpty—I wish to declare this Attribute value. I have not registered this Attribute value, but I care if there are any Participants that wish to declare it.
c) JoinIn—I wish to declare this Attribute value. I have either registered this Attribute value, or I do not care if there are any other Participants that wish to declare it (I will behave as if there are).
d) Leave—I had registered this Attribute value, but am now in the process of de-registering it.

As before, the following garbage collection message is also used:

e) LeaveAll—All registrations will shortly be de-registered; if any Participants have a continuing interest in any of the registrations they need to rejoin in order to maintain the registration.

In theory there could be LeaveIn and LeaveEmpty variants of the Leave message; these are coded in the GARP PDUs to provide maximum visibility into what implementations are doing, and to avoid missing or illegal codes. However, it will be seen that the state machines treat these two message variants identically.

There is no reason to send a simple In message, i.e., one that means "I do not wish to make this declaration but have registered the Attribute value on behalf of other Participants (or will behave as if there are other Participants that have made the declaration)."

The protocol makes good use of the distinction between JoinEmpty and JoinIn messages, and between Leave and Empty.

The JoinIn message meets the requirements for Join message suppression. If an Applicant sees a JoinIn message it can avoid sending a Join itself for that declaration, as it knows that both the recipient(s) and the transmitter of the JoinIn believe there are Participants that have made the declaration. The JoinIn is not treated as an acknowledgment, because on a shared media LAN, there are potentially many Participants who need to register the Attribute value. Moreover, Participants who don't care whether there are other Participants interested in that registration or not can always send JoinIns instead of JoinEmptys. However, on the assumption that only one JoinIn message is lost, two suffice to ensure that all Registrars have registered the group, to a high probability.

The Leave message will cause its recipients to de-register membership, while the JoinEmpty and Empty messages will just prompt them to rejoin, so JoinEmpty and Empty messages can be used at any time to prompt for rejoin without throwing recently joined members out again.

12.6.3 Applicant and Registrar

Each GARP participant maintains a single Leave All protocol component (12.6.6). It also maintains two protocol components per Attribute that it is interested in—an Applicant and a Registrar.

The job of the Registrar is to record attribute value registrations declared by the other participants on the LAN. It does not send any protocol messages.

The job of the Applicant is twofold:

a) To ensure that this Participant's declarations are registered by other Participants' Registrars—if it wants to maintain those registrations.
b) To ensure that other Participants have a chance to re-declare (rejoin), after anyone withdraws a declaration (leaves)—if there are any Participants that want to maintain the registration.

NOTE—Item b) above applies only to the behavior of the full Applicant state machine (12.6.5); the Applicant Only and Simple Applicant state machines (12.6.7 and 12.6.8) are concerned only with item a).

The Applicant is therefore looking after the interests of all would-be Participants. This allows the Registrar to be very simple.

12.6.4 Registrar behavior

The Registrar has a single timer, the leave timer, and the following three states:

a) IN—I have registered the fact that this Attribute value has been declared on this LAN.
b) MT—(Empty) All declarations for this Attribute value on this LAN have been withdrawn.
c) LV—I had registered this Attribute value, but am now timing out the registration (using the leave timer). If I do not see a declaration for this Attribute before leave timer expires, I will become MT.

The Registrar reacts to received messages as follows:

d) A Join message, either JoinIn or JoinEmpty, causes the Registrar to become IN (I have registered the Attribute).
e) If the Registrar was IN, then a Leave or LeaveAll causes it to become LV (I am timing out the registration) and starts the leave timer. Otherwise (LV or MT) there is no effect.
f) An Empty message (someone else has no registration for this Attribute) has no effect [see 12.6.2 a)].

While the Registrar does not send messages, it affects the type of Join message sent by the Applicant. If the Registrar is IN, a JoinIn is sent; otherwise, a JoinEmpty is sent.

12.6.5 Applicant behavior

Against the background of this simple Registrar, the next consideration is the behavior of the Applicant that wishes to make a declaration, starting from a point where it has neither seen nor sent any messages.

If no messages were ever lost, the Applicant could either send a Join or receive a JoinIn, and then be content that all Registrars would have registered its declaration. On the single message loss assumption it needs to send two Joins, or receive two JoinIns, or send one Join and receive one JoinIn (in either order). This part of its state could be recorded in a simple counter:

my_membership_msgs = 0, 1, or 2

which is incremented for every Join sent or JoinIn received. If the counter value is 0 or 1 when there is an opportunity to transmit a PDU, a Join message will be sent and the counter incremented.

NOTE 1—A counter value of greater than 2 is unnecessary for the purposes of successful registration.
NOTE 2—A randomized Join timer is set running to ensure such an opportunity is scheduled. There only needs to be one Join timer running for the entire Participant, not one per attribute—assuming that messages related to the maximum number of attributes can be packed into a single PDU.

If a JoinEmpty, Empty, Leave, or LeaveAll message is received, the counter is reset to 0.

When the Applicant leaves the Group, it sends a single Leave message.

12.6.5.1 Anxious Applicants

Expressing protocol behavior in terms of counter and flag variables is not always the best approach if enabling thorough analysis and maximizing implementation flexibility are primary goals. From this point on, the values assigned to the join message count are given the following state name prefixes:

a) V or Very anxious equates to my_join_msgs = 0. No Join messages have been sent, and no JoinIns received since the Applicant started, or leave or empty messages received. The Applicant has no reason to be comfortable that other Registrars have registered the Attribute value concerned.
b) A or Anxious equates to my_join_msgs = 1. If no messages have been lost, other Registrars will have registered this Attribute value.
c) Q or Quiet equates to my_join_msgs = 2. The Applicant feels no need to send further messages.

12.6.5.2 Members and Observers

The Applicant described so far needs have no existence unless it is trying to make a declaration. Bridge Ports and end stations that make active use of registered Attribute values (e.g., in the case of the GMRP Application defined in Clause 10, Bridges and end stations that implement source pruning for transmission), need to maintain their GARP machines even if they do not want to make (or have just withdrawn) a declaration. (The term GARP machine refers to the total state maintained for a given Attribute value, both Applicant and Registrar, in a Participant.)

In the context of the Applicant state machine, a Member is a Participant that is attempting to make or maintain a declaration for a given Attribute value, or that has not yet sent the Leave message to allow it to become simply an Observer. An Observer tracks the Attribute state but does not wish to make a declaration.

12.6.5.3 Active and Passive Members

The concept of Active and Passive Members is introduced to permit the minimum number of messages to be sent when a number of Participants are actively joining and leaving with respect to the same registration.

Since a Member may become Quiet without ever sending a Join, it follows that it should be allowed to become an Observer once more without sending a Leave. All Observers are passive, so there are three potential (sub)states, distinguished by the following state name suffixes:

a) A, or Active member.
b) P, or Passive member.
c) O, or Observer.

If an Observer is required to become a Member, it first becomes a Passive Member. If it was a Quiet Observer (i.e., its count of Join messages is already at two and it is therefore content that other Registrars have registered the declaration), then it has no need to transmit a Join, and becomes Passive and Quiet. Otherwise, i.e., its count of Join messages is less than two, it requests the earliest possible message transmission opportunity in order to transmit a Join.

If a Passive Member sends a Join message, it becomes an Active Member.

If an Active Member receives a Leave or LeaveAll message, it becomes a Passive Member.

12.6.5.4 Receiving a Leave

When an Applicant that is, and wishes to continue being, a Member receives a Leave Message, it becomes Very Anxious. Unless it receives a Join message from another Member, it will send a JoinEmpty itself. This has the following effect on other Members. First, it will cause them to register that Attribute. Second, it will cause them to become Very Anxious themselves if they wish to continue to be Members, and to transmit JoinIns.

This latter effect protects any Participant that is a Member from accidentally de-registering other Members due to a single packet loss following a Leave.

An Applicant that is an Observer has to prompt other Members to re-join in case they have missed the Leave. A further (sub) state is added to the Very Anxious, Anxious, Quiet set with state name prefix L or Leaving, which records the pending need to send a message at the next transmission opportunity. An Observer will send an Empty message, and then become Very Anxious.

12.6.5.5 Leaving

An Active Member has to send a Leave message in order to withdraw a declaration. The Leaving substate is used to record that fact.

12.6.5.6 Applicant State Summary

The following matrix summarizes the Applicant states and their short names: VA for Very Anxious Active member, QO for Quiet Observer, etc.

Table 12-2—Applicant: Summary of states

	Very Anxious	**Anxious**	**Quiet**	**Leaving**
Active Member	VA	AA	QA	LA
Passive Member	VP	AP	QP	
Observer	VO	AO	QO	LO

Note that there is no Leaving Passive Member (LP) state, since a Passive Member can transition directly to an Observer state when it wishes to withdraw a declaration.

12.6.6 The Leave All protocol component

The Leave All protocol component is responsible for initiating garbage collection by the Participant. This is achieved by the regular generation of LeaveAll messages by the LeaveAll state machine, as defined in 12.7.3.

The operation of the Leave All state machine causes the Participant to generate a LeaveAll message when the leaveAllTimer expires. Reception of a Leave All message from another Participant causes the timer to be restarted without generating a message, thus ensuring that, where several Participants are connected to the same LAN, multiple LeaveAll messages are suppressed.

Receipt of a LeaveAll message causes all Applicants to become Very Anxious, and all Registrars to enter the Leaving (LV) state. The effect of this is to force any Applicants that are still active to rejoin; if no Join message is seen by a Registrar state machine before its leave timer expires, it will de-register the attribute. This effectively causes the Participant to remove all registrations for which active Applicants no longer exist.

12.6.7 Applicant-Only Participants

It is possible to simplify the GARP Participant in circumstances where the Participant only wishes to make declarations; for example, as might be required in an end station implementing the ability to register to receive group addressed frames, via GMRP (Clause 10), but that does not operate as a source of such frames, and therefore does not need to implement source pruning. Such a Participant has no need to take note of declarations made by other Participants (i.e., it does not need to implement the Registrar state machine), does not send Leave All or Join Empty messages, and offers no additional administrative controls.

This leads to a potential simplification of the state machinery that such a participant would need to support, as follows:

a) No requirement to support a LeaveAll timer, or the generation of LeaveAll PDUs.
b) No requirement to support the operation of the Registrar state machine. The “IN” state for the Registrar is assumed for the purposes of the operation of the Applicant state machine, hence Join messages are always sent as JoinIn.
c) No requirement for the state machine to support the LO state or generate Empty messages.
d) No requirement to support the administrative controls defined in 12.8.

The operation of the Applicant Only state machine is described in 12.7.5 and Table 12-8.

12.6.8 Simple-Applicant Participants

The operation of the Simple-Applicant Participant is a further simplification of the Applicant Only state machine (Table 12-8), modified to reduce the Applicant to its simplest form, achieved by removing the Passive Member and Observer states. Consequently, the Simple-Applicant Participant makes no attempt to suppress its initial Join and final Leave messages. The result is the simplest possible Applicant state machine that is compatible with the operation of the full GARP Participant.

The operation of the Simple Applicant state machine is described in 12.7.6 and Table 12-9.

12.6.9 Choice of Applicant-Only Participant or Simple-Applicant Participant

The fact that the Applicant-Only Participant retains the Passive Member and Observer states of the full Applicant means that it retains the ability of the full Participant to suppress Join or Leave messages when these are unnecessary (see 12.6.5.3); in contrast, the Simple-Applicant Participant is unable to perform such suppression. Where there is the possibility of several Simple-Applicant Participants appearing on the same LAN, this could result in significant additional, and unnecessary, Join and Leave traffic. Consequently, it is recommended that the Applicant-Only Participant is implemented in preference to the Simple-Applicant Participant in devices that have no need to perform registration.

12.6.10 Use of GARP in point-to-point LANs

The full GARP participant state machine was designed to operate correctly in shared media environments. The fact that in such environments there might be three or more active GARP participants was the motivation behind the Passive Member and Observer states; the result is reduced traffic on a shared media LAN where more than one participant is interested in registering a particular attribute value.

In point-to-point LANs, i.e., where it is certain that there can at most be only two GARP participants on a LAN, the added complexity of the full Applicant state machine is redundant. In such environments, the Simple Applicant state machine should be implemented rather than the full Applicant state machine. The Point-to-Point MAC parameters (see 6.4.3) provide a means of determining whether a given MAC supports a shared media LAN or a point-to-point LAN.

The method of determining the timing of transmission opportunities in the state machines (the use of a timer value randomized between 0 and JoinTime seconds) was also chosen to accommodate the use of GARP in shared media environments, to avoid the risk of multicast storms occurring during periods of configuration change. In point-to-point LANs, these considerations do not apply. More particularly, in LANs that support RSTP, it is desirable to allow transmission opportunities to occur without delay under circumstances where a Rapid Spanning Tree configuration change is occurring, in order to minimize the period during which denial of service might occur due to the delay in propagating registration changes. Therefore, in point-to-point LANs, it is recommended that the definition of when a transmission opportunity (transmitPDU!—see 12.8) can occur is redefined as follows:

transmitPDU!	An opportunity to transmit a GARP PDU has occurred. A maximum transmission rate is imposed of no more than three such transmission opportunities in any period of 1.5*JoinTime seconds.

12.7 State machine descriptions

The following conventions are used in the abbreviations used in this subclause:

rXXX	receive PDU XXX
sXXX	send PDU XXX
ReqXXX	GID service request XXX
IndXXX	GID service indication XXX

The following abbreviations are used in the state machine descriptions. For a formal definition of their meaning, see 12.9:

Initialize	state machine (re)initialization event.
rJoinIn	receive Join In message
rJoinEmpty	receive Join Empty message
rEmpty	receive Empty message
rLeaveIn	receive Leave In message
rLeaveEmpty	receive Leave Empty message
LeaveAll	receive or send a LeaveAll message
sJ[E,I]	send Join In message if Registrar state = IN; otherwise, send Join Empty message
sJ[I]	send Join In message
sE	send Empty message
sLE	send Leave Empty message
sLeaveAll	send Leave All message
ReqJoin	GID Service Request to declare an Attribute value
ReqLeave	GID Service Request to withdraw an Attribute value declaration
IndJoin	Issue GID Service Indication signaling an Attribute value has been registered
IndLeave	Issue GID Service Indication signaling an Attribute value has been de-registered
transmitPDU!	An opportunity to transmit a GARP PDU has occurred. Such events occur at intervals randomly chosen in the interval 0 - JoinTime. JoinTime is as defined in Table 12-10
leavetimer	Leave period timer
leavetimer!	leavetimer has expired
leavealltimer	Leave All period timer.
leavealltimer!	leavealltimer has expired.
-x-	Inapplicable event/state combination. No action or state transition occurs.

Timers are used in the state machine descriptions in order to cause actions to be taken after defined time periods have elapsed. The following terminology is used in the state machine descriptions to define timer states and the actions that can be performed upon them:

a) A timer is said to be *running* if the most recent action to be performed upon it was a *start* or a *restart*.

b) A running timer is said to have *expired* when the time period associated with the timer has elapsed since the most recent start or restart action took place.

c) A timer is said to be *stopped* if it has expired or if the most recent action to be performed upon it was a *stop* action.

d) A *start* action sets a stopped timer to the running state, and associates a time period with the timer. This time period supersedes any periods that might have been associated with the timer by previous start events.

e) A *restart* action stops a running timer and then performs a start action upon it.

f) A *stop* action sets a timer to the stopped state.

The following abbreviations are used for the state names in the state tables and state diagrams:

Registrar states

LV	Leaving
IN	In
MT	Empty

Applicant states

VA	Very anxious, active member
AA	Anxious, active member
QA	Quiet, active member
LA	Leaving, active member
VP	Very anxious, passive member
AP	Anxious, passive member
QP	Quiet, passive member
VO	Very anxious, observer
AO	Anxious, observer
QO	Quiet, observer
LO	Leaving, observer

Simple Applicant states

V	Very anxious
A	Anxious
Q	Quiet

12.7.1 Applicant state machine

A full GARP Participant maintains a single instance of this state machine for each Attribute value for which the Participant needs to maintain state information.

NOTE—Conceptually, state information is maintained for all possible values of all Attribute types that are defined for a given Application; however, in real implementations of GARP, it is likely that the range of possible Attribute values in some Applications will preclude this, and the implementation will limit the state to those Attribute values in which the Participant has an immediate interest, either as a Member or as a likely future Member.

The detailed operation of this state machine is described in Table 12-3. The state transitions shown handle the possibilities of receiving either Leave In or Leave Empty messages; however, only Leave Empty messages are generated by the state machine. Sending a Leave Empty message also causes an event against the Registrar state machine, causing a transition from IN to LV.

The initial state for the Applicant state machine on (re)initialization is VO.

Table 12-3—Applicant state table

		STATE										
		VA	**AA**	**QA**	**LA**	**VP**	**AP**	**QP**	**VO**	**AO**	**QO**	**LO**
EVENT	**transmitPDU!**	sJ[E,I] AA	sJ[E,I] QA	-x-	sLE VO	sJ[E,I] AA	sJ[E,I] QA	-x-	-x-	-x-	-x-	sE VO
	rJoinIn	AA	QA	QA	LA	AP	QP	QP	AO	QO	QO	AO
	rJoinEmpty	VA	VA	VA	VO	VP	VP	VP	VO	VO	VO	VO
	rEmpty	VA	VA	VA	LA	VP	VP	VP	VO	VO	VO	VO
	rLeaveIn	VA	VA	VP	LA	VP	VP	VP	LO	LO	LO	VO
	rLeaveEmpty	VP	VP	VP	VO	VP	VP	VP	LO	LO	LO	VO
	LeaveAll	VP	VP	VP	VO	VP	VP	VP	LO	LO	LO	VO
	ReqJoin	-x-	-x-	-x-	VA	-x-	-x-	-x-	VP	AP	QP	VP
	ReqLeave	LA	LA	LA	-x-	VO	AO	QO	-x-	-x-	-x-	-x-
	Initialize	VO	VO	VO	VO	VO	VO	VO	VO	VO	VO	VO

12.7.2 Registrar state machine

A full GARP Participant maintains a single instance of this state machine for each Attribute value that is currently registered, or that the Registrar state machine is in the process of de-registering.

NOTE—As with the Applicant, state information is conceptually maintained for all possible values of all Attribute types that are defined for a given Application; however, in real implementations of GARP, it is likely that the range of possible Attribute values in some Applications will preclude this, and the implementation will limit the state to those Attribute values in which the Participant has an immediate interest. In the case of simple devices that have no interest in what other Participants have registered, it may be appropriate for that device to ignore Registrar operation altogether.

The detailed operation of this state machine is described in Table 12-4.

The initial state for the Registrar state machine on (re)initialization is MT.

12.7.3 Leave All state machine

A single Leave All state machine exists for each full GARP Participant. Leave All messages generated by this state machine also generate LeaveAll events against all the Applicant and Registrar state machines associated with that Participant and Port; hence, LeaveAll generation is treated by those state machines in the same way as reception of a LeaveAll message from an external source.

The detailed operation of this state machine is described in Table 12-5.

12.7.4 Combined Applicant and Registrar state machine

Table 12-6 shows all the reachable states, with cells containing the joint state names, Applicant.Registrar, and unreachable states marked ---. The MT and LV Registrar states are grouped together since the only event

Table 12-4—Registrar state table

		STATE		
		IN	**LV**	**MT**
EVENT	**rJoinIn**	IN	Stop leavetimer IndJoin IN	IndJoin IN
	rJoinEmpty	IN	Stop leavetimer IndJoin IN	IndJoin IN
	rEmpty	IN	LV	MT
	rLeaveIn	Start leavetimer LV	LV	MT
	rLeaveEmpty	Start leavetimer LV	LV	MT
	LeaveAll	Start leavetimer LV	LV	MT
	leavetimer!	-x-	IndLeave MT	-x-
	Initialize	MT	MT	MT

Table 12-5—Leave All state table

		STATE	
		Active	**Passive**
EVENT	**transmitPDU!**	sLeaveAll Passive	-x-
	LeaveAll	Start leavealltimer Passive	Start leavealltimer Passive
	leavealltimer!	Start leavealltimer Active	Start leavealltimer Active
	(All other events)	-x-	-x-

that differentiates the two is the expiry of the leave timer, which does not affect any of the other states. There are 24 reachable states in all.

The combined state machine is shown in Table 12-7. For compactness, the actions—what message is transmitted, when the implementation should check or start timers, when to indicate joins and leaves to the higher-layer user—have been omitted.

12.7.5 Applicant Only GARP Participant

An Applicant Only GARP Participant maintains a single instance of the Applicant Only state machine for each Attribute value for which the Participant needs to maintain state information.

Table 12-6—Combined Applicant/Registrar states

	Very Anxious		Anxious		Quiet		Leaving	
Active Member	VA.MT VA.LV	VA.IN	AA.MT AA.LV	AA.IN	QA.MT QA.LV	QA.IN	LA.MT LA.LV	LA.IN
Passive Member	VP.MT VP.LV	VP.IN	--- ---	AP.IN	---	QP.IN		
Observer	VO.MT VO.LV	VO.IN	--- ---	AO.IN	---	QO.IN	LO.MT LO.LV	---

Table 12-7—Combined Applicant/Registrar state table

		EVENT									
		leavetimer!	**transmitPDU!**	**rJoinIn**	**rJoinEmpty**	**rEmpty**	**rLeaveIn**	**rLeaveEmpty, LeaveAll**	**ReqJoin**	**ReqLeave**	**Initialize**
STATE	**VA.MT**	-x-	AA.MT	AA.IN	VA.IN	VA.MT	VA.MT	VP.MT	-x-	LA.MT	VO.MT
	VA.LV	VA.MT	AA.LV	AA.IN	VA.IN	VA.LV	VA.LV	VP.LV	-x-	LA.LV	VO.MT
	VA.IN	-x-	AA.IN	AA.IN	VA.IN	VA.IN	VA.LV	VP.LV	-x-	LA.IN	VO.MT
	AA.MT	-x-	QA.MT	QA.IN	VA.IN	VA.MT	VA.MT	VP.MT	-x-	LA.MT	VO.MT
	AA.LV	AA.MT	QA.LV	QA.IN	VA.IN	VA.LV	VA.LV	VP.LV	-x-	LA.LV	VO.MT
	AA.IN	-x-	QA.IN	QA.IN	VA.IN	VA.IN	VA.LV	VP.LV	-x-	LA.IN	VO.MT
	QA.MT	-x-	—	QA.IN	VA.IN	VA.MT	VP.MT	VP.MT	-x-	LA.MT	VO.MT
	QA.LV	QA.MT	—	QA.IN	VA.IN	VA.LV	VP.LV	VP.LV	-x-	LA.LV	VO.MT
	QA.IN	-x-	—	QA.IN	VA.IN	VA.IN	VP.LV	VP.LV	-x-	LA.IN	VO.MT
	LA.MT	-x-	VO.MT	LA.IN	VO.IN	LA.MT	LA.MT	VO.MT	VA.MT	-x-	VO.MT
	LA.LV	LA.MT	VO.LV	LA.IN	VO.IN	LA.LV	LA.LV	VO.LV	VA.LV	-x-	VO.MT
	LA.IN	-x-	VO.LV	LA.IN	VO.IN	LA.IN	LA.LV	VO.LV	VA.IN	-x-	VO.MT
	VP.MT	-x-	AA.MT	AP.IN	VP.IN	VP.MT	VP.MT	VP.MT	-x-	VO.MT	VO.MT
	VP.LV	VP.MT	AA.LV	AP.IN	VP.IN	VP.LV	VP.LV	VP.LV	-x-	VO.LV	VO.MT
	VP.IN	-x-	AA.IN	AP.IN	VP.IN	VP.IN	VP.LV	VP.LV	-x-	VO.IN	VO.MT
	AP.IN	-x-	QA.IN	QP.IN	VP.IN	VP.IN	VP.LV	VP.LV	-x-	AO.IN	VO.MT
	QP.IN	-x-	—	QP.IN	VP.IN	VP.IN	VP.LV	VP.LV	-x-	QO.IN	VO.MT
	VO.MT	-x-	—	AO.IN	VO.IN	VO.MT	LO.MT	LO.MT	VP.MT	-x-	VO.MT
	VO.LV	VO.MT	—	AO.IN	VO.IN	VO.LV	LO.LV	LO.LV	VP.LV	-x-	VO.MT
	VO.IN	-x-	—	AO.IN	VO.IN	VO.IN	LO.LV	LO.LV	VP.IN	-x-	VO.MT
	AO.IN	-x-	—	QO.IN	VO.IN	VO.IN	LO.LV	LO.LV	AP.IN	-x-	VO.MT
	QO.IN	-x-	—	QO.IN	VO.IN	VO.IN	LO.LV	LO.LV	QP.IN	-x-	VO.MT
	LO.MT	-x-	VO.MT	AO.IN	VO.IN	VO.MT	VO.MT	VO.MT	VP.MT	-x-	VO.MT
	LO.LV	LO.MT	VO.LV	AO.IN	VO.IN	VO.LV	VO.LV	VO.LV	VP.LV	-x-	VO.MT

The detailed operation of this state machine is described in Table 12-8. The state transitions shown handle the possibilities of receiving either Leave In or Leave Empty messages; however, only Leave Empty messages are generated by the state machine.

The initial state for the Applicant Only state machine on (re)initialization is MT.

Table 12-8—Applicant Only State Machine

		STATE									
		VA	**AA**	**QA**	**LA**	**VP**	**AP**	**QP**	**VO**	**AO**	**QO**
EVENT	**transmitPDU!**	sJ[I] AA	sJ[I] QA	-x-	sLE VO	sJ[I] AA	sJ[I] QA	-x-	-x-	-x-	-x-
	rJoinIn	AA	QA	QA	LA	AP	QP	QP	AO	QO	QO
	rJoinEmpty	VA	VA	VA	VO	VP	VP	VP	VO	VO	VO
	rEmpty	VA	VA	VA	LA	VP	VP	VP	VO	VO	VO
	rLeaveIn	VA	VA	VA	LA	VP	VP	VP	VO	VO	VO
	rLeaveEmpty	VP	VP	VP	VO	VP	VP	VP	VO	VO	VO
	LeaveAll	VP	VP	VP	VO	VP	VP	VP	VO	VO	VO
	ReqJoin	-x-	-x-	-x-	VA	-x-	-x-	-x-	VP	AP	QP
	ReqLeave	LA	LA	LA	-x-	VO	AO	QO	-x-	-x-	-x-
	Initialize	VO	VO	VO	VO	VO	VO	VO	VO	VO	VO

12.7.6 Simple-Applicant Participant

A Simple-Applicant Participant maintains a single instance of the Simple Applicant state machine for each Attribute value for which the Participant needs to maintain state information.

The detailed operation of this state machine is described in Table 12-9. The state transitions shown handle the possibilities of receiving either Leave In or Leave Empty messages; however, only Leave Empty messages are generated by the state machine.

12.8 Administrative controls

Associated with each instance of the Registrar state machines are *Registrar Administrative Control* parameters. These parameters allow administrative control to be exercised over the registration state of each Attribute value, and hence, via the propagation mechanism provided by GIP, allow control to be exercised over the propagation of declarations.

An overall control parameter for each Applicant state machine, the *Applicant Administrative Control,* determines whether or not the Applicant state machine participates in GARP protocol exchanges.

These parameters can be set to the values defined in 12.8.1 and 12.8.2.

12.8.1 Registrar Administrative Control values

a) *Normal Registration.* The Registrar responds normally to incoming GARP messages.
b) *Registration Fixed.* The Registrar ignores all GARP messages, and remains in the IN (registered) state.
c) *Registration Forbidden.* The Registrar ignores all GARP messages, and remains in the EMPTY (unregistered) state.

Table 12-9—Simple Applicant State Machine

		STATE		
		V	**A**	**Q**
EVENT	**transmitPDU!**	sJ[I] A	sJ[I] Q	-x-
	rJoinIn	A	Q	Q
	rJoinEmpty	V	V	V
	rEmpty	V	V	V
	rLeaveIn	V	V	V
	rLeaveEmpty	V	V	V
	LeaveAll	V	V	V
	ReqJoin	-x-	-x-	-x-
	ReqLeave	sLE O	sLE O	sLE O

The default value of this parameter is Normal Registration.

Optionally, an implementation may support the ability to record against each Registrar state machine the MAC Address of the originator of the GARP PDU that caused the most recent state change for that state machine.

NOTE—The Registrar Administrative Controls are realized by means of the contents of the Port Map parameters of static entries in the Filtering Database for all GARP applications. In the case of GMRP, the static entries concerned are Static Filtering Entries (7.9.1 and 10.3.2.3). The contents of the Port Map parameters in static entries can be modified by means of the management operations defined in 14.7. In the absence of such control information for a given attribute, the default value “Normal Registration” is assumed.

12.8.2 Applicant Administrative Control values

a) *Normal Participant.* The state machine participates normally in GARP protocol exchanges.

b) *Non-Participant.* The state machine does not send any GARP messages.

The default value of this parameter is Normal Participant.

NOTE—The Applicant Administrative Control parameters can be modified for any GARP application by means of the management operations defined in 14.9. In the absence of such information for a given attribute, the default value “Normal Participant” is assumed.

12.9 Procedures

The following subclauses define the protocol actions and procedures that are identified in the description of the State Machines contained in 12.7.

12.9.1 Discarding badly formed GARP PDUs

A GARP Participant that receives a GARP PDU shall discard that PDU if any of the following are true:

a) The PDU carries an unknown protocol identifier.
b) The PDU is not formatted according to the GARP PDU format defined in 12.10.

Items of information contained within a GARP PDU that are not understood by the GARP Application shall be discarded as described in 12.10.3.

12.9.2 Protocol parameters and timers

12.9.2.1 jointimer

The Join Period Timer, jointimer, controls the interval between transmitPDU! events that are applied to the Applicant State Machine. An instance of this timer is required on a per-Port, per-GARP Participant basis. The maximum time period between transmitPDU! events is defined by JoinTime, as defined in Table 12-10.

12.9.2.2 leavetimer

The Leave Period Timer, leavetimer, controls the period of time that the Registrar State Machine will wait in the LV state before transiting to the MT state. An instance of the timer is required for each state machine that is in the LV state. The Leave Period Timer is set to the value LeaveTime when it is started or restarted; LeaveTime is defined in Table 12-10.

12.9.2.3 leavealltimer

The Leave All Period Timer, leavealltimer, controls the frequency with which the Leave All state machine generates LeaveAll PDUs. The timer is required on a per-Port, per-GARP Participant basis. The Leave All Period Timer is set to a random value, T, in the range LeaveAllTime < T < 1.5*LeaveAllTime when it is started or restarted. LeaveAllTime is defined in Table 12-10.

12.9.3 Protocol event definitions

Unless stated otherwise in these event definitions, GARP PDU reception in a Bridge can occur through all Ports of a Bridge, and events generated as a result of such reception affect only those state machines that are associated with the Port through which the PDU was received.

12.9.3.1 Initialize

The state machine is initialized or reinitialized.

12.9.3.2 ReqJoin

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the ReqJoin event is deemed to have occurred if the GID Service User issues a GID_Join.request service primitive for the Attribute instance associated with that state machine.

12.9.3.3 ReqLeave

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the ReqLeave event is deemed to have occurred if the GID Service User issues a GID_Leave.request service primitive for the Attribute instance associated with that state machine.

12.9.3.4 rJoinIn

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the rJoinIn event is deemed to have occurred if a GARP PDU (12.10) is received, and the following conditions are true:

a) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
b) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
c) The Message contains an Attribute in which the Attribute Event (12.10.2.4) specifies the JoinIn event, and the Attribute Value (12.10.2.6) is equal to the value associated with the state machine.

12.9.3.5 rJoinEmpty

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the rJoinEmpty event is deemed to have occurred if a GARP PDU (12.10) is received, and the following conditions are true:

a) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
b) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
c) The Message contains an Attribute in which the Attribute Event (12.10.2.4) specifies the JoinEmpty event, and the Attribute Value (12.10.2.6) is equal to the value associated with the state machine.

12.9.3.6 rEmpty

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the rEmpty event is deemed to have occurred if a GARP PDU (12.10) is received, and the following conditions are true:

a) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
b) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
c) The Message contains an Attribute in which the Attribute Event (12.10.2.4) specifies the Empty event, and the Attribute Value (12.10.2.6) is equal to the value associated with the state machine.

12.9.3.7 rLeaveIn

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the rLeaveIn event is deemed to have occurred if a GARP PDU (12.10) is received, and the following conditions are true:

a) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
b) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
c) The Message contains an Attribute in which the Attribute Event (12.10.2.4) specifies the LeaveIn event, and the Attribute Value (12.10.2.6) is equal to the value associated with the state machine.

12.9.3.8 rLeaveEmpty

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the rLeaveEmpty event is deemed to have occurred if a GARP PDU (12.10) is received, and the following conditions are true:

a) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
b) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
c) The Message contains an Attribute in which the Attribute Event (12.10.2.4) specifies the LeaveEmpty event, and the Attribute Value (12.10.2.6) is equal to the value associated with the state machine.

12.9.3.9 LeaveAll

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, the Simple Applicant state machine or the Leave All state machine, the LeaveAll event is deemed to have occurred if

a) A GARP PDU (12.10) is received, and the following conditions are all true:
    1) The PDU was addressed to the GARP Application address (Table 12-1) of the GARP Application associated with the state machine.
    2) The PDU contains a Message (12.10.1) in which the Attribute Type is the type associated with the state machine.
    3) The Message contains a LeaveAll Attribute in which the LeaveAll Event (12.10.2.5) is present.

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine or the Simple Applicant state machine, the LeaveAll event is deemed to have occurred if

b) The Leave All state machine associated with that state machine performs the sLeaveAll action (12.9.4.5).

NOTE—The LeaveAll state machine operates on a per-Application (not per-Attribute Type) basis, but the LeaveAll message operates on a per-Attribute Type basis. Hence, when the LeaveAll state machine issues a LeaveAll, it must generate a LeaveAll Attribute for each Attribute Type supported by the Application concerned.

12.9.3.10 leavetimer!

For an instance of the combined Applicant/Registrar state machine, the leavetimer! event is deemed to have occurred when the leavetimer associated with that state machine expires.

12.9.3.11 leavealltimer!

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, the Simple Applicant state machine, or LeaveAll state machine, the leavealltimer! event is deemed to have occurred when the leavealltimer associated with that state machine expires.

12.9.3.12 transmitPDU!

For an instance of the combined Applicant/Registrar state machine, the Applicant Only state machine, or the Simple Applicant state machine, the transmitPDU! event is deemed to have occurred when the jointimer associated with that state machine expires.

For an instance of the LeaveAll state machine, the transmitPDU! event is deemed to have occurred when the state machine has an opportunity to transmit a LeaveAll message.

12.9.4 Action definitions

Unless stated otherwise in these action definitions, GARP PDU transmission as a result of the operation of a state machine in a Bridge occurs only through the Port associated with that state machine, and only if that Port is in the Forwarding state.

12.9.4.1 -x-

No action is taken.

12.9.4.2 sJ[E, I], sJ[I]

A GARP PDU, formatted as defined in 12.10.1, is transmitted. The PDU shall be formatted such that

a) The PDU contains a Message (12.10.1.2) that carries an Attribute Type (12.10.2.2) that specifies the type associated with the state machine.
b) The Message contains an Attribute (12.10.1.2) that specifies an Attribute Event (12.10.2.4) equal to JoinIn (if the Registrar is in the IN state, or if no Registrar functionality is implemented) or JoinEmpty (if the Registrar is in either the LV or the MT state), and an Attribute Value equal to the value associated with the state machine.

The PDU shall be transmitted using, as the destination MAC Address, the GARP Application address of the GARP Application associated with the state machine.

12.9.4.3 sE

A GARP PDU, formatted as defined in 12.10.1, is transmitted. The PDU shall be formatted such that

a) The PDU contains a Message (12.10.1.2) that carries an Attribute Type (12.10.2.2) that specifies the type associated with the state machine.
b) The Message contains an Attribute (12.10.1.2) that specifies an Attribute Event (12.10.2.4) equal to Empty, and an Attribute Value equal to the value associated with the state machine.

The PDU shall be transmitted using, as the destination MAC Address, the GARP Application address of the GARP Application associated with the state machine.

12.9.4.4 sLE

A GARP PDU, formatted as defined in 12.10.1, is transmitted. The PDU shall be formatted such that

a) The PDU contains a Message (12.10.1.2) that carries an Attribute Type (12.10.2.2) that specifies the type associated with the state machine.
b) The Message contains an Attribute (12.10.1.2) that specifies an Attribute Event (12.10.2.4) equal to LeaveEmpty, and an Attribute Value equal to the value associated with the state machine.

The PDU shall be transmitted using, as the destination MAC Address, the GARP Application address of the GARP Application associated with the state machine.

12.9.4.5 sLeaveAll

A GARP PDU, formatted as defined in 12.10.1, is transmitted. The PDU shall be formatted such that, for each Attribute Type associated with the GARP Application

a) The PDU contains a Message (12.10.1.2) that carries an Attribute Type (12.10.2.2) that specifies the Attribute Type concerned.
b) The Message contains a LeaveAll Attribute (12.10.1.2).

The PDU shall be transmitted using, as the destination MAC Address, the GARP Application address of the GARP Application associated with the state machine.

The sLeaveAll action also gives rise to a LeaveAll event against all instances of the combined Applicant/ Registrar state machine, the Applicant Only state machine and the Simple Applicant state machine associated with the GARP Application.

12.9.4.6 Start leavetimer

Causes leavetimer to be started, in accordance with the definition of the timer in 12.9.2.2.

12.9.4.7 Stop leavetimer

Causes leavetimer to be stopped.

12.9.4.8 Start leavealltimer

Causes leavealltimer to be started, in accordance with the definition of the timer in 12.9.2.3.

12.9.4.9 IndJoin

When an instance of the Registrar state machine makes a transition from the LV or MT state to the IN state, the IndJoin action causes a GID_Join.indication primitive to be issued to the GID Service User, indicating the Attribute instance corresponding to the state machine concerned.

12.9.4.10 IndLeave

When an instance of the Registrar state machine makes a transition from the LV state to the MT state, the IndLeave action causes a GID_Leave.indication primitive to be issued to the GID Service User, indicating the Attribute instance corresponding to the state machine concerned.

12.9.4.11 Failure to register

Each GARP Participant maintains a count of the number of times that it has received a registration request, but has failed to register the attribute concerned due to lack of space in the Filtering Database to record the registration. The value of this count may be examined by management.

NOTE—Further action to be taken on such events is a matter for implementation choice.

12.10 Structure and encoding of GARP Protocol Data Units

This subclause describes the generic structure and encoding of the GARP Protocol Data Units (GARP PDUs) exchanged between all GARP Participants. The structure and encoding of elements that are specific to the operation of the GARP Applications are defined by the Applications themselves.

Each GARP PDU identifies the GARP Application by which it was generated, and to which it is being transmitted. Bridges that receive GARP PDUs identified as belonging to a GARP Application that they do not support shall forward such PDUs on all other Ports that are in a Forwarding state.

NOTE 1—If GARP is used to support an Application that can operate in any GIP Context other than 0 (the Base Spanning Tree), the application specification describes how that context is identified in protocol exchanges.

Each GARP PDU carries one or more GARP messages, each of which identify a GARP event (e.g., Join, Leave, LeaveAll) and the attribute class(es) and value(s) to which that event applies. A given GARP Participant shall process GARP PDUs in the order in which they are received, and shall process the GARP Messages in a PDU in the order in which they were put into the Data Link Service Data Unit (DLSDU).

NOTE 2—Any messages generated as a consequence of state machine responses to an sLeaveAll action and its associated LeaveAll events will be put into the DLSDU after the LeaveAll message(s), or into a later DLSDU.

12.10.1 Structure

12.10.1.1 Transmission and representation of octets

All GARP PDUs consist of an integral number of words, numbered starting from 1 and increasing in the order that they are put into a Data Link Service Data Unit (DLSDU). The bits in each octet are numbered from 1 to 8, where 1 is the low-order bit.

When consecutive octets are used to represent a binary number, the lower octet number has the most significant value.

When the encoding of (an element of) a GARP PDU is represented using a diagram in this clause, the following representations are used:

a) Octet 1 is shown towards the top of the page, higher numbered octets being towards the bottom.
b) Where more than one octet appears on a given line, octets are shown with the lowest numbered octet to the left, higher numbered octets being to the right.
c) Within an octet, bits are shown with bit 8 to the left and bit 1 to the right.

12.10.1.2 Structure definition

A Protocol Identifier shall be encoded in the initial octets of all GARP PDUs. This standard reserves a single Protocol Identifier value to identify the GARP. GARP PDUs operating the Protocol specified in this clause carry this reserved Protocol Identifier value, and shall have the following structure:

a) The first two octets contain the *Protocol Identifier* value.
b) Following the Protocol Identifier are one or more *Messages*. The last element in the PDU is an *End Mark*.
c) Each Message consists of an *Attribute Type* and an *Attribute List,* in that order.
d) An Attribute List consists of one or more *Attribute*s. The last element in the Attribute List is an End Mark.
e) An Attribute consists of an *Attribute Length*, an *Attribute Event*, and an *Attribute Value*, in that order. In the case where the Attribute Event is “LeaveAll,” the Attribute Value is omitted.

The following BNF productions give the formal description of the GARP PDU structure:

```
GARP PDU ::= Protocol ID, Message {, Message}, End Mark
Protocol ID SHORT ::= 1
Message ::= Attribute Type, Attribute List
```

Attribute Type BYTE ::= *Defined by the specific GARP Application*
Attribute List ::= Attribute {,Attribute}, End Mark
Attribute ::= Ordinary Attribute | LeaveAll Attribute
Ordinary Attribute ::= Attribute Length, Attribute Event, Attribute Value
LeaveAll Attribute ::= Attribute Length, LeaveAll Event
Attribute Length BYTE ::= 2-255
Attribute Event BYTE ::= JoinEmpty | JoinIn | LeaveEmpty | LeaveIn | Empty
LeaveAll Event BYTE ::= LeaveAll
Attribute Value ::= *Defined by the specific GARP Application*
End Mark ::= 0x00 | *End of PDU*
LeaveAll ::= 0
JoinEmpty ::= 1
JoinIn ::= 2
LeaveEmpty ::= 3
LeaveIn ::= 4
Empty ::= 5

The parameters carried in GARP PDUs, as identified in this structure definition, shall be encoded as specified in 12.10.2.

Figure 12-6 illustrates the structure of the GARP PDU and its components.

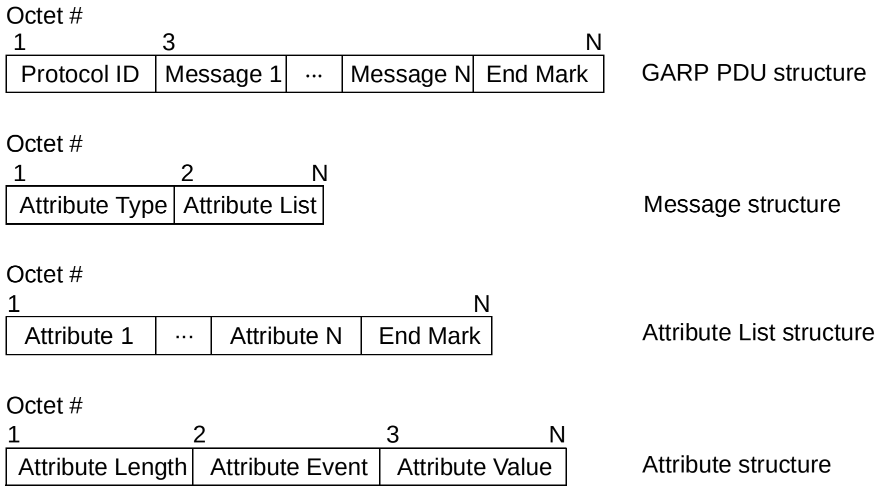


Figure 12-6—Format of the major components of a GARP PDU

12.10.2 Encoding of GARP PDU parameters

12.10.2.1 Encoding of Protocol Identifier

A Protocol Identifier shall be encoded in two octets, taken to represent an unsigned binary number. It takes the hexadecimal value 0x0001, which identifies the GARP protocol as defined in this clause.

12.10.2.2 Encoding of Attribute Type

An Attribute Type shall be encoded as a single octet, taken to represent an unsigned binary number. The Attribute Type identifies the type of Attribute to which the message applies. The range of values that can be taken by the Attribute Type, and the meanings of those values, are defined by the Application concerned. The value 0 is reserved, and shall not be used as an Attribute Type by any GARP Application. GARP

Applications may otherwise allocate meanings to any set of values of Attribute Type in the range 1 through 255.

12.10.2.3 Encoding of Attribute Length

An Attribute Length shall be encoded as a single octet, taken to represent an unsigned binary number, equal to the number of octets occupied by an Attribute, inclusive of the Attribute Length field. Valid values of Attribute Length are in the range 2 through 255.

Further values of Attribute Length are reserved and shall not be used.

12.10.2.4 Encoding of Attribute Event

An Attribute Event shall be encoded as a single octet, taken to represent an unsigned binary number. The permitted values and meanings of the Attribute Event are as follows:

1:	JoinEmpty operator
2:	JoinIn operator
3:	LeaveEmpty operator
4:	LeaveIn operator
5:	Empty operator

Further values of Attribute Event are reserved.

The Attribute Event is interpreted on receipt as a GID event to be applied to the state machine for the Attribute defined by the Attribute Type and Attribute Value fields.

12.10.2.5 Encoding of LeaveAll Event

A LeaveAll Event shall be encoded as a single octet, taken to represent an unsigned binary number. The permitted values and meanings of LeaveAll Event are as follows:

0:	LeaveAll operator

Further values of LeaveAll Event are reserved.

The LeaveAll Event is interpreted on receipt as a GID Leave All event to be applied to the state machines for all Attributes of the type defined by the Attribute Type field.

12.10.2.6 Encoding of Attribute Value

An Attribute Value is encoded in N octets, in accordance with the specification for the Attribute Type, as defined by the GARP Application concerned.

12.10.2.7 Encoding of End Mark

An End Mark shall be encoded as a single octet, taken to represent the unsigned binary number. It takes the value 0.

Further values of End Mark are reserved and shall not be used.

NOTE—As defined by the GARP PDU structure definition in 12.10.1, if the end of the GARP PDU is encountered, this is taken to be an End Mark from the point of view of processing the PDU contents.

12.10.3 Packing and parsing GARP PDUs

The use of the End Mark (12.10.2.7) to signal the end of an Attribute List and the end of a GARP PDU, and the fact that the (physical) end of the PDU is interpreted as an End Mark, simplifies the requirements both for packing information into GARP PDUs and for correctly interpreting that information on receipt.

12.10.3.1 Packing

Successive Messages are packed into the GARP PDU, and within each Message, successive Attributes are packed into each Message, until the end of the PDU is encountered or there are no more attributes to pack at that time. The following cases can occur:

a) The PDU has sufficient room for all the Attributes that require to be transmitted at that time to be packed. In this case, the PDU is transmitted, and subsequent PDUs are transmitted when there are further Attributes to transmit.

b) The PDU has exactly enough room for the first N Attributes that require to be transmitted at that time to be packed. In this case, the PDU is transmitted, and the next N Attributes are encoded in a subsequent PDU.

c) The PDU has enough room for the first N Attributes that require to be transmitted at that time to be packed, but the remaining space in the PDU is too small for Attribute N+1, so the last few octets of the PDU carry a partial encoding of Attribute N+1. In this case, the PDU can be transmitted as it is, and Attribute N+1 and its successors are encoded in full in a subsequent PDU.

12.10.3.2 Parsing

Successive Messages, and within each Message, successive Attributes, are unpacked from the PDU. If this process terminates because the end of the PDU is reached, then the end of the PDU is taken to signal termination both of the current Attribute List and the overall PDU. The following two cases can occur:

a) The last Attribute to be unpacked was complete. In this case, the Attribute is processed normally, and processing of the PDU terminates.

b) The last Attribute to be unpacked was incomplete. In this case, the partial Attribute is discarded, and processing of the PDU terminates.

12.10.3.3 Discarding unrecognized information

In order to allow backward compatibility with previous versions of a given GARP Application, the following procedure is adopted when unrecognized elements within a received GARP PDU are encountered:

a) If a Message is encountered in which the Attribute Type is not recognized, then that Message is discarded. This is achieved by discarding the successive Attributes in the Attribute List until either an End Mark or the end of the PDU is reached. If an End Mark is reached, processing continues with the next Message.

b) If an Attribute is encountered in which the Attribute Event is not recognized for the Attribute Type concerned, then the Attribute is discarded and processing continues with the next Attribute or Message if the end of the PDU has not been reached.

12.11 Timer values, granularity and relationships

12.11.1 Timer values

The default timer values used in the GARP protocol are defined in Table 12-10. The values used for the GARP timers may be modified on a per-Port basis by means of the management functionality defined in Clause 14.

Table 12-10—GARP timer parameter values

Parameter	Value (centiseconds)
JoinTime	20
LeaveTime	60
LeaveAllTime	1000

NOTE—The default values for the GARP timers are independent of media access method or data rate. This is a deliberate choice, made in the interests of maximizing the “plug and play” characteristics of the protocol.

12.11.2 Timer resolution

Implementation of the timers for GARP shall be based on a timer resolution of 5 centiseconds or less.

12.11.3 Timing relationships

GARP protocol *correctness* does not depend critically on timing relationships; however, the protocol operates more efficiently, and with less likelihood of unwanted de-registrations, if the following relationships are maintained between the protocol timers operating in state machines that exchange GARP PDUs on the same LAN:

a) JoinTime should be chosen such that at least two JoinTimes can occur within the value of LeaveTime being used on the LAN. This ensures that after a Leave or LeaveAll message has been issued, the Applicants can re-Join before a further Leave is issued;
b) LeaveAllTime should be larger than the value of LeaveTime being used on the LAN. In order to minimize the volume of re-joining traffic generated following a LeaveAll, the value chosen for LeaveAllTime should be large relative to LeaveTime.

These relationships are illustrated in Figure 12-7. The time intervals labeled A (LeaveTime minus two JoinTimes) and B (LeaveAll Period minus LeaveTime) should all be positive and non-zero in value for the efficient operation of GARP. The time parameter values specified in Table 12-10 have been chosen in order to ensure that these timer relationships are maintained.

12.12 Interoperability considerations

Correct operation of the GARP protocol for a given GARP application requires that protocol exchanges among a given set of communicating GARP Participants maintain sequentiality; i.e., that Participant A cannot receive GARP PDU B (generated as a consequence of Participant B receiving GARP PDU A) before Participant A has received GARP PDU A. In circumstances where the Participants concerned are all attached to the same LAN, such sequentiality is ensured. However, if a set of GARP Participants communicates via an intervening Bridge that does not implement that GARP application (or does not implement GARP at all), the sequentiality constraints expressed in 7.7.3 are insufficient to guarantee the correct operation of the

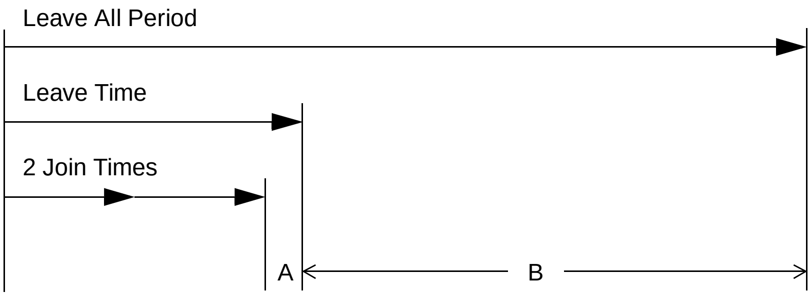


Figure 12-7—GARP timing relationships

GARP protocol. In order for the correct sequencing of PDUs to be maintained through such a Bridge, the following constraint must be met:

> If GARP PDU A is received on Port X, and is due to be forwarded on Ports Y and Z, and subsequent to being forwarded on Y, GARP PDU B is received on Port Y for forwarding on Port Z, then forwarding of B cannot precede A on Port Z.

NOTE—This expresses a stronger sequencing constraint for multicast frames than is stated in 7.7.3, but a weaker constraint than was required for conformance to IEEE Std 802.1D, 1993 Edition.

The consequence of failure to meet this constraint is that the users of a given GARP application may experience an increased incidence of loss of registration. Therefore, it is inadvisable to construct LAN configurations involving forwarding of GARP PDUs through intervening Bridges if those Bridges do not meet the constraint expressed above.

13. Example “C” code implementation of GARP

In IEEE Std 802.1D, 1998 Edition, this clause provided an example implementation of GARP (Clause 12). It contained no normative provisions and has been superseded by other ways of sharing implementation information.

14. Bridge management

Management facilities are provided by MAC Bridges in accordance with the principles and concepts of the OSI Management Framework.

This clause

a) Introduces the Functional Areas of OSI Management to assist in the identification of the requirements placed on Bridges for the support of management facilities.
b) Establishes the correspondence between the Processes used to model the operation of the Bridge (7.3) and the managed objects of the Bridge.
c) Specifies the management operations supported by each managed object.

14.1 Management functions

The Functions of Management relate to the users' needs for facilities that support the planning, organization, supervision, control, protection, and security of communications resources, and account for their use. These facilities may be categorized as supporting the Functional Areas of Configuration, Fault, Performance, Security, and Accounting Management. Each of these is summarized in 14.1.1 through 14.1.5, together with the facilities commonly required for the management of communication resources, and the particular facilities provided in that functional area by Bridge Management.

14.1.1 Configuration Management

Configuration Management provides for the identification of communications resources, initialization, reset and close down, the supply of operational parameters, and the establishment and discovery of the relationship between resources. The facilities provided by Bridge Management in this functional area are as follows:

a) The identification of all Bridges that together make up the Bridged Local Area Network and their respective locations and, as a consequence of that identification, the location of specific end stations to particular individual LANs.
b) The ability to remotely reset, i.e., reinitialize, specified Bridges.
c) The ability to control the priority with which a Bridge Port transmits frames.
d) The ability to force a specific configuration of the spanning tree.
e) The ability to control the propagation of frames with specific group MAC Addresses to certain parts of the configured Bridged Local Area Network.

14.1.2 Fault Management

Fault Management provides for fault prevention, detection, diagnosis, and correction. The facilities provided by Bridge Management in this functional area are as follows:

a) The ability to identify and correct Bridge malfunctions, including error logging and reporting.

14.1.3 Performance Management

Performance management provides for evaluation of the behavior of communications resources and of the effectiveness of communication activities. The facilities provided by Bridge Management in this functional area are as follows:

a) The ability to gather statistics relating to performance and traffic analysis. Specific metrics include network utilization, frame forward, and frame discard counts for individual Ports within a Bridge.

14.1.4 Security Management

Security Management provides for the protection of resources. Bridge Management does not provide any specific facilities in this functional area.

14.1.5 Accounting Management

Accounting Management provides for the identification and distribution of costs and the setting of charges. Bridge Management does not provide any specific facilities in this functional area.

14.2 Managed objects

Managed objects model the semantics of management operations. Operations upon an object supply information concerning, or facilitate control over, the Process or Entity associated with that object.

The managed resources of a MAC Bridge are those of the Processes and Entities established in 7.3 and 12.1. Specifically

a) The Bridge Management Entity (14.4 and 7.11).
b) The individual MAC Entities associated with each Bridge Port (14.5, 7.2, 7.5, and 7.6).
c) The Forwarding Process of the MAC Relay Entity (14.6, 7.2, and 7.7).
d) The Filtering Database of the MAC Relay Entity (14.7 and 7.9).
e) The Spanning Tree Protocol Entity (14.8, 7.10 and Clause 17).
f) GARP Participants (Clause 12).
g) GMRP participants (14.10 and Clause 10).

The management of each of these resources is described in terms of managed objects and operations below.

NOTE—The values specified in this clause, as inputs and outputs of management operations, are abstract information elements. Questions of formats or encodings are a matter for particular protocols that convey or otherwise represent this information.

14.3 Data types

This subclause specifies the semantics of operations independent of their encoding in management protocol. The data types of the parameters of operations are defined only as required for that specification.

The following data types are used:

a) Boolean.
b) Enumerated, for a collection of named values.
c) Unsigned, for all parameters specified as “the number of” some quantity, and for values that are numerically compared. In the case of numeric comparisons of Spanning Tree priority values, the lower number represents the higher priority value.
d) MAC Address.
e) Latin1 String, as defined by ANSI X3.159-1989, for all text strings.
f) Time Interval, an Unsigned value representing a positive integral number of seconds, for all Spanning Tree protocol timeout parameters.
g) Counter, for all parameters specified as a “count” of some quantity. A counter increments and wraps with a modulus of 2 to the power of 64.
h) GARP Time Interval, an Unsigned value representing a positive integral number of centiseconds, for all GARP protocol time-out parameters.

i) Port Number, an Unsigned value assigned to a Port as part of a Port Identifier. Valid Port Numbers are in the range 1 through 4095.
j) Port Priority, an Unsigned value used to represent the priority component of a Port Identifier. Valid Port Priorities are in the range 0 through 240, in steps of 16.
k) Bridge Priority, an Unsigned value used to represent the priority component of a Bridge Identifier. Valid Bridge Priorities are in the range 0 through 61 440, in steps of 4096.

14.4 Bridge Management Entity

The Bridge Management Entity is described in 7.11.

The objects that comprise this managed resource are as follows:

a) The Bridge Configuration.
b) The Port Configuration for each Port.

14.4.1 Bridge Configuration

The Bridge Configuration object models the operations that modify, or enquire about, the configuration of the Bridge’s resources. There is a single Bridge Configuration object per Bridge.

The management operations that can be performed on the Bridge Configuration are Discover Bridge, Read Bridge, Set Bridge Name, and Reset Bridge.

14.4.1.1 Discover Bridge

14.4.1.1.1 Purpose

To solicit configuration information regarding the Bridge(s) in the Bridged Local Area Network.

14.4.1.1.2 Inputs

a) Inclusion Range, a set of ordered pairs of specific MAC Addresses. Each pair specifies a range of MAC Addresses. A Bridge shall respond if and only if
    1) For one of the pairs, the numerical comparison of its Bridge Address with each MAC Address of the pair shows it to be greater than or equal to the first, and
    2) Less than or equal to the second, and
    3) Its Bridge Address does not appear in the Exclusion List parameter.
b) Exclusion List, a list of specific MAC Addresses.

The numerical comparison of one MAC Address with another, for the purpose of this operation, is achieved by deriving a number from the MAC Address according to the following procedure. The consecutive octets of the MAC Address are taken to represent a binary number; the first octet that would be transmitted on a LAN medium when the MAC Address is used in the source or destination fields of a MAC frame has the most significant value, the next octet the next most significant value. Within each octet the first bit of each octet is the least significant bit.

14.4.1.1.3 Outputs

a) Bridge Address—the MAC Address for the Bridge from which the Bridge Identifier used by the Rapid Spanning Tree Protocol is derived.
b) Bridge Name—a text string of up to 32 characters, of locally determined significance.
c) Number of Ports—the number of Bridge Ports (MAC Entities).

d) Port Addresses—a list specifying the following for each Port:
   1) Port Number—the number of the Bridge Port.
   2) Port Address—the specific MAC Address of the individual MAC Entity for the Port.
e) Uptime—count in seconds of the time elapsed since the Bridge was last reset or initialized.

14.4.1.2 Read Bridge

14.4.1.2.1 Purpose

To obtain general information regarding the Bridge.

14.4.1.2.2 Inputs

None.

14.4.1.2.3 Outputs

a) Bridge Address—the MAC Address for the Bridge from which the Bridge Identifier used by the Rapid Spanning Tree Algorithm Protocol is derived.
b) Bridge Name—a text string of up to 32 characters, of locally determined significance.
c) Number of Ports—the number of Bridge Ports (MAC Entities).
d) Port Addresses—a list specifying the following for each Port:
   1) Port Number.
   2) Port Address—the specific MAC Address of the individual MAC Entity for the Port.
e) Uptime—count in seconds of the time elapsed since the Bridge was last reset or initialized.

14.4.1.3 Set Bridge Name

14.4.1.3.1 Purpose

To associate a text string, readable by the Read Bridge operation, with a Bridge.

14.4.1.3.2 Inputs

a) Bridge Name—a text string of up to 32 characters.

14.4.1.3.3 Outputs

None.

14.4.1.4 Reset Bridge

14.4.1.4.1 Purpose

To reset the specified Bridge. The Filtering Database is cleared and initialized with the entries specified in the Permanent Database, and the Spanning Tree Protocol Entity is initialized (17.18.1).

14.4.1.4.2 Inputs

None.

14.4.1.4.3 Outputs

None.

14.4.2 Port Configuration

The Port Configuration object models the operations that modify, or inquire about, the configuration of the Ports of a Bridge. There are a fixed set of Bridge Ports per Bridge (one for each MAC interface), and each is identified by a permanently allocated Port Number.

The allocated Port Numbers are not required to be consecutive. Also, some Port Numbers may be dummy entries, with no actual LAN Port (for example, to allow for expansion of the Bridge by addition of further MAC interfaces in the future). Such dummy Ports shall support the Port Configuration management operations, and other Port-related management operations in a manner consistent with the Port being permanently disabled.

The information provided by the Port Configuration consists of summary data indicating its name and type. Specific counter information pertaining to the number of packets forwarded, filtered, and in error is maintained by the Forwarding Process resource. The management operations supported by the Spanning Tree Protocol Entity allow for controlling the states of each Port.

The management operations that can be performed on the Port Configuration are Read Port and Set Port Name.

14.4.2.1 Read Port

14.4.2.1.1 Purpose

To obtain general information regarding a specific Bridge Port.

14.4.2.1.2 Inputs

a) Port Number—the number of the Bridge Port.

14.4.2.1.3 Outputs

a) Port Name—a text string of up to 32 characters, of locally determined significance.
b) Port Type—the MAC Entity type of the Port (IEEE Std 802.3; IEEE Std 802.4; IEEE Std 802.5; IEEE Std 802.6; IEEE Std 802.9; IEEE Std 802.9a; IEEE Std 802.12 (IEEE Std 802.3 format); IEEE Std 802.12 (IEEE Std 802.5 format); IEEE Std 802.11; ISO 9314-2; other).

NOTE—To support compatibility with equipment implementing prior versions of this standard, the Port Type includes the ability to report ports implemented to standards now withdrawn.

14.4.2.2 Set port name

14.4.2.2.1 Purpose

To associate a text string, readable by the Read Port operation, with a Bridge Port.

14.4.2.2.2 Inputs

a) Port Number.
b) Port Name—a text string of up to 32 characters.

14.4.2.2.3 Outputs

None.

14.5 MAC Entities

The Management Operations and Facilities provided by the MAC Entities are those specified in the Layer Management standards of the individual MACs. A MAC Entity is associated with each Bridge Port.

14.6 Forwarding Process

The Forwarding Process contains information relating to the forwarding of frames. Counters are maintained that provide information on the number of frames forwarded, filtered, and dropped due to error. Configuration data, defining how frame priority is handled, is maintained by the Forwarding Process.

The objects that comprise this managed resource are

a) The Port Counters.
b) The Priority Handling objects for each Port.
c) The Traffic Class Table for each Port.

14.6.1 The Port Counters

The Port Counters object models the operations that can be performed on the Port counters of the Forwarding Process resource. There are multiple instances (one for each MAC Entity) of the Port Counters object per Bridge.

The management operation that can be performed on the Port Counters is Read Forwarding Port Counters.

14.6.1.1 Read Forwarding Port Counters

14.6.1.1.1 Purpose

To read the forwarding counters associated with a specific Bridge Port.

14.6.1.1.2 Inputs

a) Port Number.

14.6.1.1.3 Outputs

a) Frames Received—count of all valid frames received (including BPDUs, frames addressed to the Bridge as an end station, and frames that were submitted to the forwarding process).
b) Discard Inbound—count of valid frames received that were discarded by the Forwarding Process.
c) Forward Outbound—count of frames forwarded to the associated MAC Entity.
d) Discard Lack of Buffers—count of frames that were to be transmitted through the associated Port but were discarded due to lack of buffers.
e) Discard Transit Delay Exceeded—count of frames that were to be transmitted but were discarded due to the maximum bridge transit delay being exceeded (buffering may have been available).
f) Discard on Error—count of frames that were to be forwarded on the associated MAC but could not be transmitted (e.g., frame would be too large).
g) Discard on Error Details—a list of 16 elements, each containing the source address of a frame and the reason why the frame was discarded (frame too large). The list is maintained as a circular buffer. The only reason for discard on error, at present, is transmissible service data unit size exceeded.

14.6.2 Priority Handling

The Priority Handling object models the operations that can be performed upon, or inquire about, the Default User Priority parameter, the User Priority Regeneration Table parameter, and the Outbound Access Priority Table parameter for each Port. The operations that can be performed on this object are Read Port Default User Priority, Set Port Default User Priority, Read Port User Priority Regeneration Table, Set Port User Priority Regeneration Table, and Read Outbound Access Priority Table.

14.6.2.1 Read Port Default User Priority

14.6.2.1.1 Purpose

To read the current state of the Default User Priority parameter (6.4) for a specific Bridge Port.

14.6.2.1.2 Inputs

a) Port number.

14.6.2.1.3 Outputs

a) Default User Priority value—Integer in range 0–7.

14.6.2.2 Set Port Default User Priority

14.6.2.2.1 Purpose

To set the current state of the Default User Priority parameter (6.4) for a specific Bridge Port.

14.6.2.2.2 Inputs

a) Port number.
b) Default User Priority value—Integer in range 0–7.

14.6.2.2.3 Outputs

None.

14.6.2.3 Read Port User Priority Regeneration Table

14.6.2.3.1 Purpose

To read the current state of the User Priority Regeneration Table parameter (7.5.1) for a specific Bridge Port.

14.6.2.3.2 Inputs

a) Port number.

14.6.2.3.3 Outputs

a) Regenerated User Priority value for Received User Priority 0—Integer in range 0–7.
b) Regenerated User Priority value for Received User Priority 1—Integer in range 0–7.
c) Regenerated User Priority value for Received User Priority 2—Integer in range 0–7.
d) Regenerated User Priority value for Received User Priority 3—Integer in range 0–7.
e) Regenerated User Priority value for Received User Priority 4—Integer in range 0–7.

f) Regenerated User Priority value for Received User Priority 5—Integer in range 0–7.
g) Regenerated User Priority value for Received User Priority 6—Integer in range 0–7.
h) Regenerated User Priority value for Received User Priority 7—Integer in range 0–7.

14.6.2.4 Set Port User Priority Regeneration Table

14.6.2.4.1 Purpose

To set the current state of the User Priority Regeneration Table parameter (7.5.1) for a specific Bridge Port.

14.6.2.4.2 Inputs

a) Port number.
b) Regenerated User Priority value for Received User Priority 0—Integer in range 0–7.
c) Regenerated User Priority value for Received User Priority 1—Integer in range 0–7.
d) Regenerated User Priority value for Received User Priority 2—Integer in range 0–7.
e) Regenerated User Priority value for Received User Priority 3—Integer in range 0–7.
f) Regenerated User Priority value for Received User Priority 4—Integer in range 0–7.
g) Regenerated User Priority value for Received User Priority 5—Integer in range 0–7.
h) Regenerated User Priority value for Received User Priority 6—Integer in range 0–7.
i) Regenerated User Priority value for Received User Priority 7—Integer in range 0–7.

14.6.2.4.3 Outputs

None.

14.6.3 Traffic Class Table

The Traffic Class Table object models the operations that can be performed upon, or inquire about, the current contents of the Traffic Class Table for a given Port. The operations that can be performed on this object are Read Port Traffic Class Table and Set Port Traffic Class Table.

14.6.3.1 Read Port Traffic Class Table

14.6.3.1.1 Purpose

To read the contents of the Traffic Class Table (7.7.3) for a given Port.

14.6.3.1.2 Inputs

a) Port Number.

14.6.3.1.3 Outputs

a) The number of Traffic Classes, in the range 1 through 8, supported on the Port.
b) For each value of Traffic Class supported on the Port, the value of the Traffic Class in the range 0 through 7, and the set of user_priority values assigned to that Traffic Class.

14.6.3.2 Set Port Traffic Class Table

14.6.3.2.1 Purpose

To set the contents of the Traffic Class Table (7.7.3) for a given Port.

14.6.3.2.2 Inputs

a) Port number
b) For each value of Traffic Class supported on the Port, the value of the Traffic Class in the range 0–7, and the set of user_priority values assigned to that Traffic Class.

NOTE—If a Traffic Class value greater than the largest Traffic Class available on the Port is specified, then the value applied to the Traffic Class Table is the largest available Traffic Class.

14.6.3.2.3 Outputs

None.

14.6.3.3 Read Outbound Access Priority Table

14.6.3.3.1 Purpose

To read the state of the Outbound Access Priority Table parameter (Table 7-4) for a specific Bridge Port.

14.6.3.3.2 Inputs

a) Port number.

14.6.3.3.3 Outputs

a) Access Priority value for User Priority 0—Integer in range 0–7.
b) Access Priority value for User Priority 1—Integer in range 0–7.
c) Access Priority value for User Priority 2—Integer in range 0–7.
d) Access Priority value for User Priority 3—Integer in range 0–7.
e) Access Priority value for User Priority 4—Integer in range 0–7.
f) Access Priority value for User Priority 5—Integer in range 0–7.
g) Access Priority value for User Priority 6—Integer in range 0–7.
h) Access Priority value for User Priority 7—Integer in range 0–7.

14.7 Filtering Database

The Filtering Database is described in 7.9. It contains filtering information used by the Forwarding Process (7.7) to decide which Ports of the Bridge frames should be forwarded.

The objects that comprise this managed resource are as follows:

a) The Filtering Database.
b) The Static Filtering Entries.
c) The Dynamic Filtering Entries.
d) The Group Registration Entries.
e) The Permanent Database.

14.7.1 The Filtering Database

The Filtering Database object models the operations that can be performed on, or affect, the Filtering Database as a whole. There is a single Filtering Database object per Bridge.

The management operations that can be performed on the Database are Read Filtering Database, Set Filtering Database Ageing Time, and the Create Filtering Entry, Delete Filtering Entry, Read Filtering Entry, and Read Filtering Entry Range operations defined in 14.7.6.

14.7.1.1 Read Filtering Database

14.7.1.1.1 Purpose

To obtain general information regarding the Bridge’s Filtering Database.

14.7.1.1.2 Inputs

None.

14.7.1.1.3 Outputs

a) Filtering Database Size—the maximum number of entries that can be held in the Filtering Database.
b) Number of Static Filtering Entries—the number of Static Filtering Entries currently in the Filtering Database.
c) Number of Dynamic Filtering Entries—the number of Dynamic Filtering Entries currently in the Filtering Database.
d) Ageing Time—for ageing out Dynamic Filtering Entries when the Port associated with the entry is in the Forwarding state.
e) If extended filtering services are supported, Number of Group Registration Entries—the number of Group Registration Entries currently in the Filtering Database.

14.7.1.2 Set Filtering Database ageing time

14.7.1.2.1 Purpose

To set the ageing time for Dynamic Filtering Entries.

14.7.1.2.2 Inputs

a) Ageing Time.

14.7.1.2.3 Outputs

None.

14.7.2 A Static Filtering Entry

A Static Filtering Entry object models the operations that can be performed on a single Static Filtering Entry in the Filtering Database. The set of Static Filtering Entry objects within the Filtering Database changes only under management control.

A Static Filtering Entry object supports the Create Filtering Entry, Delete Filtering Entry, Read Filtering Entry, and Read Filtering Entry Range operations defined in 14.7.6.

14.7.3 A Dynamic Filtering Entry

A Dynamic Filtering Entry object models the operations that can be performed on a single Dynamic Filtering Entry (i.e., one that is created by the Learning Process as a result of the observation of network traffic) in the Filtering Database.

A Dynamic Filtering Entry object supports the Delete Filtering Entry, Read Filtering Entry, and Read Filtering Entry Range operations defined in 14.7.6.

14.7.4 A Group Registration Entry

A Group Registration Entry object models the operations that can be performed on a single Group Registration Entry in the Filtering Database. The set of Group Registration Entry objects within the Filtering Database changes only as a result of GARP protocol exchanges.

A Group Registration Entry object supports the Read Filtering Entry and Read Filtering Entry Range operations defined in 14.7.6.

14.7.5 Permanent Database

The Permanent Database object models the operations that can be performed on, or affect, the Permanent Database. There is a single Permanent Database per Filtering Database.

The management operations that can be performed on the Permanent Database are Read Permanent Database, and the Create Filtering Entry, Delete Filtering Entry, Read Filtering Entry, and Read Filtering Entry Range operations defined in 14.7.6.

14.7.5.1 Read Permanent Database

14.7.5.1.1 Purpose

To obtain general information regarding the Permanent Database.

14.7.5.1.2 Inputs

None.

14.7.5.1.3 Outputs

a) Permanent Database Size—maximum number of entries that can be held in the Permanent Database.
b) Number of Static Filtering Entries—number of Static Filtering Entries currently in the Permanent Database.

14.7.6 General Filtering Database operations

14.7.6.1 Create Filtering Entry

14.7.6.1.1 Purpose

To create or update a Filtering Entry in the Filtering Database or Permanent Database. Only Static Filtering Entries may be created in the Filtering Database or Permanent Database.

14.7.6.1.2 Inputs

a) Identifier—Filtering Database or Permanent Database.
b) Address—MAC Address of the entry.
c) Inbound Port—the Inbound Port to which the operation applies. This parameter specifies either
    1) All Inbound Ports, or
    2) A Port number.
d) Port Map—a set of control indicators, one for each Port, as specified in 7.9.1.

Where the implementation does not support the creation of more than one Static Filtering Entry for the address specified, the value of the Inbound Port parameter is assumed to specify All Inbound Ports.

Where the implementation does not support the ability for static filtering entries to specify the use of dynamic filtering information (7.9.1), the use of this operation to create a Static Filtering Entry in the Filtering Database with the same MAC Address as an existing Dynamic Filtering Entry will cause the existing entry to be replaced by the (new) Static Filtering Entry.

Where the implementation supports the creation of multiple Static Filtering Entries for the same MAC Address (7.9.1), the creation of a new Static Filtering Entry will cause any existing Static Filtering Entry for the same Inbound Port and MAC Address to be replaced by the (new) Static Filtering Entry. The creation of a Static Filtering Entry for All Inbound Ports causes all existing Static Entries for the same MAC Address to be replaced by the (new) Static Filtering Entry.

14.7.6.1.3 Outputs

None.

14.7.6.2 Delete Filtering Entry

14.7.6.2.1 Purpose

To delete a Filtering Entry from the Filtering Database or Permanent Database.

14.7.6.2.2 Inputs

a) Identifier—Filtering Database or Permanent Database.
b) Address—MAC Address of the desired entry.
c) Inbound Port—the Inbound Port to which the operation applies. This parameter specifies either
    1) All Inbound Ports, or
    2) A Port number.

Where the implementation does not support the creation of more than one Static Filtering Entry for the address specified, the value of the Inbound Ports parameter is assumed to specify All Inbound Ports.

Where the implementation supports the creation of more than one Static Filtering Entry for the address specified, a value of the Inbound Ports parameter of All Inbound Ports results in deletion of all Static Filtering Entries for the MAC Address specified.

14.7.6.2.3 Outputs

None.

14.7.6.3 Read filtering entry

14.7.6.3.1 Purpose

To read Filtering Entry and Group Registration Entry information from the Filtering or Permanent Databases. This operation returns both the static and dynamic information held in a given database for a given MAC Address and inbound Port specification.

14.7.6.3.2 Inputs

a) Identifier—Filtering Database or Permanent Database.

b) Address—MAC Address of the desired information.
c) Type—Static or Dynamic entry.
d) If Type = Static entry, then Inbound Port—the Inbound Port to which the operation applies. This parameter specifies either
   1) All Inbound Ports, or
   2) A Port number.

Where the implementation does not support the creation of more than one Static Filtering Entry for the address specified, the value of the Inbound Ports parameter is assumed to specify All Inbound Ports.

NOTE—Dynamic entry types are Dynamic Filtering Entries and Group Registration Entries.

14.7.6.3.3 Outputs

a) Address—MAC Address of the desired entry.
b) Type—Static or Dynamic entry.
c) Port Map—A set of control indicators as appropriate to the entry type, as specified in 7.9.1 through 7.9.3.

14.7.6.4 Read Filtering Entry range

14.7.6.4.1 Purpose

To read a range of Filtering Entries and/or Group Registration Entries from the Filtering or Permanent Databases.

Since the number of values to be returned in the requested range may have exceeded the capacity of the service data unit conveying the management response, the returned entry range is identified. The indices that define the range take on values from zero up to Filtering Database Size minus one.

14.7.6.4.2 Inputs

a) Identifier—Filtering Database or Permanent Database.
b) Start Index—inclusive starting index of the desired entry range.
c) Stop Index—inclusive ending index of the desired range.

14.7.6.4.3 Outputs

a) Start Index—inclusive starting index of the returned entry range.
b) Stop Index—inclusive ending index of the returned entry range.
c) For each index of the returned entry range, the following are returned:
   1) Address—MAC Address of the desired entry.
   2) Type—Static or Dynamic entry.
   3) Port Map—A set of control indicators as appropriate to the entry type, as specified in 7.9.1 through 7.9.3.

14.8 Spanning Tree Protocol Entity

The Spanning Tree Protocol Entity is described in 7.10, Clause 8, and Clause 17. The objects that comprise this managed resource are as follows:

a) The Protocol Entity itself, and
b) The Ports under its control.

14.8.1 The Protocol Entity

The Protocol Entity object models the operations that can be performed upon, or inquire about, the operation of the Spanning Tree Protocol. There is a single Protocol Entity per Bridge; it can, therefore, be identified as a single fixed component of the Protocol Entity resource.

The management operations that can be performed on the Protocol Entity are Read Spanning Tree Protocol Parameters and Set Spanning Tree Protocol Parameters.

14.8.1.1 Read Spanning Tree Protocol parameters

14.8.1.1.1 Purpose

To obtain information regarding the Bridge's Spanning Tree Protocol Entity.

14.8.1.1.2 Inputs

None.

14.8.1.1.3 Outputs

a) Bridge Identifier—as defined in 17.18.3.
b) Time Since Topology Change—the count in seconds of the time since the tcWhile timer (17.17.8) for any Port was non-zero.
c) Topology Change Count—the count of times that there has been at least one non-zero tcWhile timer (17.17.8).
d) Topology Change— asserted if the tcWhile timer (17.17.8) for any Port is non-zero.
e) Designated Root (17.18.6).
f) Root Path Cost (17.18.6).
g) Root Port (17.18.6).
h) Max Age (17.18.7).
i) Hello Time (17.13.6).
j) Forward Delay (17.13.5).
k) Bridge Max Age (17.18.4).
l) Bridge Hello Time (17.18.4).
m) Bridge Forward Delay (17.18.4).
n) TxHoldCount (17.13.12).
o) forceVersion (17.13.4).

14.8.1.2 Set Spanning Tree Protocol parameters

14.8.1.2.1 Purpose

To modify parameters in the Bridge's Spanning Tree Protocol Entity in order to force a configuration of the spanning tree and/or tune the reconfiguration time to suit a specific topology, this operation causes these values to be set for all Ports of the Bridge.

14.8.1.2.2 Inputs

a) Bridge Max Age—the new value (17.18.4).
b) Bridge Hello Time—the new value (17.18.4).
c) Bridge Forward Delay—the new value (17.18.4).
d) Bridge Priority—the new value of the priority part of the Bridge Identifier (17.18.3).
e) forceVersion—the new value of the Force Protocol Version parameter (17.13.4).

f) TxHoldCount— the new value of TxHoldCount (17.13.12).

14.8.1.2.3 Outputs

a) Operation status. This takes one of the following values:
   1) Operation rejected due to invalid Bridge Priority value (14.1); or
   2) Operation accepted.

14.8.1.2.4 Procedure

The input parameter values are checked for compliance with 17.14. If they do not comply, or the value of any of Bridge Max Age, Bridge Forward Delay, or Bridge Hello Time is outside the range specified in Table 17-1, no action shall be taken for any of the supplied parameters.

Otherwise:

a) The Bridge's Bridge Max Age, Bridge Hello Time, and Bridge Forward Delay parameters are set to the supplied values.
b) The priority component of the Bridge Identifier (17.18.3) is updated using the supplied value as specified in 17.13.

14.8.2 Bridge Port

A Bridge Port object models the operations related to an individual Bridge Port in relation to the operation of the Rapid Spanning Tree Protocol. There are a fixed set of Bridge Ports per Bridge; each can, therefore, be identified by a permanently allocated Port Number, as a fixed component of the Protocol Entity resource. The management operations that can be performed on a Bridge Port are Read Port Parameters, Force Port State, Set Port Parameters, and Force BPDU Migration Check.

14.8.2.1 Read Port Parameters

14.8.2.1.1 Purpose

To obtain information regarding a specific Port within the Bridge's Spanning Tree Protocol Entity.

14.8.2.1.2 Inputs

a) Port Number—the number of the Bridge Port.

14.8.2.1.3 Outputs

a) Uptime—count in seconds of the time elapsed since the Port was last reset or initialized.
b) State—the current state of the Port (i.e., Discarding, Forwarding, or Blocking) (7.4).
c) Port Identifier—the unique Port identifier comprising two parts, the Port Number and the Port Priority field (17.19.19).
d) Path Cost (17.13.11).
e) Designated Root (17.19.21).
f) Designated Cost (17.19.21).
g) Designated Bridge (17.19.21).
h) Designated Port (17.19.21).
i) Topology Change Acknowledge (17.19.41).
j) adminEdgePort (17.13.1). Present in implementations that support the identification of edge ports.
k) operEdgePort (17.19.17). Present in implementations that support the identification of edge ports.

l) autoEdgePort (17.13.3). Optional and provided only by RSTP Bridges that support the automatic identification of edge ports.
m) MAC Enabled—the current state of the MAC Enabled parameter (6.4.2). Present if the implementation supports the MAC Enabled parameter.
n) MAC Operational—the current state of the MAC Operational parameter (6.4.2). Present if the implementation supports the MAC Operational parameter.
o) adminPointToPointMAC—the current state of the adminPointToPointMAC parameter (6.4.3). Present if the implementation supports the adminPointToPointMAC parameter.
p) operPointToPointMAC—the current state of the operPointToPointMAC parameter (6.4.3). Present if the implementation supports the operPointToPointMAC parameter.

14.8.2.2 Force port state

14.8.2.2.1 Purpose

To set the Administrative Bridge Port state (see 7.4) for the specified Port to Disabled or Enabled.

14.8.2.2.2 Inputs

a) Port Number—the number of the Bridge Port.
b) State—either Disabled or Enabled.

14.8.2.2.3 Outputs

None.

14.8.2.2.4 Procedure

The effect of changing this parameter is defined by 17.19.18 and the state machines specified in Clause 17.

14.8.2.3 Set Port Parameters

14.8.2.3.1 Purpose

To modify parameters for a Port in the Bridge’s Spanning Tree Protocol Entity in order to force a configuration of the spanning tree.

14.8.2.3.2 Inputs

a) Port Number—the number of the Bridge Port.
b) Path Cost—the new value (17.13.11).
c) Port Priority—the new value of the priority field for the Port Identifier (17.19.21).
d) adminEdgePort—the new value of the adminEdgePort parameter (17.13.1). Present in implementations that support the identification of edge ports.
e) autoEdgePort — the new value of the autoEdgePort parameter (17.13.3). Optional and provided only by RSTP Bridges that support the automatic identification of edge ports.
f) MAC Enabled—the new value of the MAC Enabled parameter (6.4.2). May be present if the implementation supports the MAC Enabled parameter.
g) adminPointToPointMAC—the new value of the adminPointToPointMAC parameter (6.4.3). May be present if the implementation supports the adminPointToPointMAC parameter.

14.8.2.3.3 Outputs

a) Operation status. This takes one of the following values:

1) Operation rejected due to invalid Port Priority value (14.3); or
2) Operation accepted.

14.8.2.3.4 Procedure

The Path Cost (17.13.11) and Port Priority (17.19.21) parameters for the Port are updated using the supplied values. The reselect parameter for the Port (17.19.34) is set TRUE, and the selected parameter for the Port (17.19.36) is set FALSE.

14.8.2.4 Force BPDU Migration Check

14.8.2.4.1 Purpose

To force the specified Port to transmit RST BPDUs (see 17.24).

14.8.2.4.2 Inputs

a) Port Number—the number of the Bridge Port.

14.8.2.4.3 Outputs

None.

14.8.2.4.4 Procedure

The mcheck variable (17.19.13) for the specified Port is set to the value TRUE if the value of the forceVersion variable (17.13.4) is greater than or equal to 2.

14.9 GARP Entities

The operation of GARP is described in Clause 12. The objects that comprise this managed resource are

a) The GARP Timer objects.
b) The GARP Attribute Type objects.
c) The GARP State Machine objects.

14.9.1 The GARP Timers object

The GARP Timer object models the operations that can be performed upon, or inquire about, the current settings of the timers used by the GARP protocol on a given Port. The management operations that can be performed on the GARP Timers object are Read GARP Timers and Set GARP Timers.

14.9.1.1 Read GARP Timers

14.9.1.1.1 Purpose

To read the current values of the GARP Timers for a given Port.

14.9.1.1.2 Inputs

a) The Port identifier.

14.9.1.1.3 Outputs

a) Current value of JoinTime—Centiseconds.
b) Current value of LeaveTime—Centiseconds.
c) Current value of LeaveAllTime—Centiseconds.

14.9.1.2 Set GARP Timers

14.9.1.2.1 Purpose

To set new values for the GARP Timers for a given Port.

14.9.1.2.2 Inputs

a) The Port identifier.
b) New value of JoinTime—Centiseconds.
c) New value of LeaveTime—Centiseconds.
d) New value of LeaveAllTime—Centiseconds.

14.9.1.2.3 Outputs

None.

14.9.2 The GARP Attribute Type object

The GARP Attribute Type object models the operations that can be performed upon, or inquire about, the operation of GARP for a given Attribute Type. The management operations that can be performed on a GARP Attribute Type are Read GARP Applicant Controls and Set GARP Applicant Controls.

14.9.2.1 Read GARP Applicant controls

14.9.2.1.1 Purpose

To read the current values of the GARP Applicant Administrative parameters (12.8.2) associated with all GARP Participants for a given Port, GARP Application, and Attribute Type.

14.9.2.1.2 Inputs

a) The Port identifier.
b) The GARP Application address (Table 12-1).
c) The Attribute Type (12.10.2.2).

14.9.2.1.3 Outputs

a) The current Applicant Administrative Control Value (12.8.2).
b) Failed Registrations—Count of the number of times that this GARP Application has failed to register an attribute of this type due to lack of space in the Filtering Database.

14.9.2.2 Set GARP Applicant controls

14.9.2.2.1 Purpose

To set new values for the GARP Applicant Administrative control parameters (12.8.2) associated with all GARP Participants for a given Port, GARP Application, and Attribute Type.

14.9.2.2.2 Inputs

a) The Port identifier.
b) The GARP Application address (Table 12-1).
c) The Attribute Type (12.10.2.2).
d) The desired Applicant Administrative Control Value (12.8.2).

14.9.2.2.3 Outputs

None.

14.9.3 The GARP State Machine object

The GARP State Machine object models the operations that can be performed upon, or inquire about, the operation of GARP for a given State Machine. The management operation that can be performed on a GARP State Machine is Read GARP State.

14.9.3.1 Read GARP State

14.9.3.1.1 Purpose

To read the current value of an instance of a GARP State Machine.

14.9.3.1.2 Inputs

a) The Port identifier.
b) The GARP Application address (Table 12-1).
c) The GIP Context (12.2.4).
d) The Attribute Type (12.10.2.2) associated with the State Machine.
e) The Attribute Value (12.10.2.6) associated with the State Machine.

14.9.3.1.3 Outputs

a) The current value of the combined Applicant and Registrar state machine for the attribute (Table 12-6).
b) Optionally, Originator address—the MAC Address of the originator of the most recent GARP PDU that was responsible for causing a state change in this state machine (12.8.1).

14.10 GMRP entities

The following managed object defines the semantics of the management operations that can be performed upon the operation of GMRP in a Bridge:

a) The GMRP Configuration managed object (14.10.1).

14.10.1 GMRP Configuration managed object

The GMRP Configuration managed object models operations that modify, or inquire about, the overall configuration of the operation of GMRP. There is a single GMRP Configuration managed object per Bridge.

The management operations that can be performed on the GMRP Configuration managed object are as follows:

a) Read GMRP Configuration (14.10.1.1).
b) Notify Group registration failure (14.10.1.2).
c) Configure Restricted_Group_Registration parameters (14.10.1.3).

14.10.1.1 Read GMRP Configuration

14.10.1.1.1 Purpose

To obtain general GMRP configuration information from a Bridge.

14.10.1.1.2 Inputs

None.

14.10.1.1.3 Outputs

a) For each Port:
    1) Port number.
    2) State of the Restricted_Group_Registration parameter (10.3.2.3 in IEEE Std 802.1D), TRUE or FALSE.

14.10.1.2 Notify Group registration failure

14.10.1.2.1 Purpose

To notify a manager that GMRP has failed to register a given Group owing to lack of resources in the Filtering Database for the creation of a Group Registration Entry (7.9.3).

14.10.1.2.2 Inputs

None.

14.10.1.2.3 Outputs

a) The MAC address of the Group that GMRP failed to register.
b) The Port number of the Port on which the registration request was received.
c) The reason for the failure:
    1) Lack of Resources; or
    2) Registration Restricted.

14.10.1.3 Configure Restricted_Group_Registration parameters

14.10.1.3.1 Purpose

To configure the Restricted_Group_Registration parameter (10.3.2.3) associated with one or more Ports.

14.10.1.3.2 Inputs

a) For each Port to be configured, a Port number and the value of the Restricted_Group_Registration parameter. The permissible values of this parameter are (as defined in 10.3.2.3):
   1) TRUE;
   2) FALSE.

14.10.1.3.3 Outputs

None.

15. Management protocol

In IEEE Std 802.1D, 1998 Edition, this clause contained managed object definitions for use with CMIP defined using the GDMO object specification language. As the preponderance of LAN management implementations assume the use of SNMP rather than CMIP as the management protocol, the use of GDMO-based managed object definitions is no longer supported by this standard and is deprecated. The definition of the SNMP MIB for MAC Bridges can be found in IETF RFC 1493 and IETF RFC 2674.

16. Bridge performance

This clause specifies a set of parameters that represent the performance of a Bridge. These parameters have been selected to allow a basic level of confidence to be established in a Bridge, for use in an initial determination of its suitability for a given application. They cannot be considered to provide an exhaustive description of the performance of a Bridge. It is recommended that further performance information be provided and sought concerning the applicability of a Bridge implementation.

The following set of performance parameters is defined:

a) Guaranteed Port Filtering Rate, and a related time interval T_F , that together characterize the traffic for which filtering is guaranteed.
b) Guaranteed Bridge Relaying Rate, and a related time interval T_R.

16.1 Guaranteed Port Filtering Rate

For a specific Bridge Port, a valid Guaranteed Port Filtering Rate, in frames per second, is a value that, given any set of frames from the specific Bridge Port to be filtered during any T_F interval, the Forwarding Process shall filter all of the set as long as all of the following are true:

a) The number of frames in the set does not exceed the specific Bridge Port's Guaranteed Port Filtering Rate multiplied by T_F.
b) The Guaranteed Port Filtering Rate of each of the other Bridge Port(s) is not exceeded.
c) The Guaranteed Bridge Relaying Rate is not exceeded.
d) Relayed frames are not discarded due to output congestion (7.7.3).
e) The information upon which the filtering decisions are based has been configured in the Filtering Database prior to the start of time interval T_F.

16.2 Guaranteed Bridge Relaying Rate

For a Bridge, a valid Guaranteed Bridge Relaying Rate, in frames per second, is a value that given any set of frames from the specific Bridge Port to be relayed during any T_R interval, the Forwarding Process shall relay all of the set as long as all of the following are true:

a) The number of frames in the set does not exceed the Bridge's Guaranteed Bridge Relaying Rate multiplied by T_R.
b) The Guaranteed Port Filtering Rate of each Bridge Port is not exceeded.
c) Relayed frames are not discarded due to output congestion (7.7.3).
d) The information upon which the forwarding decisions are based has been configured in the Filtering Database prior to the start of time interval T_R.

17. Rapid Spanning Tree Protocol (RSTP)

The Rapid Spanning Tree Algorithm Protocol configures full, simple, and symmetric connectivity throughout a Bridged Local Area Network that comprises individual LANs interconnected by Bridges.

NOTE—RSTP supersedes the Spanning Tree Algorithm and Protocol (STP) specified in Clause 8 of previous revisions of this standard. RSTP interoperates with STP to facilitate migration. Bridges conforming to either specification can be used in the same network without configuration restrictions beyond those previously imposed by STP. However, such mixed networks will not always provide rapid reconfiguration.

17.1 Protocol design requirements

RSTP operates in Bridged Local Area Networks comprising individual point-to-point or shared media LANs arbitrarily interconnected by Bridges, each operating RSTP, MSTP (IEEE Std 802.1Q), or STP. RSTP supports, preserves, and maintains the quality of the MAC Service in all its aspects, as specified by Clause 6, meeting the following requirements:

a) It configures the Port State (7.4) of each Bridge Port, selecting some Bridge Ports to forward frames and others to discard frames, to provide a fully (spanning) and simply (tree) connected active topology.

NOTE—The configured active topology provides symmetric connectivity, i.e., frames transmitted from any given end station A to any other end station B traverse the same Bridge Ports (in reversed order) as those transmitted from B to A, thus allowing learning from a frame’s source address to select the path for subsequent frames addressed to that source.

b) It provides for fault tolerance by automatic reconfiguration of the active topology as a result of the failure of LAN components, and accommodates addition of any Bridge or Bridge Port without the formation of transient data loops (6.1, 6.3.3, 6.3.4).

c) The active topology will, with a high probability, stabilize within a short, known bounded interval, minimizing the time for which the service is unavailable for communication between any pair of end stations (6.1).

d) The active topology will be predictable and reproducible, and may be selected by management of the parameters of the algorithm, thus allowing the application of Configuration Management, following traffic analysis, to meet the goals of Performance Management (6.1 and 6.3.10).

e) It operates transparently to the end stations, such that they are unaware of their attachment to a single LAN or a bridged LAN when using the MAC Service (6.2).

f) The communications bandwidth consumed by RSTP on any particular LAN is always a very small fraction of the total available bandwidth and is independent of the total traffic supported by the network regardless of the total number of Bridges or LANs (6.3.10).

Additionally, the algorithm and protocol meet the following goals, which limit the complexity of Bridges and their configuration:

g) The memory requirements associated with each Bridge Port are independent of the number of Bridges and LANs in the network.

h) Bridges do not have to be individually configured before being added to the network, other than having their MAC Addresses assigned through normal procedures.

i) In normal operation, the time taken to configure the active topology of a network comprising point-to-point LANs is independent of the timer values of the protocol.

17.2 Protocol support requirements

In order for the Bridge Protocol to operate, the following are required:

a) A unique Group MAC Address, recognized by all the Bridges attached to a LAN, that identifies the Spanning Tree Protocol Entities (7.12.3) of the Bridges.
b) An identifier for each Bridge, unique within the Bridged Local Area Network.
c) A identifier for each Bridge Port, unique within a Bridge.

Values for each of these parameters shall be provided by each Bridge. The unique MAC Address that identifies the Spanning Tree Protocol Entities is the Bridge Group Address (7.12.3).

To allow management of the active topology, means of assigning values to the following are required:

d) The relative priority of each Bridge in the network.
e) The relative priority of each Port of a Bridge.
f) A Port Path Cost for each Port.

17.3 RSTP overview

The Rapid Spanning Tree Protocol (RSTP) configures the Port State (7.4) of each Bridge Port in the Bridge Local Area Network. RSTP ensures that the stable connectivity provided by each Bridge between its Ports and by the individual LANs to which those Ports attach is predictable, manageable, full, simple, and symmetric. RSTP further ensures that temporary loops in the active topology do not occur if the network has to reconfigure in response to the failure, removal, or addition of a network component, and that erroneous station location information is removed from the Filtering Database after reconfiguration.

NOTE—The Rapid Spanning Tree Algorithm and Protocol cannot protect against temporary loops caused by the interconnection of two LAN segments by devices other than Bridges (e.g., LAN repeaters) that operate invisibly with respect to support of the Bridges' MAC Internal Sublayer Service.

Each of the Bridges in the network transmits Configuration Messages (17.8). Each Configuration Message contains spanning tree priority vector (17.5) information that identifies one Bridge as the Root Bridge of the network, and allows each Bridge to compute its own lowest path cost to that Root Bridge, information that will in turn be transmitted in Configuration Messages. A Port Role (17.7) of Root Port is assigned to the one Port on each Bridge that provides that lowest cost path to the Root Bridge, and a Port Role of Designated Port to the one Port attached to each LAN that provides the lowest cost path from that LAN to the Root Bridge. Port roles of Alternate Port and Backup Port are assigned to Bridge Ports that can provide connectivity if other network components fail.

State machines associated with the Port Roles maintain and change the Port States (7.4) that control forwarding (7.7) and learning (7.8) of frames by a MAC Relay Entity (7.3), supporting (6.1) and maintaining the quality (6.3) of the MAC Service. In a stable network, Root Ports and Designated Ports are Forwarding, while Alternate, Backup, and Disabled Ports are Discarding.

Each Port's role can change if a Bridge, Bridge Port, or LAN fails, is added to, or removed from network. Port state transitions to Learning and Forwarding are delayed, and ports can temporarily transition to the Discarding state to ensure that misordering (6.3.3) and duplication (6.3.4) rates remain negligible.

RSTP provides rapid recovery of connectivity to minimize frame loss (6.3.2). A new Root Port, and Designated Ports attached to point-to-point LANs, can transition to Forwarding without waiting for protocol timers to expire. A Root Port can transition to Forwarding without transmitting or receiving messages from other Bridges, while a Designated Port attached to a point-to-point LAN can transition when it receives an explicit role agreement transmitted by the other Bridge attached to that LAN. The forwarding transition delay used by a Designated Port attached to a shared media LAN is long enough for other Bridges attached to that LAN to receive and act on transmitted messages, but is independent of the overall network size. If all the LANs in a network are point-to-point, RSTP timers define worst-case delays that only occur if protocol messages are lost or rate transmission limits are exceeded.

A Bridge Port attached to a LAN that has no other Bridges attached to it may be administratively configured as an Edge Port. RSTP monitors the LAN to ensure that no other Bridges are connected, and may be configured to automatically detect an Edge Port. Each Edge Port transitions directly to the Forwarding Port State, since there is no possibility of it participating in a loop.

17.3.1 Computation of the active topology

The Bridge with the best Bridge Identifier is selected as the Root Bridge. The unique Bridge Identifier for each Bridge is derived, in part, from the Bridge Address (7.12.5) and, in part, from a manageable priority component (9.2.5, 17.18.3, 17.14). The relative priority of Bridges is determined by the numerical comparison of the unique identifiers, with the lower numerical value indicating the better identifier.

Every Bridge has a Root Path Cost associated with it. For the Root Bridge this is zero. For all other Bridges, it is the sum of the Port Path Costs on the least cost path to the Root Bridge. Each Port’s Path Cost may be managed, 17.14 recommends default values for Ports attached to LANs of various speeds.

The Port on each Bridge with the lowest Root Path Cost is assigned the role of Root Port for that Bridge (the Root Bridge does not have a Root Port). If a Bridge has two or more ports with the same Root Path Cost, then the port with the best Port Identifier is selected as the Root Port. Part of the Port Identifier is fixed and is different for each Port on a Bridge, and part is a manageable priority component (9.2.7). The relative priority of Ports is determined by the numerical comparison of the unique identifiers, with the lower numerical value indicating the better identifier.

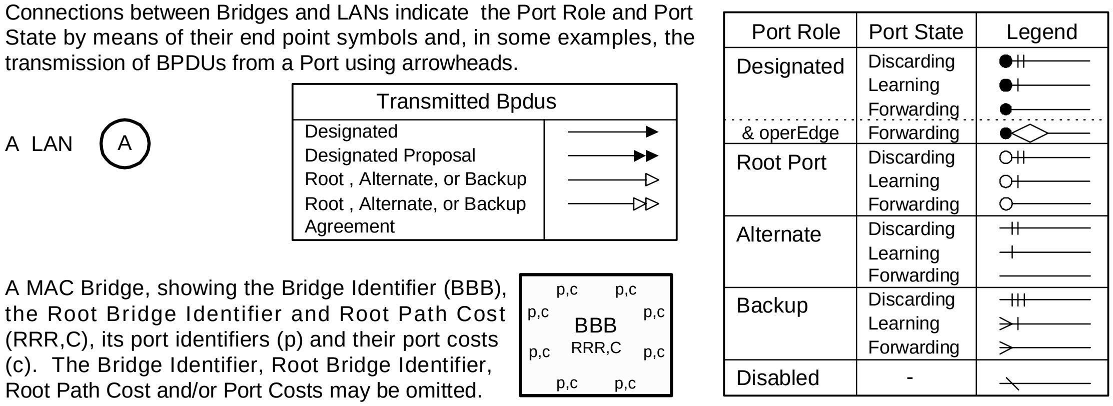
Each LAN in the Bridged Local Area Network also has an associated Root Path Cost. This is the Root Path Cost of the lowest cost Bridge with a Bridge Port connected to that LAN. This Bridge is selected as the Designated Bridge for that LAN. If there are two or more Bridges with the same Root Path Cost, then the Bridge with the best priority (least numerical value) is selected as the Designated Bridge. The Bridge Port on the Designated Bridge that is connected to the LAN is assigned the role of Designated Port for that LAN. If the Designated Bridge has two or more ports connected to the LAN, then the Bridge Port with the best priority Port Identifier (least numerical value) is selected as the Designated Port.

In a Bridged Local Area Network whose physical topology is stable, i.e RSTP has communicated consistent information throughout the network, every LAN has one and only one Designated Port, and every Bridge with the exception of the Root Bridge has a single Root Port connected to a LAN. Since each Bridge provides connectivity between its Root Port and its Designated Ports, the resulting active topology connects all LANs (is “spanning”) and will be loop free (is a “tree”).

Any operational Bridge Port that is not a Root or Designated Port is a Backup Port if that Bridge is the Designated Bridge for the attached LAN, and an Alternate Port otherwise. An Alternate Port offers an alternate path in the direction of the Root Bridge to that provided by the Bridge’s own Root Port, whereas a Backup Port acts as a backup for the path provided by a Designated Port in the direction of the leaves of the Spanning Tree. Backup Ports exist only where there are two or more connections from a given Bridge to a given LAN; hence, they (and the Designated Ports that they back up) can only exist where two ports are connected together in loopback by a point-to-point link, or where the Bridge has two or more connections to a shared media LAN.

17.3.2 Example topologies

The examples shown in this subclause make use of the diagrammatic conventions shown in Figure 17-1.



Connections between Bridges and LANs indicate the Port Role and Port State by means of their end point symbols and, in some examples, the transmission of BPDUs from a Port using arrowheads.

A LAN Ⓐ

Transmitted Bpdus	
Designated	——▶
Designated Proposal	——▶▶
Root , Alternate, or Backup	——▷
Root , Alternate, or Backup Agreement	——▷▷

A MAC Bridge, showing the Bridge Identifier (BBB), the Root Bridge Identifier and Root Path Cost (RRR,C), its port identifiers (p) and their port costs (c). The Bridge Identifier, Root Bridge Identifier, Root Path Cost and/or Port Costs may be omitted.

Port Role	Port State	Legend
Designated	Discarding	●╫——
	Learning	●┼——
	Forwarding	●——
& operEdge	Forwarding	●◇——
Root Port	Discarding	○╫——
	Learning	○┼——
	Forwarding	○——
Alternate	Discarding	╫——
	Learning	┼——
	Forwarding	——
Backup	Discarding	╫┼——
	Learning	>┼——
	Forwarding	>——
Disabled	-	—\——

Figure 17-1—Diagrammatic conventions

NOTE—These conventions allow the representation of Alternate and Backup Ports that are in Learning or Forwarding states; this can happen as a transient condition due to implementation-dependent delays in switching off Learning and/or Forwarding on a Port that changes role from Designated or Root to Alternate or Backup.

Figure 17-2 is a physical topology example showing a simple, redundantly connected, structured wiring configuration, connecting Bridges with links that form point-to-point LANs A through N. For clarity, only Bridges and LANs are shown, with the unused Bridge Ports (Ports 3 and 4 of Bridges 555 through 888) available for connecting further devices to the network.

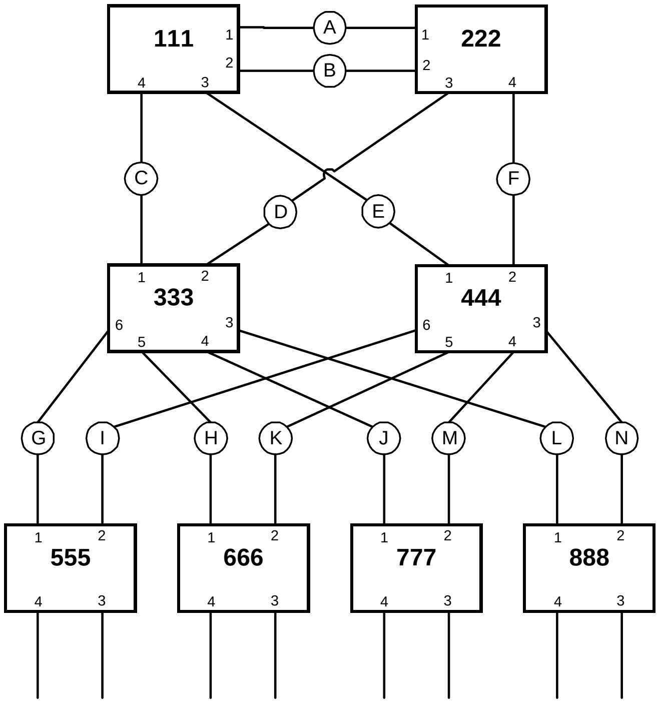


Figure 17-2—Physical topology example

Figure 17-3 shows the spanning tree active topology of the same network. Bridge 111 has been selected as the Root (though one cannot tell simply by looking at the active topology which Bridge is the Root).

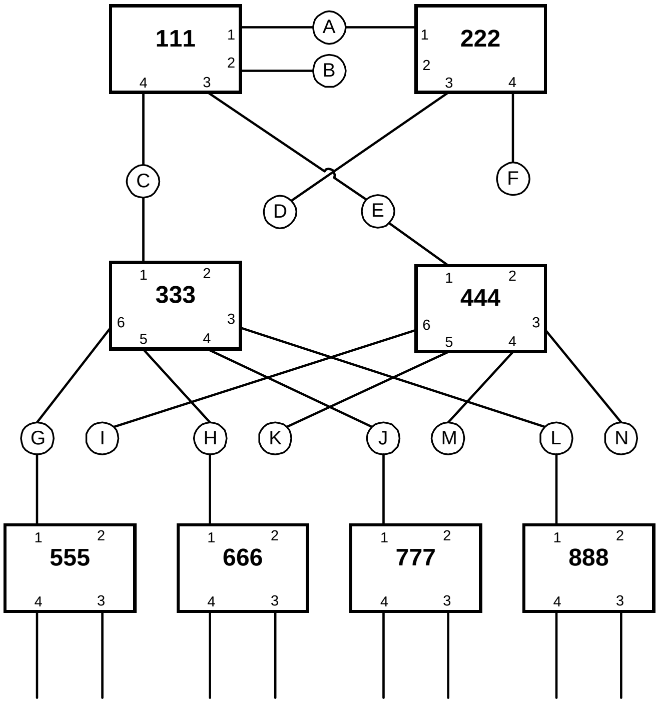


Figure 17-3—Active topology example

Figure 17-4 shows the Port Roles and Port States of each Bridge Port. It can be seen that Bridge 111 is the Root, as its Ports are all Designated Ports, each of the remaining Bridges have one Root Port.

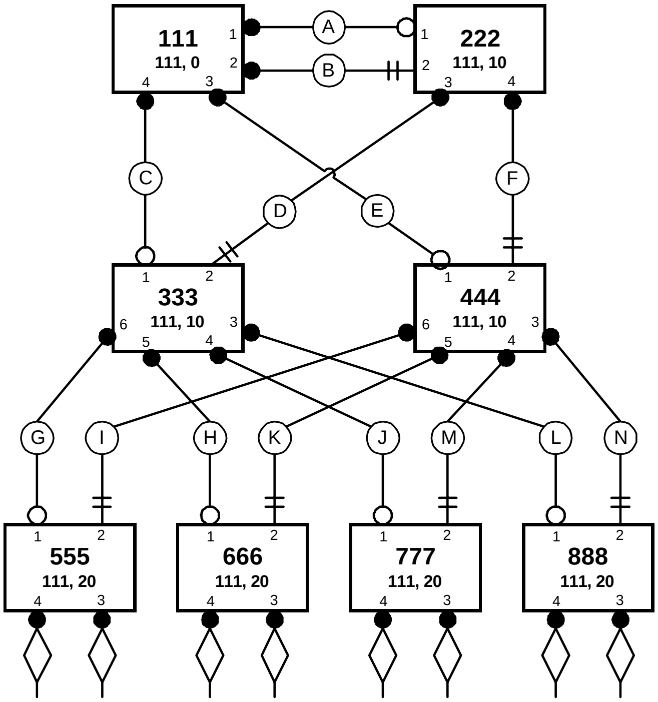


Figure 17-4—Port Roles and Port States

Figure 17-5 shows the result of connecting two of the Ports of Bridge 888 to the same LAN. As Port 4 of Bridge 888 has worse priority than Port 3 and both offer the same Root Path Cost, Port 4 will be assigned the Backup Port Role and will therefore be in the Discarding Port State. Should Port 3 or its connection to LAN O fail, Port 4 will be assigned the Designated Port Role and will transition to the Forwarding Port State.

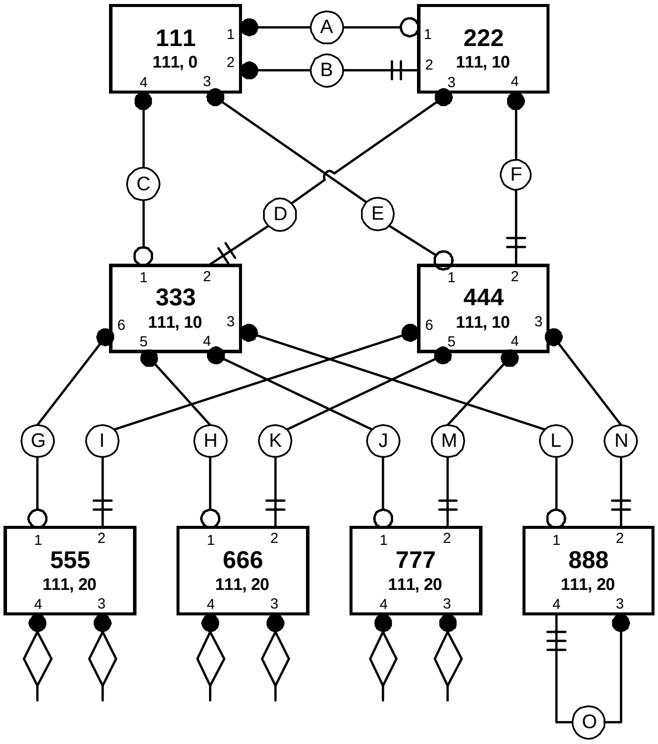


Figure 17-5—Backup Port example

Figure 17-6 shows a “ring” topology constructed from point-to-point links, as in some resilient backbone configurations. Bridge 111 is the Root, as in previous examples.

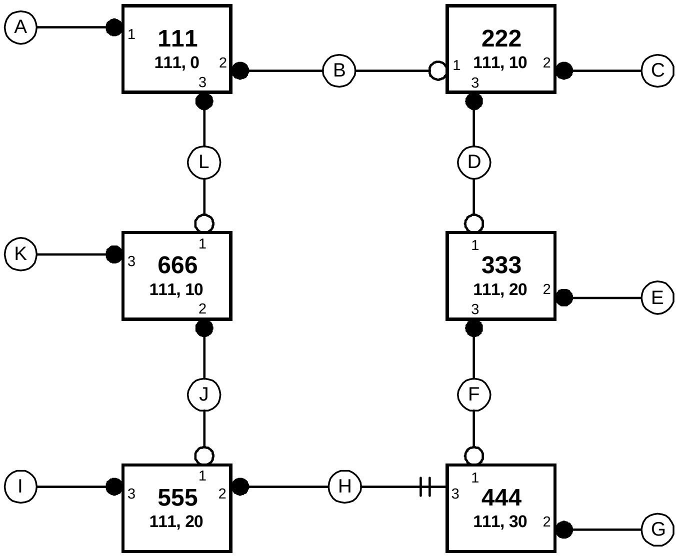


Figure 17-6—“Ring Backbone” example

17.4 STP compatibility

RSTP is designed to be compatible and interoperable with the Spanning Tree Algorithm and Protocol (STP) specified in Clause 8 of previous revisions of this standard, without additional operational management practice.

An RSTP Bridge Port automatically adjusts to provide interoperability, if it is attached to the same LAN as an STP Bridge. Protocol operation on other ports is unchanged. Configuration (9.3.1) and Topology Change Notification (9.3.2) BPDUs are transmitted instead of RST BPDUs (9.3.3), which are not recognized by STP Bridges. Port state transition timer values are increased to ensure that temporary loops are not created through the STP Bridge. Topology changes are propagated for longer to support the different Filtering Database flushing paradigm used by STP.

It is possible that RSTP's rapid state transitions will increase rates of frame duplication and misordering, as discussed in Annex G (informative). An administrative Force Protocol Version parameter (17.13.4) causes an RSTP Bridge to use STP compatible BPDUs and timer values on all Bridge Ports. Rapid transitions are disabled, supporting applications and protocols that are particularly sensitive to frame duplication and misordering.

17.5 Spanning tree priority vectors

RSTP Bridges send information to each other, in Configuration Messages (17.8), to select a Root Bridge and the shortest path to it from each LAN and each of the other Bridges. The information sent for this purpose is known as a *spanning tree priority vector*. Spanning tree priority vectors provide the basis for a concise specification of RSTP's computation of the active topology. Each priority vector comprises the following:

a) Root Bridge Identifier, the Bridge Identifier of the Bridge believed to be the Root by the transmitter
b) Root Path Cost, to that Root Bridge from the transmitting Bridge
c) Bridge Identifier, of the transmitting Bridge
d) Port Identifier, of the Port through which the message was transmitted
e) Port Identifier, of the Port through which the message was received (where relevant)

The first two components of a spanning tree priority vector are significant throughout the network; they are propagated and updated along each path in the active topology. The next two components are locally significant; they are assigned hop by hop for each LAN or Bridge and used as tie-breakers in decisions between spanning tree priority vectors that are otherwise equal. The fifth component is never conveyed in Configuration Messages, but is used as a tie-breaker within a Bridge.

The set of all spanning tree priority vectors is totally ordered. For all components, a lesser numerical value is better, and earlier components in the above list are more significant. As each Bridge Port receives a priority vector from Ports closer to the Root, additions are made to one or more components to yield a worse priority vector. This process of receiving information, adding to it, and passing it on, can be described in terms of the message priority vector received and a set of priority vectors used to facilitate the computation of a priority vector for each Port, to be transmitted in further Configuration Messages to Bridges further from the Root.

17.6 Priority vector calculations

The *port priority vector* is the spanning tree priority vector held for the port when the reception of BPDUs and any pending update of information has been completed:

port priority vector = {RootBridgeID : RootPathCost : DesignatedBridgeID :
DesignatedPortID : BridgePortID}

A *message priority vector* is the spanning tree priority vector conveyed in a received Configuration Message. For a Bridge *B* receiving a Configuration Message on Port P_B from a Designated Port P_D on Bridge *D* claiming a Root identifier of R_D and a Root Path Cost of RPC_D:

message priority vector = $\{R_D : RPC_D : D : P_D : P_B\}$

This message priority vector is superior to the port priority vector and will replace it if, and only if, the message priority vector is better than the port priority vector, or the message has been transmitted from the same Designated Bridge and Designated Port as the port priority vector, i.e., if the following is true:

((R$_D$ < RootBridgeID)) ||
((R$_D$ == RootBridgeID) && (RPC$_D$ < RootPathCost)) ||
((R$_D$ == RootBridgeID) && (RPC$_D$ == RootPathCost) && (D < DesignatedBridgeID)) ||
((R$_D$ == RootBridgeID) && (RPC$_D$ == RootPathCost)
&& (D == DesignatedBridgeID) && (P$_D$ < DesignatedPortID)) ||
((D == DesignatedBridgeID.BridgeAddress) && (P$_D$ == DesignatedPortID.PortNumber))

A *root path priority vector* can be calculated from a received port priority vector, by adding the receiving Port’s path cost PPC_{PB} to the *Root Path Cost* component.

root path priority vector= $\{R_D : RPC_D + PPC_{PB} : D : P_D : P_B\}$

The *bridge priority vector* for a Bridge *B* is the priority vector that would, with the *Designated Port ID* set equal to the transmitting *Port ID*, be used as the message priority vector in Configuration Messages transmitted on Bridge *B’s* Designated Ports if *B* was selected as the Root Bridge.

bridge priority vector = $\{B : 0 : B : 0 : 0\}$

The *root priority vector* for *B* is the best of the set comprising the bridge priority vector plus all root path priority vectors whose DesignatedBridgeID *D* is not equal to *B*. If the bridge priority vector is the best, *B* has been selected as the Root. Assuming the best root path priority vector is that of port P_B above, then:

root priority vector = $\{B : 0 : B : 0 : 0\}$ if *B* is better than R_D, or
= $\{R_D : RPC_D + PPC_{PB} : D : P_D : P_B\}$ if *B* is worse than R_D

The *designated priority vector* for a port *Q* on Bridge *B* is the root priority vector with *B’s* Bridge Identifier *B* substituted for the *DesignatedBridgeID* and *Q’s* Port Identifier Q_B substituted for the *DesignatedPortID* and *BridgePortID* components.

designated priority vector = $\{B : 0 : B : Q_B : Q_B\}$ if *B* is better than R_D, or
= $\{R_D : RPC_D + PPC_{PB} : B : Q_B : Q_B\}$ if *B* is worse than R_D

If the designated priority vector is better than the port priority vector, the Port will be the Designated Port for the attached LAN and the port priority vector will be updated. The message priority vector in RST BPDUs transmitted by a Port always comprises the first four components of the designated priority vector of the Port, even if the Port is a Root, Alternate, or Backup Port.

NOTE —The consistent use of lower numerical values to indicate better information is deliberate as the Designated Port that is closest to the Root Bridge, i.e., has a numerically lowest path cost component, is selected from amongst potential alternatives for any given LAN. Adopting the conventions that lower numerical values indicate better information, that where possible more significant priority components are encoded earlier in the octet sequence of a BPDU, and that earlier octets in the encoding of individual components are more significant allow concatenated octets that compose a priority vector to be compared as if they were a multiple octet encoding of a single number, without regard to the boundaries between the encoded components. To reduce the confusion that naturally arises from having the lesser of two numerical values represent the better of the two, i.e., the one to be chosen all other factors being equal, this clause uses the following consistent terminology. Relative numeric values are described as “least,” “lesser,” “equal,” and “greater,” and their comparisons as “less than,” “equal to,” or “greater than,” while relative Spanning Tree priorities are described as “best,” “better,” “the same,” “different,” and “worse” and their comparisons as “better than,” “the same as,” “different from,” and “worse than”. The operators “<” and “=” represent less than and equal to respectively. The terms “superior” and “inferior” are used for comparisons that are not simply based on priority but can include the fact that a Designated Port’s priority vector can replace an earlier vector transmitted by the same Bridge Port.

17.7 Port Role assignments

The Rapid Spanning Tree Algorithm assigns one of the following Port Roles to each Bridge Port: Root Port, Designated Port, Alternate Port, Backup Port, or Disabled Port.

The Disabled Port role is assigned if the port is not operational or is excluded from the active topology by management, i.e., its MAC_Operational status (6.4.2) is FALSE, or it is a network access port (IEEE Std 802.1X) and its AuthControlledPortStatus is Unauthorized, or its Administrative Bridge Port state is Disabled (14.8.2.2) (see also 7.12.7). Ports that are enabled have Port Roles assigned according to the source and relative priority of the spanning tree port priority vectors (17.5, 17.6) as follows:

a) If the Bridge is not the Root Bridge, the source of the root priority vector is the Root Port.
b) Each Port whose port priority vector is its designated priority vector is a Designated Port.
c) Each Port, other than the Root Port, whose port priority vector has been received from another Bridge is an Alternate Port.
d) Each Port that has a port priority vector that has been received from another Port on this Bridge is a Backup Port.

17.8 Communicating spanning tree information

Bridges transmit MAC frames, each containing a Bridge Protocol Data Unit (Clause 9), to communicate Spanning Tree messages. MAC frames conveying BPDUs are addressed to the Bridge Group Address, one of a small number of addresses that identify frames not forwarded by Bridges (7.12.6), and are received by all the Bridges connected to the LAN on which the frame is transmitted.

BPDUs convey Configuration and Topology Change Notification (TCN) Messages. A Configuration Message can be encoded and transmitted as a Configuration BPDU (9.3.1), or as an RST BPDU (9.3.3). A TCN Message can be encoded as a TCN BPDU (9.3.2), or as an RST BPDU (9.3.3) with the TC flag set. The Port Protocol Migration state machine (17.24) determines the BPDU types used.

Configuration Messages are transmitted if the information they convey changes, subject to a maximum transmission rate (see Transmit Hold Count in 17.14). Designated Ports also transmit Configuration Messages at regular intervals to guard against loss and to assist in the detection of failed components (LANs, Bridges, or Bridge Ports).

17.9 Changing spanning tree information

If the physical connectivity of the network or management parameters change, new spanning tree information will propagate rapidly. Each Bridge accepts better information from any Bridge on a LAN or revised information from the prior Designated Bridge for that LAN. Updated Configuration Messages are transmitted through Designated Ports, until the leaves of the Spanning Tree defined by a new configuration are reached. The immediate stimulation and transmission of information will cease as these new Configuration Messages reach Designated Ports that have already received the new information through redundant paths in the network, or reach LANs that are not redundantly connected.

To ensure that old information does not endlessly circulate through redundant paths in the network and prevent propagation of new information, each Configuration Message includes a message age and a maximum age. The message age is incremented on receipt, and the information discarded if it exceeds the maximum. Thus the number of Bridges the information can traverse is limited.

The MAC_Operational parameter (6.4.2) for each Bridge Port can signal failure conditions in some MACs. If it becomes FALSE the Port becomes a Disabled Port and received information is immediately discarded. If the Bridge has Designated Ports, changed information will be transmitted and propagated. This enables rapid reassignment of Port Roles that depended on the prior physical topology.

Not all MAC component failure conditions can be detected and signalled by changes in MAC_Operational status, so received spanning tree information is aged out if not refreshed by the regular reception of Configuration Messages from the Designated Port.

17.10 Changing Port States

The Port State of each Bridge Port is controlled by the Port Role Transitions state machine (17.29), whose goal is to maximize connectivity without introducing temporary loops. It attempts to transition Root Ports and Designated Ports to the Forwarding Port State, and Alternate Ports and Backup Ports to the Discarding Port State, as rapidly as possible.

Transitions to Discarding can be effected without the risk of data loops. Transitions to Forwarding need to be consistent with the Port Roles and States of the other Ports in the region of the network bounded by this Port and by Ports that are not Forwarding or that attach to a LAN with no other attached Bridges.

A Bridge knows that a Port can become Forwarding, following a change in spanning tree information that causes it to be assigned a Root or Designated Port Role, if:

a) Enough time has elapsed for the information to reach all Bridges in the region, and for contradictory information to be received from any Bridge in the region, or
b) The Port is now a Root Port, and any Ports on the Bridge that have been Root Port are not and will not become Forwarding [with the exception of item c) below] until the spanning tree information reaches all the other Bridges in the network, or
c) The Port is a Designated Port attached, via a LAN, to at most one other Bridge whose Port States are either consistent with the spanning tree information or Discarding, as are the Port States of the further Bridges connected through their Forwarding Ports, or
d) The Port is an Edge Port (17.3, 17.19.17).

Figure 17-7 illustrates conditions b) and c). As a result of management of the Port priorities of Bridge 222, an Alternate Port becomes its new Root Port, and the old Root Port an Alternate Port. Assuming that the initial configuration had been stable for the necessary time, and that the old Root Port’s State is now Discarding, the new Root Port can be made Forwarding, by applying condition b).

If the Designated Port attached to LAN G is not Forwarding for some reason (a very recent transition from administratively disabled to enabled, for example), Bridge 111 will exchange messages with Bridge 222 to satisfy condition c).

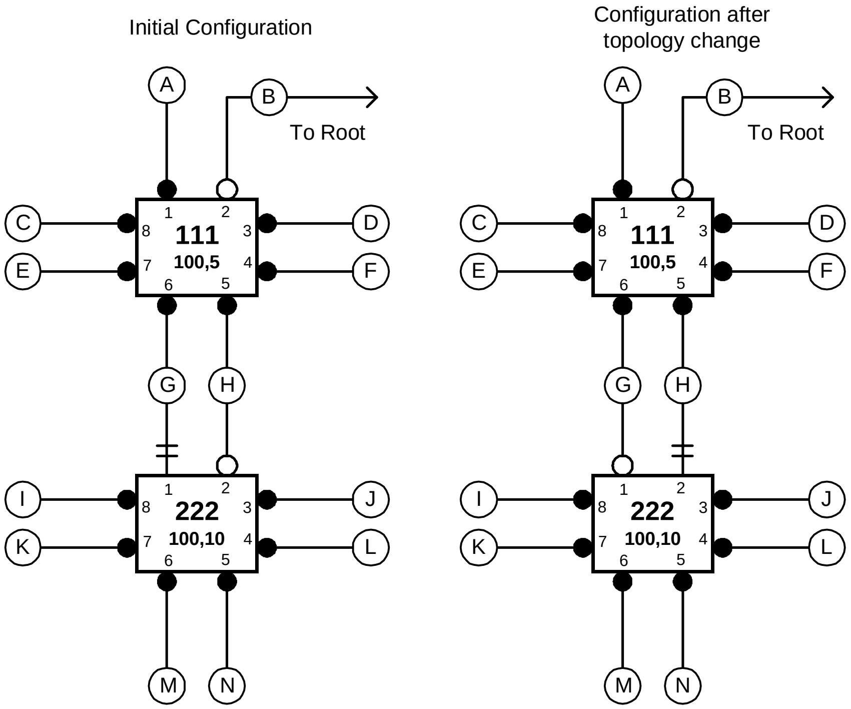


Figure 17-7—Root Port transition example

The message exchange used by the Port Role Transitions state machine (17.29) to transition a Designated Port to Forwarding is illustrated in Figure 17-8. It uses the following Boolean state machine variables:

a) **proposing (17.19.24).** Set by a Designated Port that is not Forwarding, and conveyed to the Root Port or Alternate Port of a neighboring Bridge in the Proposal flag of an RST BPDU (9.3.3).

b) **proposed (17.19.23).** Set when an RST BPDU with a Designated Port role and the Proposal flag set is received. If **agree** is not set, **proposed** causes **sync** to be set for all other Ports.of the Bridge.

c) **sync (17.19.39).** If set the Port transitions to Discarding, unless it is an Edge Port or **synced** is set.

d) **synced (17.19.40).** Set when the Port is Discarding or **agreed** is set.

e) **agree (17.19.3).** Set if **synced** is set for all other Ports. An RST BPDU with the Agreement flag set is transmitted and **proposed** is reset when **agree** is first set, and when **proposed** is set.

f) **agreed (17.19.3).** Set when an RST BPDU is received with a Port Role of Root, Alternate, or Backup Port, the Agreement flag set, and a message priority the same or worse than the port priority. When **agreed** is set, the Designated Port knows that its neighbouring Bridge has confirmed that it can proceed to the Forwarding state without further delay.

Any Designated Port that transitions to Discarding requests permission to transition to Forwarding in turn from its neighboring Bridge. The effect is that a “cut” in the active topology is propagated away from the Root Bridge until it reaches its final position in the stable active topology or the edge of the network.

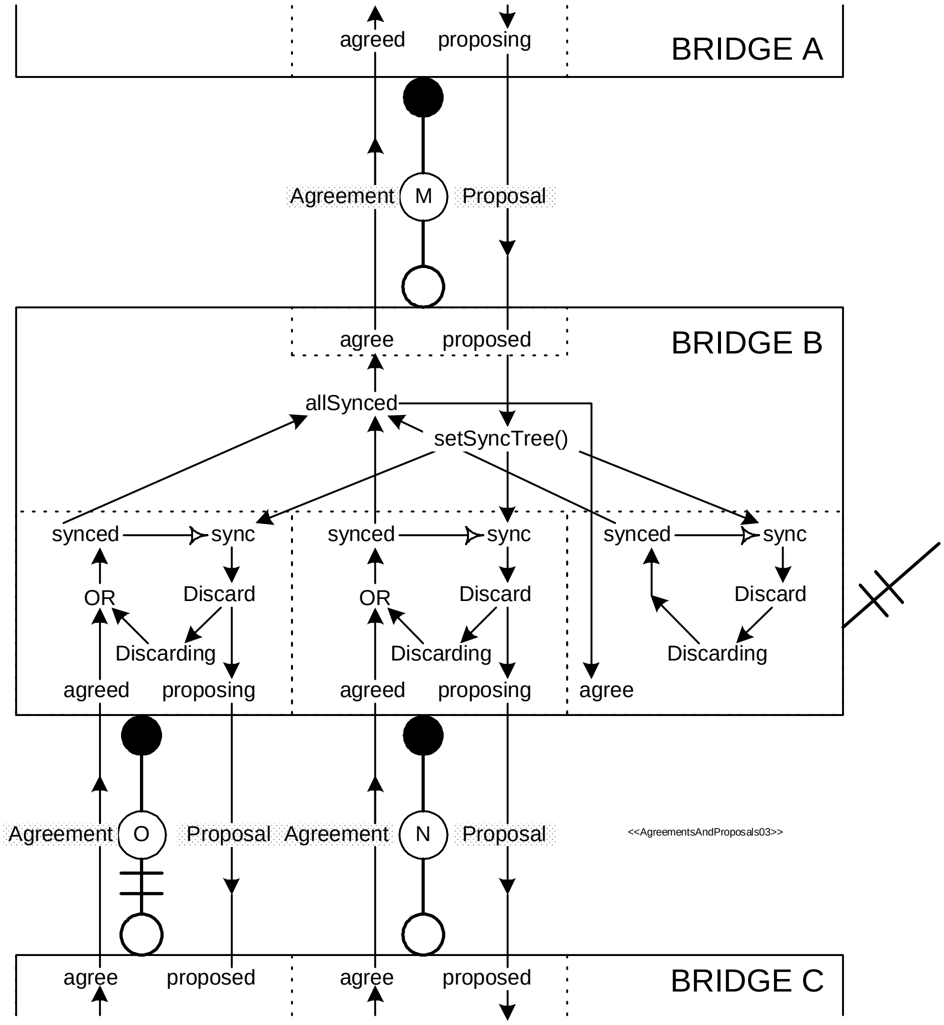


Figure 17-8—Agreements and proposals

17.11 Updating learned station location information

In normal stable operation, learned station location information held in the Filtering Database need only change as a consequence of the physical relocation of stations. It is therefore desirable to employ a long ageing time for Dynamic Filtering Entries (7.9.2), especially as many end stations transmit frames following power-up causing the information to be relearned.

However, when the active topology reconfigures, end stations can appear to move from the point of view of a Bridge even if that Bridge's Port States have not changed. Information has to be relearned when Ports become or cease to be part of the active topology, even if only part of the network has reconfigured.

If a Port is no longer part of the active topology, stations are no longer reachable through that Port, and its Dynamic Filtering Entries are removed from the Filtering Database. Conversely, stations formerly reachable through other ports might be reachable through a newly active Port. Dynamic Filtering Entries for the other Ports are removed and Topology Change Notification Messages are transmitted both through the newly active Port and through the other active Ports on that Bridge. A Bridge that receives a TCN on an active port removes Dynamic Filtering Entries for their other active Ports and propagates the TCN through those Ports.

The Topology Change state machine (17.31) avoids removing Dynamic Filtering Entries throughout the network when Designated Ports temporarily revert to Discarding to suppress loops. It treats a Port as becoming active in the network topology once it transitions to Forwarding, and no longer active when it becomes an Alternate, Backup, or Disabled Port and stops learning from received frames. The Topology Change state machine does not generate TCNs following Edge Port (17.3, 17.19.17) Port State changes, as these do not affect connectivity or station location information in the rest of the network, nor does it remove Dynamic Filtering Entries for Edge Ports when TCNs are received from other Bridges.

NOTE—The rules described require removal of potentially invalid learned information for the minimum set of Ports on each Bridge. A Bridge can flush information from more Ports than strictly necessary if desired for implementation reasons. For example, a Bridge can choose to remove all learned addresses from its Filtering Database rather than just the addresses learned on the specified Ports. This does not result in incorrect operation, as it simply returns the Filtering Database to its initial state before any information was learned; however, it can result in more flooding of frames with unknown destination addresses than is necessary for correct operation.

Figure 17-9 and Figure 17-10 illustrate flushing of learned addresses following a topology change.

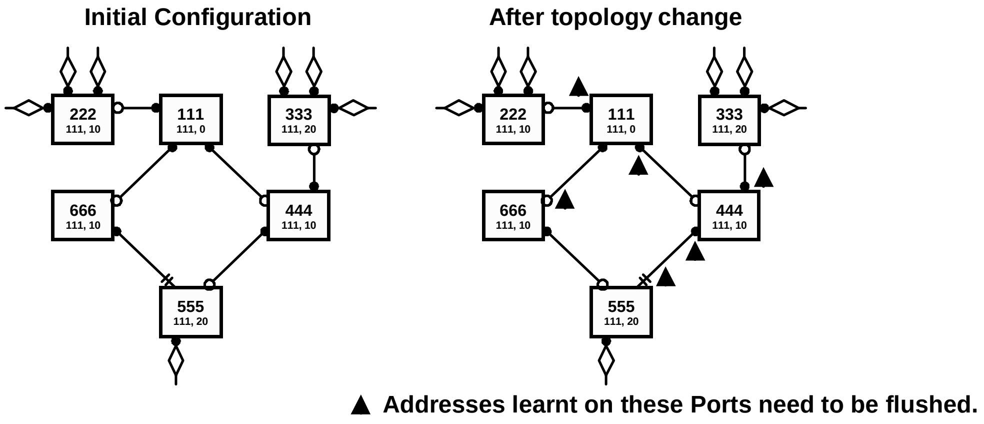


Figure 17-9—Address flushing example

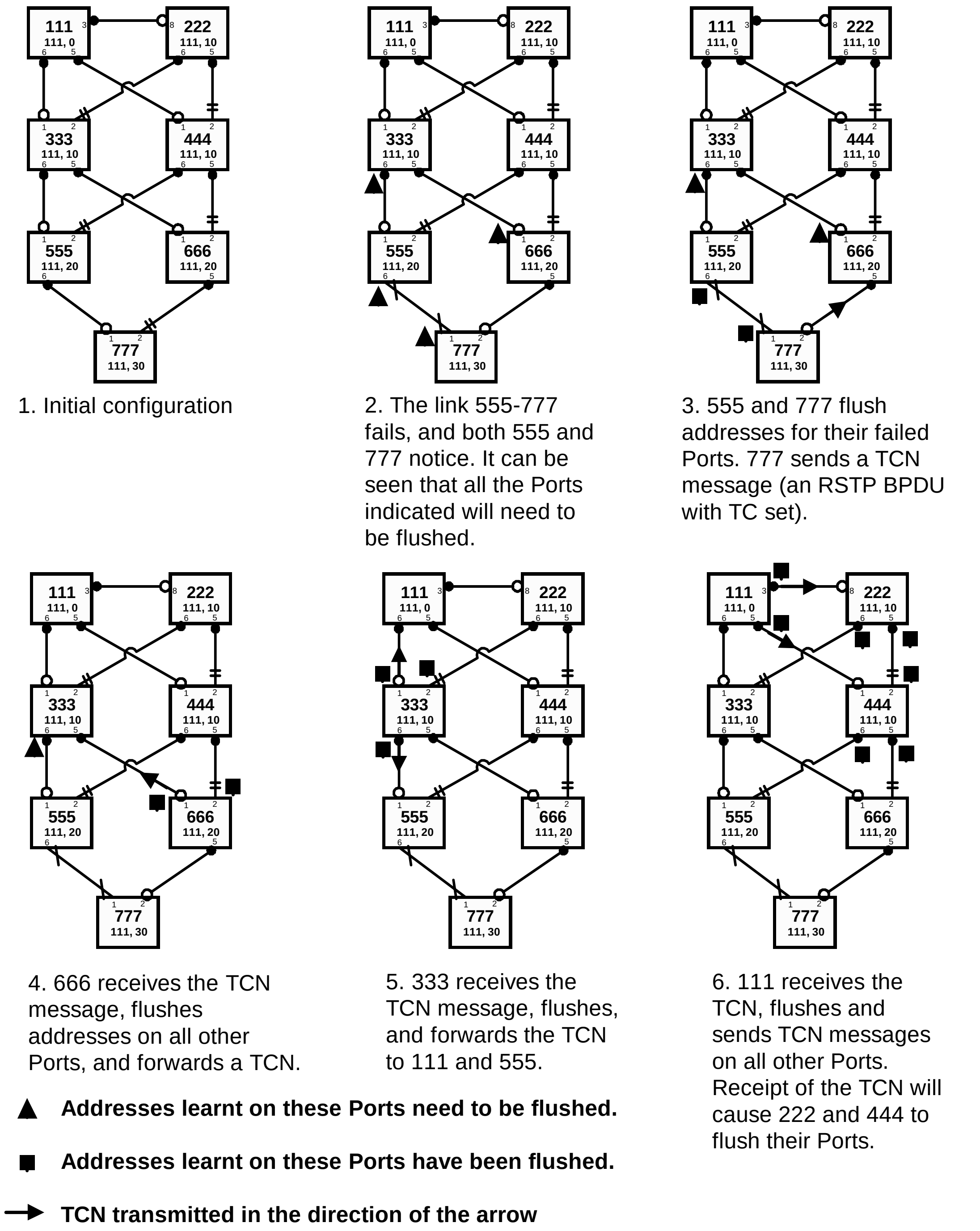
In Figure 17-9, Bridge 555 selects a new Root Port, changing its Port State from Discarding to Forwarding. Any addresses learned on the former Root Port, which is now Discarding, have to be flushed as do addresses learned on other Bridges' Ports from frames forwarded through that former Root Port. A Bridge receiving a TCN can be only be certain that it does not need to flush addresses learned on the Port receiving the TCN and on Edge Ports (17.19.17). Dynamic Filtering Entries for all other Ports are removed.

In Figure 17-10, a structured wiring configuration example illustrates the sequence of events following the loss of a link towards the periphery of the network. This example assumes that all Bridges are RSTP Bridges so all TCN messages are transmitted as RST BPDUs with the TC flag set (see 17.8).

All MAC addresses previously learned on a Root Port can be moved to an Alternate Port that becomes the new Root Port; i.e., Dynamic Filtering Entries for those addresses may be modified to show the new Root Port as their source, reducing the need to flood frames when recovering from some component failures. This optional optimization is possible because a retiring Root Port that becomes Discarding temporarily partitions the active topology into:

a) A main tree containing all Bridges and LANs hitherto reachable through the retiring Root Port, and
b) A subtree containing the retiring Root Port's Bridge and all other LANs and Bridges between that Bridge and the edge of the network.

The new Root Port once more provides the path from stations in the subtree to those in the main tree.



1. Initial configuration

2. The link 555-777 fails, and both 555 and 777 notice. It can be seen that all the Ports indicated will need to be flushed.

3. 555 and 777 flush addresses for their failed Ports. 777 sends a TCN message (an RSTP BPDU with TC set).

4. 666 receives the TCN message, flushes addresses on all other Ports, and forwards a TCN.

5. 333 receives the TCN message, flushes, and forwards the TCN to 111 and 555.

6. 111 receives the TCN, flushes and sends TCN messages on all other Ports. Receipt of the TCN will cause 222 and 444 to flush their Ports.

▲ **Addresses learnt on these Ports need to be flushed.**

■ **Addresses learnt on these Ports have been flushed.**

→ **TCN transmitted in the direction of the arrow**

Figure 17-10—Address flushing—worked example

Figure 17-11 illustrates this optimization. Disabling the Root Port (Port 1) of Bridge 888 temporarily partitions the active topology, all addresses learned on that Port reside somewhere on the main tree (and not on the subtree). When the Alternate Port (Port 2) becomes the new Root Port and transitions to Forwarding, it provides the new path from the subtree to the Root, and hence the path to those addresses.

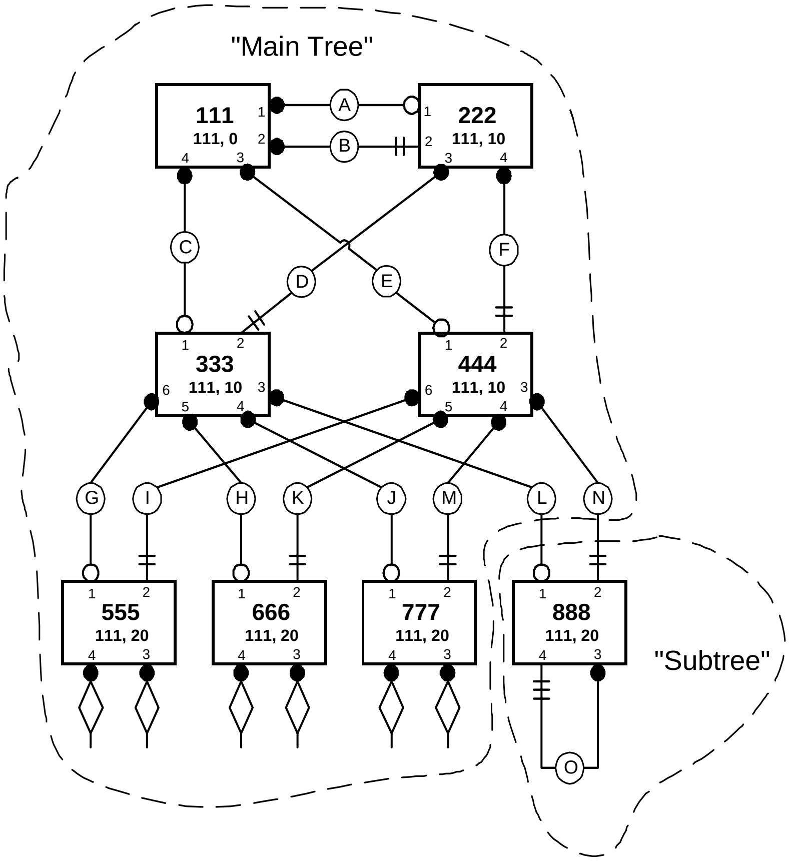


Figure 17-11—Root Port transition—tree partitioning

17.12 RSTP and point-to-point links

The rapid transition of a Designated Port to Forwarding depends on the Port being directly connected to at most one other Bridge [it is an Edge Port (17.3, 17.19.17), or is attached to a point-to-point LAN, rather than a shared medium]. The adminPointToPointMAC and operPointToPointMAC parameters (6.4.3) provide management and signalling of the point-to-point status to RSTP state machines.

A newly selected Root Port can transition to Forwarding rapidly, even if attached to shared media.

17.13 RSTP performance parameters

These parameters are not modified by the operation of RSTP, but are treated as constants by the RSTP state machines and the associated variables and procedures (17.20, 17.18, 17.19). They may be modified by management.

The Spanning Tree Protocol Entity shall be reinitialized, as specified by the assertion of BEGIN (17.18.1) in the state machine specification, if the following parameters are modified:

a) Force Protocol Version (17.13.4)

The spanning tree priority vectors and Port Role assignments for a Bridge shall be recomputed, as specified by the operation of the Port Role Selection state machine (17.28) by clearing selected (17.19.36) and setting reselect (17.19.34) for any Port or Ports for which the following parameters are modified:

b) Bridge Identifier Priority (17.13.7)
c) Port Identifier Priority (17.13.10)
d) Port Path Cost (17.13.11)

If the Transmit Hold Count is modified the value of txCount (17.19.44) for all Ports shall be set to zero.

The RSTP specification permits changes in other performance parameters without exceptional actions.

17.13.1 Admin Edge Port

The AdminEdgePort parameter for the Port (14.8.2).

17.13.2 Ageing Time

The Ageing Time parameter for the Bridge (7.9.2, Table 7-5).

17.13.3 AutoEdge

The AutoEdgePort parameter for the Port (14.8.2).

17.13.4 Force Protocol Version

The Force Protocol Version parameter for the Bridge (17.4, 14.8.1). This can take the value 0 (“STP Compatibility” mode) or 2 (the default, normal operation).

17.13.5 Bridge Forward Delay

The delay used by STP Bridges (17.4) to transition Root and Designated Ports to Forwarding (Table 17-1).

17.13.6 Bridge Hello Time

The interval between periodic transmissions of Configuration Messages by Designated Ports (Table 17-1).

17.13.7 Bridge Identifier Priority

The manageable component of the Bridge Identifier, also known as the Bridge Priority (14.8.1, Table 17-2).

17.13.8 Bridge Max Age

The maximum age of the information transmitted by the Bridge when it is the Root Bridge (Table 17-1).

17.13.9 Migrate Time

The initial value of the mdelayWhile and edgeDelayWhile timers (17.17.4, 17.17.1), fixed for all RSTP implementations conforming to this specification (Table 17-1).

17.13.10 Port Identifier Priority

The manageable component of the Port Identifier, also known as the Port Priority (14.8.2, Table 17-2).

17.13.11 PortPathCost

The Port’s contribution, when it is the Root Port, to the Root Path Cost (17.3.1, 17.5, 17.6) for the Bridge.

17.13.12 Transmit Hold Count

The Transmit Hold Count (Table 17-1) used by the Port Transmit state machine to limit transmission rate.

17.14 Performance parameter management

Table 17-1 specifies default values and ranges for timer and transmission rate limiting performance parameters. Defaults are specified to avoid the need to set values prior to operation in most cases, and have been chosen for their wide applicability to maximize ease of operation. Ranges are specified to ensure that the protocol operates correctly, and provide guidance to implementors.

Table 17-1—RSTP Timer and Transmit Hold Count parameter values

Parameter	Recommended or Default value	Permitted Range	Compatibility Range
Migrate Time (17.13.9)	3.0	—[a]	—[a]
Bridge Hello Time (17.13.6)	2.0	—[a]	1.0–2.0
Bridge Max Age (17.13.8)	20.0	6.0–40.0	6.0–40.0
Bridge Forward Delay (17.13.5)	15.0	4.0–30.0	4.0–30.0
Transmit Hold Count (17.13.12)	6	1–10	1–10

All times are in seconds. —[1] Not applicable, value is fixed.

The recommended values for Bridge Hello Time, Bridge Max Age, and Bridge Forward Delay are the same as those specified for STP in previous revisions of this standard. They have been retained to facilitate interoperability and integration of RSTP Bridges into networks that include legacy STP Bridges, to support migration of existing networks to RSTP.

NOTE—Changes to Bridge Forward Delay do not affect reconfiguration times, unless the network includes Bridges that do not conform to this revision of this standard. Changes to Bridge Max Age can have an effect, as it is possible for old information to persist in loops in the physical topology for a number of “hops” equal to the value of Max Age in seconds, and thus exhaust the Transmit Hold Count in small loops.

Bridge Max Age, Bridge Forward Delay, and Transmit Hold Count may be set by management, if this capability is provided the Bridge shall have the capability to use the full range of values in the parameter ranges specified in the Permitted Range column of Table 17-1, with a timer resolution of *r* seconds, where $0 < r <= 1$. To support interoperability with legacy Bridges, a Bridge shall enforce the following relationships:

$$2 \times (\mathit{Bridge\_Forward\_Delay} - 1.0\ \mathit{seconds}) >= \mathit{Bridge\_Max\_Age}$$

$$\mathit{Bridge\_Max\_Age} >= 2 \times (\mathit{Bridge\_Hello\_Time} + 1.0\ \mathit{seconds})$$

The Bridge Identifier Priority, and the Port Path Cost and Port Identifier Priority for each Port, may be set to manage the active topology of the network.Table 17-2 specifies default values and ranges for Bridge and Port Identifier Priorities. If these parameters can be updated by management, the Bridge shall have the capability to use the full range of values with the granularity specified.

Table 17-2—Bridge and Port Identifier Priority values

Parameter	Recommended or default value	Range
Bridge Priority	32 768	0–61 440 in steps of 4096
Port Priority	128	0–240 in steps of 16

NOTE 1—The stated ranges and granularities for Bridge Priority and Port Priority differ from the equivalent text and table in IEEE Std 802.1D, 1998 Edition and earlier versions of this standard as explained in 9.2.5 and 9.2.7. Expressing these values in steps of 4096 and 16 (rather than, for example, as a 4-bit value with a range of 0 to 15) allows consistent management across old and new implementations of this standard; the steps chosen ensure that bits that have been re-assigned are not modified, but priority values can be directly compared with those based on previous versions of the standard.

Table 17-3 recommends defaults and ranges for Port Path Cost (17.13.11) values, chosen according to the speed of the attached LAN. If Port Path Cost can be set by management, the Bridge shall be able to use the full range of values in the parameter ranges specified, with a granularity of 1.

Table 17-3—Port Path Cost values

Link Speed	Recommended value	Recommended range	Range
<=100 Kb/s	200 000 000*	20 000 000–200 000 000	1–200 000 000
1 Mb/s	20 000 000[a]	2 000 000–200 000 000	1–200 000 000
10 Mb/s	2 000 000[a]	200 000–20 000 000	1–200 000 000
100 Mb/s	200 000[a]	20 000–2 000 000	1–200 000 000
1 Gb/s	20 000	2 000–200 000	1–200 000 000
10 Gb/s	2 000	200–20 000	1–200 000 000
100 Gb/s	200	20–2 000	1–200 000 000
1 Tb/s	20	2–200	1–200 000 000
10 Tb/s	2	1–20	1–200 000 000

*Bridges conformant to IEEE Std 802.1D, 1998 Edition, i.e., that support only 16-bit values for Path Cost, should use 65 535 as the Path Cost for these link speeds when used in conjunction with Bridges that support 32-bit Path Cost values.

Where intermediate link speeds are created as a result of the aggregation of two or more links of the same speed (see IEEE Std 802.3-2002), it can be appropriate to modify the recommended values shown in Table 17-3 to reflect the change in link speed. However, as the primary purpose of the Path Cost is to establish the active topology of the network, it can be inappropriate for the Path Cost to track the effective speed of such links too closely, as the resultant active topology could differ from that intended by the network administrator. For example, if the network administrator had chosen an active topology that makes use of aggregated links for resilience (rather than for increased data rate), it would be inappropriate to cause a Spanning Tree topology change as a result of one of the physical links in an aggregation failing. Similarly, with links that can autonegotiate their data rate, reflecting such changes of data rate in changes to Path Cost is not necessarily appropriate, depending upon the intent of the network administrator. As a default, dynamic changes of data rate shall not automatically cause changes in Port Path Cost.

NOTE 2—The values shown in Table 17-3 apply to both full duplex and half duplex operation. The intent of the recommended values and ranges shown is to minimize the number of Bridges in which path costs need to be managed to exert control over the topology of the Bridged Local Area Network.

NOTE 3—BPDUs are capable of carrying 32 bits of Root Path Cost information; however, IEEE Std. 802.1D, 1998 Edition and earlier revisions of this standard limited the range of the Port Path Cost parameter to a 16-bit unsigned integer value. The recommended values shown in Table 17-3 make use of the full 32-bit range available in BPDUs in order to extend the range of link speeds supported by the protocol. The recommended values for any intermediate link speed can be calculated as 20 000 000 000/(Link Speed in Kb/s). Limiting the range of the Path Cost parameter to 1–200 000 000 ensures that the accumulated Path Cost cannot exceed 32 bits over a concatenation of 20 hops. In LANs where Bridges that use the recommended values defined in IEEE Std 802.1D, 1998 Edition and Bridges that use the recommended values shown in this table are required to interwork, either the older Bridges will need to be re-configured in order to make use of the Path Cost values shown, or the new Bridges will need to be re-configured to make use of Path Cost values compatible with the values used by the older Bridges. The range of Path Costs that can be configured in an older Bridge is insufficient to accommodate the range of data rates available.

17.15 Rapid Spanning Tree state machines

The behavior of an RSTP implementation in a Bridge is specified by a number of cooperating state machines. Figure 17-12 is not itself a state machine, but illustrates the machines, their interrelationships, the principal variables used to communicate between them, their local variables, and performance parameters.

A single Port Role Selection state machine shall be implemented per Bridge, and one instance of each of the other state machines shall be implemented per Bridge Port.

17.16 Notational conventions used in state diagrams

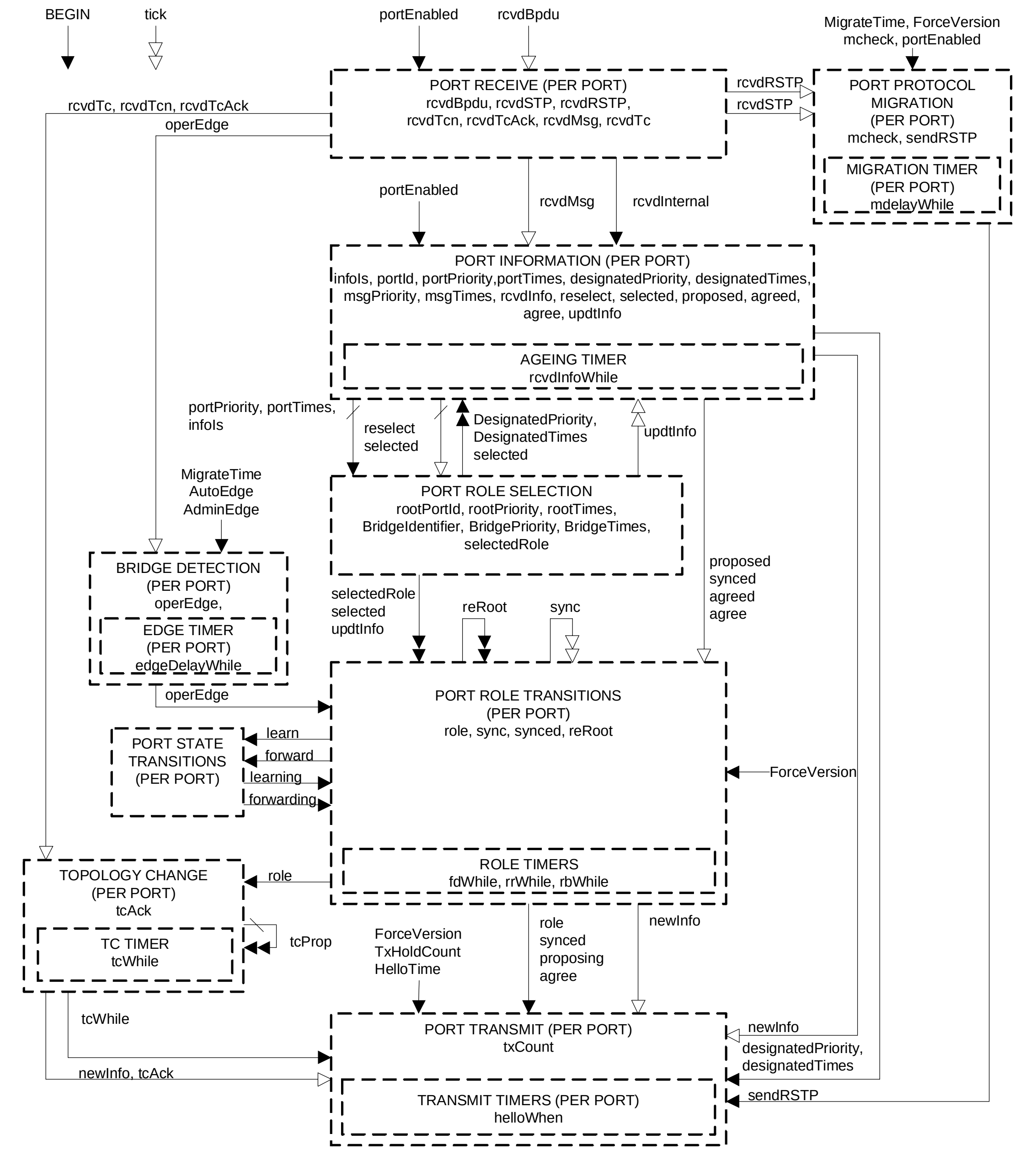
State diagrams are used to represent the operation of the protocol by a number of cooperating state machines each comprising a group of connected, mutually exclusive states. Only one state of each machine can be active at any given time.

Each state is represented in the state diagram as a rectangular box, divided into two parts by a horizontal line. The upper part contains the state identifier, written in upper case letters. The lower part contains any procedures that are executed on entry to the state.

All permissible transitions between states are represented by arrows, the arrowhead denoting the direction of the possible transition. Labels attached to arrows denote the condition(s) that must be met in order for the transition to take place. All conditions are expressions that evaluate to TRUE or FALSE; if a condition evaluates to TRUE, then the condition is met. The label UCT denotes an unconditional transition (i.e., UCT always evaluates to TRUE). A transition that is global in nature (i.e., a transition that occurs from any of the possible states if the condition attached to the arrow is met) is denoted by an open arrow, i.e., no specific state is identified as the origin of the transition. When the condition associated with a global transition is met, it supersedes all other exit conditions including UCT. The special global condition BEGIN supersedes all other global conditions, and once asserted remains asserted until all state blocks have executed to the point that variable assignments and other consequences of their execution remain unchanged.

On entry to a state, the procedures defined for the state (if any) are executed exactly once, in the order that they appear on the page. Each action is deemed to be atomic, i.e., execution of a procedure completes before the next sequential procedure starts to execute. No procedures execute outside of a state block. The procedures in only one state block execute at a time, even if the conditions for execution of state blocks in different state machines are satisfied. All procedures in an executing state block complete execution before the transition to and execution of any other state block occurs, i.e., the execution of any state block appears to be atomic with respect to the execution of any other state block and the transition condition to that state from the previous state is TRUE when execution commences. The order of execution of state blocks in different state machines is undefined except as constrained by their transition conditions. A variable that is set to a particular value in a state block retains this value until a subsequent state block executes a procedure that modifies the value.

On completion of all of the procedures within a state, all exit conditions for the state (including all conditions associated with global transitions) are evaluated continuously until one of the conditions is met. The label ELSE denotes a transition that occurs if none of the other conditions for transitions from the state are met (i.e., ELSE evaluates to TRUE if all other possible exit conditions from the state evaluate to FALSE). Where two or more exit conditions with the same level of precedence become TRUE simultaneously, the choice as to which exit condition causes the state transition to take place is arbitrary.



NOTATION:

Variables are shown both within the machine where they are principally used and between machines where they are used to communicate information. In the latter case a variety of arrow styles, running from one machine to another, show how each is typically used:

——▶ Not changed by the target machine. Where the machines are both per Port, this variable communicates between instances for the same port.

——▷ Set (or cleared) by the originating machine, cleared (or set) by the target machine. Where the machines are both per Port, this communicates between instances for the same port.

——▶▶ / ——▷▷ As above except that the originating per port machine instance communicates with multiple port machine instances (by setting or clearing variables owned by those Ports).

—/—▶ / —/—▷ / —/—▶▶ As above except that multiple per Port instances communicate with (an)other instance(s) (by setting or clearing variables owned by the originating Ports).

ABBREVIATIONS:

PIM:	Port Information Machine
PRS:	Port Role Selection Machine
PRT:	Port Role Transitions Machine
PRX:	Port Receive Machine
PST:	Port State Transitions Machine
TCM:	Topology Change Machine
PPM:	Port Protocol Migration Machine
PTX:	Port Transmit Machine
PTI:	Port TImers Machine
BDM:	Bridge Detection Machine

NOTE: All timer variables are decremented by one machine -- PTI (not shown).

Figure 17-12—RSTP state machines—overview and interrelationships

Where it is necessary to split a state machine description across more than one diagram, a transition between two states that appear on different diagrams is represented by an exit arrow drawn with dashed lines, plus a reference to the diagram that contains the destination state. Similarly, dashed arrows and a dashed state box are used on the destination diagram to show the transition to the destination state. In a state machine that has been split in this way, any global transitions that can cause entry to states defined in one of the diagrams are deemed to be potential exit conditions for all of the states of the state machine, regardless of which diagram the state boxes appear in.

Should a conflict exist between the interpretation of a state diagram and either the corresponding global transition tables or the textual description associated with the state machine, the state diagram takes precedence. The interpretation of the special symbols and operators used in the state diagrams is as defined in Table 17-4; these symbols and operators are derived from the notation of the “C++” programming language, ISO/IEC 14882. If a Boolean variable is described in this clause as being set it has or is assigned the value TRUE, if reset or clear the value FALSE.

Table 17-4—State machine symbols

Symbol	Interpretation
( )	Used to force the precedence of operators in Boolean expressions and to delimit the argument(s) of actions within state boxes.
;	Used as a terminating delimiter for actions within state boxes. Where a state box contains multiple actions, the order of execution follows the normal English language conventions for reading text.
=	Assignment action. The value of the expression to the right of the operator is assigned to the variable to the left of the operator. Where this operator is used to define multiple assignments, e.g., a = b = X the action causes the value of the expression following the right-most assignment operator to be assigned to all of the variables that appear to the left of the right-most assignment operator.
!	Logical NOT operator.
&&	Logical AND operator.
||	Logical OR operator.
if...then...	Conditional action. If the Boolean expression following the if evaluates to TRUE, then the action following the then is executed.
!=	Inequality. Evaluates to TRUE if the expression to the left of the operator is not equal in value to the expression to the right.
==	Equality. Evaluates to TRUE if the expression to the left of the operator is equal in value to the expression to the right.
*	Arithmetic multiplication operator.
-	Arithmetic subtraction operator.

17.17 State machine timers

The timer variables declared in this subclause are part of the specification of the operation of the RSTP. The accompanying descriptions of their meaning and use are provided to aid in the comprehension of the protocol only, and are not part of the specification. An RSTP implementation shall implement a single instance of each timer variable per port.

Each timer variable represents an integral number of seconds before timer expiry.

17.17.1 edgeDelayWhile

The Edge Delay timer. The time remaining, in the absence of a received BPDU, before this port is identified as an operEdgePort.

17.17.2 fdWhile

The Forward Delay timer. Used to delay Port State transitions until other Bridges have received spanning tree information.

17.17.3 helloWhen

The Hello timer. Used to ensure that at least one BPDU is transmitted by a Designated Port in each HelloTime period.

17.17.4 mdelayWhile

The Migration Delay timer. Used by the Port Protocol Migration state machine to allow time for another RSTP Bridge on the same LAN to synchronize its migration state with this Port before the receipt of a BPDU can cause this Port to change the BPDU types it transmits. Initialized to MigrateTime (17.13.9).

17.17.5 rbWhile

The Recent Backup timer. Maintained at its initial value, twice HelloTime, while the Port is a Backup Port.

17.17.6 rcvdInfoWhile

The Received Info timer. The time remaining before the spanning tree information received by this Port [portPriority (17.19.21) and portTimes (17.19.22)] is aged out if not refreshed by the receipt of a further Configuration Message.

17.17.7 rrWhile

The Recent Root timer.

17.17.8 tcWhile

The Topology Change timer. TCN Messages are sent while this timer is running.

17.18 Per-Bridge variables

The variables declared in this subclause are part of the specification of the operation of the RSTP. The accompanying descriptions of their use are provided to aid in the comprehension of the protocol only, and are not part of the specification.

17.18.1 BEGIN

A Boolean controlled by the system initialization (17.16). If TRUE causes all state machines, including per Port state machines, to continuously execute their initial state.

17.18.2 BridgeIdentifier

The unique Bridge Identifier assigned to this Bridge, comprising two components—the Bridge Identifier Priority, which may be modified by management (see 9.2.5 and 14.8.1.2) and is the more significant when Bridge Identifiers are compared, and a component derived from the Bridge Address (7.12.5), which guarantees uniqueness of the Bridge Identifiers of different Bridges.

17.18.3 BridgePriority

The bridge priority vector, as defined in 17.6. The first (RootBridgeID) and third (DesignatedBridgeID) components are both equal to the value of the Bridge Identifier (17.18.2). The other components are zero.

17.18.4 BridgeTimes

BridgeTimes comprises four components—the current values of Bridge Forward Delay, Bridge Hello Time, and Bridge Max Age (17.13, Table 17-1), and a Message Age of zero.

17.18.5 rootPortId

The Port Identifier of the Root Port—this is the fifth component of the root priority vector, as defined in 17.6.

17.18.6 rootPriority

The first four components of the Bridge’s root priority vector, as defined in 17.6.

17.18.7 rootTimes

The rootTimes variable comprises the Bridge’s operational timer parameter values (Message Age, Max Age, Forward Delay, and Hello Time), derived from the values stored in portTimes (17.19.22) for the Root Port or from BridgeTimes (17.18.4).

17.19 Per-Port variables

The variables declared in this subclause are part of the specification of the operation of the RSTP. The accompanying descriptions are not part of the specification.

17.19.1 ageingTime

Filtering database entries for this Port are aged out after ageingTime has elapsed since they were first created or refreshed by the Learning Process. The value of this parameter is normally Ageing Time (7.9.2, Table 7-5), and is changed to FwdDelay (17.20.6) for a period of FwdDelay after fdbFlush (17.19.7) is set by the topology change state machine if stpVersion (17.19.7) is TRUE.

17.19.2 agree

A boolean. See 17.10.

17.19.3 agreed

A boolean. See 17.10.

17.19.4 designatedPriority

The first four components of the Port's designated priority vector value, as defined in 17.6. The fifth component of the designated priority vector value is portId (17.19.19).

17.19.5 designatedTimes

The designatedTimes variable comprises the set of timer parameter values (Message Age, Max Age, Forward Delay, and Hello Time) that used to update Port Times when updtInfo is set. Updated by the updtRolesTree() procedure (17.21.25).

17.19.6 disputed

A boolean. See 17.21.10.

17.19.7 fdbFlush

A boolean. Set by the topology change state machine to instruct the filtering database to remove all entries for this Port, immediately if rstpVersion (17.20.11) is TRUE, or by rapid ageing (17.19.1) if stpVersion (17.20.12) is TRUE. Reset by the filtering database once the entries are removed if rstpVersion is TRUE, and immediately if stpVersion is TRUE.

17.19.8 forward

A boolean. See 17.30.

17.19.9 forwarding

A boolean. See 17.30.

17.19.10 infoIs

A variable that takes the values Mine, Aged, Received, or Disabled, to indicate the origin/state of the Port's Spanning Tree information (portInfo) held for the Port, as follows:

a) If infoIs is **Received**, the port has received current (not aged out) information from the Designated Bridge for the attached LAN (a point-to-point bridge link being a special case of a LAN).
b) If infoIs is **Mine**, information for the port has been derived from the Root Port for the Bridge (with the addition of root port cost information). This includes the possibility that the Root Port is "Port 0," i.e., the bridge is the Root Bridge for the Bridged Local Area Network.
c) If infoIs is **Aged**, information from the Root Bridge has been aged out. Just as for "reselect" (see 17.19.34), the state machine does not formally allow the "Aged" state to persist. However, if there is a delay in recomputing the new root port, correct processing of a received BPDU is specified.
d) Finally if the port is disabled, infoIs is **Disabled**.

17.19.11 learn

A boolean. See 17.30.

17.19.12 learning

A boolean. See 17.30.

17.19.13 mcheck

A boolean. May be set by management to force the Port Protocol Migration state machine to transmit RST BPDUs for a MigrateTime (17.13.9) period, to test whether all STP Bridges (17.4) on the attached LAN have been removed and the Port can continue to transmit RSTP BPDUs. Setting mcheck has no effect if stpVersion (17.20.12) is TRUE, i.e., the Bridge is operating in “STP Compatibility” mode.

17.19.14 msgPriority

The first four components of the message priority vector conveyed in a received BPDU, as defined in 17.6.

17.19.15 msgTimes

The msgTimes variable comprises the timer parameter values (Message Age, Max Age, Forward Delay, and Hello Time) conveyed in a received BPDU.

17.19.16 newInfo

A boolean. Set if a BPDU is to be transmitted. Reset by the Port Transmit state machine.

17.19.17 operEdge

A boolean. The value of the operEdgePort parameter, as determined by the operation of the Bridge Detection state machine (17.25).

17.19.18 portEnabled

A boolean. Set if the Bridge’s MAC Relay Entity and Spanning Tree Protocol Entity can use the MAC Service provided by the Port’s MAC entity to transmit and receive frames to and from the attached LAN, i.e., portEnabled is TRUE if and only if:

a) MAC_Operational (6.4.2) is TRUE; and
b) Administrative Bridge Port State (14.8.2.2) for the Port is Enabled; and
c) AuthControlledPortStatus is Authorized [if the port is a network access port (IEEE Std 802.1X)].

17.19.19 portId

The Port Identifier. This variable forms the fifth component of the port priority and designated priority vectors defined in 17.6.

17.19.20 PortPathCost

The Port’s contribution, when it is the Root Port, to the Root Path Cost (17.3.1, 17.5, 17.6) for the Bridge.

17.19.21 portPriority

The first four components of the Port’s port priority vector value, as defined in 17.6.

17.19.22 portTimes

The portTimes variable comprises the Port’s timer parameter values (Message Age, Max Age, Forward Delay, and Hello Time). These timer values are used in BPDUs transmitted from the Port.

17.19.23 proposed

A boolean. See 17.10.

17.19.24 proposing

A boolean. See 17.10.

17.19.25 rcvdBPDU

A boolean. Set by system dependent processes, this variable notifies the Port Receive state machine (17.23) when a valid (9.3.4) Configuration, TCN, or RST BPDU (9.3.1, 9.3.2, 9.3.3) is received on the Port. Reset by the Port Receive state machine.

17.19.26 rcvdInfo

Set to the result of the rcvInfo() procedure (17.21.8).

17.19.27 rcvdMsg

A boolean. See 17.23.

17.19.28 rcvdRSTP

A boolean. See 17.23.

17.19.29 rcvdSTP

A boolean. See 17.23.

17.19.30 rcvdTc

A boolean. See 17.21.17 and 17.31.

17.19.31 rcvdTcAck

A boolean. See 17.21.17 and 17.31.

17.19.32 rcvdTcn

A boolean. See 17.21.17 and 17.31.

17.19.33 reRoot

A boolean. See 17.29.2.

17.19.34 reselect

A boolean. See 17.28.

17.19.35 role

The assigned Port Role (17.7).

17.19.36 selected

A boolean. See 17.28, 17.21.16.

17.19.37 selectedRole

The newly computed role for the Port (17.7, 17.28, 17.21.25, 17.19.35).

17.19.38 sendRSTP

A boolean. See 17.24, 17.26.

17.19.39 sync

A boolean. See 17.10.

17.19.40 synced

A boolean. See 17.10.

17.19.41 tcAck

A boolean. Set if a Configuration Message with a topology change acknowledge flag set is to be transmitted.

17.19.42 tcProp

A boolean. Set by the Topology Change state machine of any other Port, to indicate that a topology change should be propagated through this Port.

17.19.43 tick

A boolean. See 17.22.

17.19.44 txCount

A counter. Incremented by the Port Transmission (17.26) state machine on every BPDU transmission, and decremented used by the Port Timers state machine (17.22) once a second. Transmissions are delayed if txCount reaches TxHoldCount (17.13.12).

17.19.45 updtInfo

A boolean. Set by the Port Role Selection state machine (17.28, 17.21.25) to tell the Port Information state machine that it should copy designatedPriority to portPriority and designatedTimes to portTimes.

17.20 State machine conditions and parameters

The following variable evaluations are defined for notational convenience in the state machines.

17.20.1 AdminEdge

The AdminEdgePort parameter for the Port (14.8.2).

17.20.2 AutoEdge

The AutoEdgePort parameter for the Port (14.8.2).

17.20.3 allSynced

TRUE if and only if, for all Ports for the given Tree, selected is true and the port’s role is the same as its selectedRole and either

a) synced is true; or
b) The port is the Root Port.

17.20.4 EdgeDelay

Returns the value of MigrateTime if operPointToPointMAC is TRUE, and the value of MaxAge otherwise.

17.20.5 forwardDelay

Returns the value of HelloTime if sendRSTP is TRUE, and the value of FwdDelay otherwise.

17.20.6 FwdDelay

The Forward Delay component of designatedTimes (17.19.5).

17.20.7 HelloTime

The Hello Time component of designatedTimes (17.19.5).

17.20.8 MaxAge

The Max Age component of designatedTimes (17.19.5).

17.20.9 MigrateTime

The Migrate Time parameter (17.13.9).

17.20.10 reRooted

TRUE if the rrWhile timer is clear (zero) for all Ports for the given Tree other than the given Port.

17.20.11 rstpVersion

TRUE if Force Protocol Version (17.13.4) is greater than or equal to 2.

17.20.12 stpVersion

TRUE if Force Protocol Version (17.13.4) is less than 2.

17.20.13 TxHoldCount

The Transmit Hold Count (17.13.12, Table 17-1).

17.21 State machine procedures

The following naming convention is used for the names of procedures that modify multiple variables (either multiple variables of a single Port or variables of multiple Ports):

a) ***set***: The procedure sets the value of the variables to TRUE.
b) ***clear:*** The procedure clears (resets) the value of the variables to FALSE.
c) ***updt:*** The procedure updates the variables in some other way.

The suffix “Tree” is used for procedures that can modify a variable in all Ports of the Bridge. For example, ***setSyncTree()*** is the name of a procedure that sets a variable TRUE for all Bridge Ports.

Where procedures are used to determine the value of a single variable, the procedure’s returned value is explicitly assigned to the variable in the state machine concerned.

17.21.1 betterorsameinfo(newInfoIs)

Returns TRUE if either

a) The procedure’s parameter newInfoIs is Received, and infoIs is Received and the msgPriority vector is better than or the same as (17.6) the portPriority vector; or,
b) The procedure’s parameter newInfoIs is Mine, and infoIs is Mine and the designatedPriority vector is better than or the same as (17.6) the portPriority vector.

Returns False otherwise.

17.21.2 clearReselectTree()

Clears reselect for all Ports of the Bridge.

17.21.3 disableForwarding()

An implementation dependent procedure that causes the Forwarding Process (7.7) to stop forwarding frames through the Port. The procedure does not complete until forwarding has stopped.

17.21.4 disableLearning()

An implementation dependent procedure that causes the Learning Process (7.8) to stop learning from the source address of frames received on the Port. The procedure does not complete until learning has stopped.

17.21.5 enableForwarding()

An implementation dependent procedure that causes the Forwarding Process (7.7) to start forwarding frames through the Port. The procedure does not complete until forwarding has been enabled.

17.21.6 enableLearning()

An implementation dependent procedure that causes the Learning Process (7.8) to start learning from frames received on the Port. The procedure does not complete until learning has been enabled.

17.21.7 newTcWhile()

If the value of tcWhile is zero and sendRstp is true, this procedure sets the value of tcWhile to HelloTime plus one second and sets newInfo true.

If the value of tcWhile is zero and sendRstp is false, this procedure sets the value of tcWhile to the sum of the Max Age and Forward Delay components of rootTimes and does not change the value of newInfo.

Otherwise the procedure takes no action.

17.21.8 rcvInfo()

Decodes the message priority and timer values from the received BPDU storing them in the msgPriority and msgTimes variables.

Returns SuperiorDesignatedInfo if:

a) The received message conveys a Designated Port Role, and
   1) The message priority is superior (17.6) to the Port’s port priority vector, or
   2) The message priority vector is the same as the Port’s port priority vector, and any of the received timer parameter values (msgTimes—17.19.15) differ from those already held for the Port (portTimes—17.19.22).

Returns RepeatedDesignatedInfo if:

b) The received message conveys Designated Port Role, and a message priority vector and timer parameters that are the same as the Port’s port priority vector or timer values.

Returns InferiorDesignatedInfo if:

c) The received message conveys a Designated Port Role, and a message priority vector that is worse than the Port’s port priority vector.

Returns InferiorRootAlternateInfo if:

d) The received message conveys a Root Port, Alternate Port, or Backup Port Role and a message priority that is the same as or worse than the port priority vector.

Otherwise, returns OtherInfo.

NOTE—A Configuration BPDU explicitly conveys a Designated Port Role.

17.21.9 recordAgreement()

If rstpVersion is TRUE, operPointToPointMAC (6.4.3) is TRUE, and the received Configuration Message has the Agreement flag set, the agreed flag is set and the proposing flag is cleared. Otherwise, the agreed flag is cleared.

17.21.10 recordDispute()

If an RST BPDU with the learning flag set has been received:

a) The agreed flag is set; and
b) The proposing flag is cleared.

17.21.11 recordProposal()

If the received Configuration Message conveys a Designated Port Role, and has the Proposal flag is set, the proposed flag is set. Otherwise, the proposed flag is not changed.

17.21.12 recordPriority()

Sets the components of the portPriority variable to the values of the corresponding msgPriority components.

17.21.13 recordTimes()

Sets portTimes' Message Age, MaxAge, and Forward Delay to the received values held in the messageTimes parameter. and portTimes' Hello time to messageTimes' HelloTime if that is greater than the minimum specified in the Compatibility Range column of Table 17-1, and to that minimum otherwise.

17.21.14 setSyncTree()

Sets sync TRUE for all Ports of the Bridge.

17.21.15 setReRootTree()

Sets reRoot TRUE for all Ports of the Bridge.

17.21.16 setSelectedTree()

Sets the selected variable TRUE for all Ports of the Bridge if reselect is FALSE for all Ports. If reselect is TRUE for any Port, this procedure takes no action.

17.21.17 setTcFlags()

Sets rcvdTc and/or rcvdTcAck if the Topology Change and/or Topology Change Acknowledgment flags, respectively, are set in a ConfigBPDU or RST BPDU. Sets rcvdTcn TRUE if the BPDU is a TCN BPDU.

17.21.18 setTcPropTree()

Sets tcprop for all Ports except the Port that called the procedure.

17.21.19 txConfig()

Transmits a Configuration BPDU. The components of the message priority vector (17.6) conveyed in the BPDU are set to the value of designatedPriority (17.19.21) for this Port. The topology change flag is set if (tcWhile ! = 0) for the Port. The topology change acknowledgement flag is set to the value of TcAck for the Port. The value of the Message Age, Max Age, Fwd Delay, and Hello Time parameters conveyed in the BPDU are set to the values held in designatedTimes (17.19.22) for the Port.

17.21.20 txRstp()

Transmits an RST BPDU. The components of the message priority vector (17.6) conveyed in the BPDU are set to the value of designatedPriority (17.19.4) for this Port. The Port Role in the BPDU (9.3.3) is set to the current value of the role variable for the transmitting port (17.19.35). The Agreement and Proposal flags in the BPDU are set to the values of the agree (17.19.2) and proposing (17.19.24) variables for the transmitting Port, respectively. The topology change flag is set if (tcWhile ! = 0) for the Port. The topology change acknowledge flag in the BPDU is never used and is set to zero. The Learning and Forwarding flags in the BPDU are set to the values of the learning (17.19.12) and forwarding (17.19.9) variables for the transmitting Port, respectively. The value of the Message Age, Max Age, Fwd Delay, and Hello Time parameters conveyed in the BPDU are set to the values held in designatedTimes (17.19.5) for the Port.

17.21.21 txTcn()

Transmits a TCN BPDU.

17.21.22 updtBPDUVersion()

Sets rcvdSTP TRUE if the BPDU received is a version 0 or version 1 TCN or a Config BPDU. Sets rcvdRSTP TRUE if the received BPDU is an RST BPDU.

17.21.23 updtRcvdInfoWhile()

Updates rcvdInfoWhile (17.17.6). The value assigned to rcvdInfoWhile is the three times the Hello Time, if Message Age, incremented by 1 second and rounded to the nearest whole second, does not exceed Max Age, and is zero otherwise. The values of Message Age, Max Age, and Hello Time used in this calculation are taken from portTimes (17.19.22).

17.21.24 updtRoleDisabledTree()

Sets selectedRole to DisabledPort for all Ports of the Bridge.

17.21.25 updtRolesTree()

This procedure calculates the following spanning tree priority vectors (17.5, 17.6) and timer values:

a) The *root path priority vector* for each Port that has a *port priority vector* (portPriority plus portId; 17.19.19, 17.19.21), recorded from a received message and not aged out (infoIs == Received)

b) The Bridge’s *root priority vector* (rootPriority plus rootPortId; 17.18.6, 17.18.5), chosen as the best of the set of priority vectors comprising the Bridge’s own *bridge priority vector* (BridgePriority; 17.18.3) and all the calculated root path priority vectors whose DesignatedBridgeID Bridge Address component is not equal to that component of the Bridge’s own bridge priority vector (see 17.6)

c) The Bridge’s rootTimes (17.18.7) parameter, set equal to:

    1) BridgeTimes (17.18.4), if the chosen root priority vector is the bridge priority vector, otherwise

    2) portTimes (17.19.22) for the port associated with the selected root priority vector, with the Message Age component incremented by 1 second and rounded to the nearest whole second.

d) The first four components of the *designated priority vector* (designatedPriority, 17.19.4) for each port.

e) The designatedTimes (17.19.5) for each Port, set equal to the value of rootTimes, except for the Hello Time component, which is set equal to BridgeTimes’ Hello Time

The port role for each Port is assigned, and its port priority vector and Spanning Tree timer information are updated as follows:

f) If the Port is Disabled (infoIs = Disabled), selectedRole is set to DisabledPort. Otherwise:

g) If the port priority vector information was aged (infoIs = Aged), updtInfo is set and selectedRole is set to DesignatedPort.

h) If the port priority vector was derived from another port on the Bridge or from the Bridge itself as the Root Bridge (infoIs = Mine), selectedRole is set to DesignatedPort. Additionally, updtInfo is set if the port priority vector differs from the designated priority vector or the Port’s associated timer parameters differ from those for the Root Port.

i) If the port priority vector was received in a Configuration Message and is not aged (infoIs == Received), and the root priority vector is now derived from it, selectedRole is set to RootPort and updtInfo is reset.

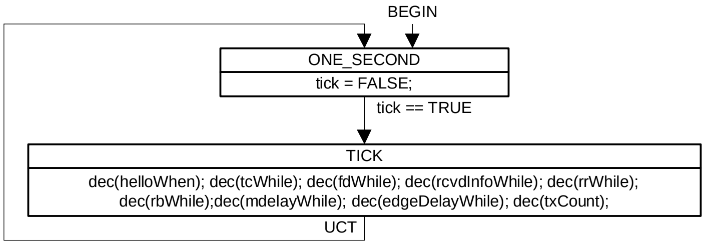
j) If the port priority vector was received in a Configuration Message and is not aged (infoIs = Received), the root priority vector is not now derived from it, the designated priority vector is not higher than the port priority vector, and the designated bridge and designated port components of the port priority vector do not reflect another port on this bridge, selectedRole is set to AlternatePort and updtInfo is reset.

k) If the port priority vector was received in a Configuration Message and is not aged (infoIs = Received), the root priority vector is not now derived from it, the designated priority vector is not higher than the port priority vector, and the designated bridge and designated port components of the port priority vector reflect another port on this bridge, selectedRole is set to BackupPort and updtInfo is reset.

l) If the port priority vector was received in a Configuration Message and is not aged (infoIs == Received), the root priority vector is not now derived from it, the designated priority vector is higher than the port priority vector, selectedRole is set to DesignatedPort and updtInfo is set.

17.22 Port Timers state machine

The Port Timers state machine shall implement the function specified by the state diagram in Figure 17-13, the definitions in 17.16, and the variable declarations and procedures specified in 17.17 through 17.21. It uses the tick (17.19.43) signal, set by an implementation specific system clock function at one second intervals, to decrement the timer variables for the Port. The state machine that uses a given timer variable is responsible for setting the variable to its initial value.


BEGIN
ONE_SECOND
tick = FALSE;
tick == TRUE
TICK
dec(helloWhen); dec(tcWhile); dec(fdWhile); dec(rcvdInfoWhile); dec(rrWhile);
dec(rbWhile);dec(mdelayWhile); dec(edgeDelayWhile); dec(txCount);
UCT

The following abbreviation is used in this state diagram:
dec(x)
{ if (x !=0) x= x-1;
}

If there is more than one instance of x, e.g. per port, all instances are separately decremented.

Figure 17-13—Port Timers state machine

During initialization (i.e., when BEGIN is TRUE) the state machine continually enters the ONE_SECOND state, clearing the tick signal. Following initialization, each tick causes a transition to the TICK state. On entry to TICK, all non-zero timer variables are decremented by one. The state machine then unconditionally transitions to the ONE_SECOND state to clear the tick variable and await the next tick.

17.23 Port Receive state machine

The Port Receive state machine shall implement the function specified by the state diagram in Figure 17-14, the definitions in 17.16, and the variable declarations and procedures specified in 17.17 through 17.21. It receives each valid BPDU (9.3.4), setting rcvdMsg to communicate the BPDU’s arrival to the Port Information Machine (17.27), and using the updtBPDUversion procedure (17.21.22) to set either rcvdRSTP (17.19.28) or rcvdSTP (17.19.29) to communicate the BPDU’s arrival and type to the Port Protocol Migration machine (17.24).

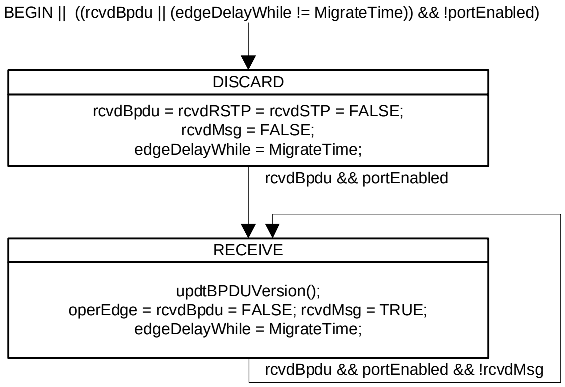

BEGIN || ((rcvdBpdu || (edgeDelayWhile != MigrateTime)) && !portEnabled)
DISCARD
rcvdBpdu = rcvdRSTP = rcvdSTP = FALSE;
rcvdMsg = FALSE;
edgeDelayWhile = MigrateTime;
rcvdBpdu && portEnabled
RECEIVE
updtBPDUVersion();
operEdge = rcvdBpdu = FALSE; rcvdMsg = TRUE;
edgeDelayWhile = MigrateTime;
rcvdBpdu && portEnabled && !rcvdMsg

Figure 17-14—Port Receive state machine

17.24 Port Protocol Migration state machine

The Port Protocol Migration state machine shall implement the function specified by the state diagram in Figure 17-15, the definitions in 17.16, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19. It updates sendRSTP (17.19.38) to tell the Port Transmit state machine (17.26) which BPDU types (9.3) to transmit, to support interoperability (17.4) with the Spanning Tree Algorithm and Protocol specified in previous revisions of this standard.

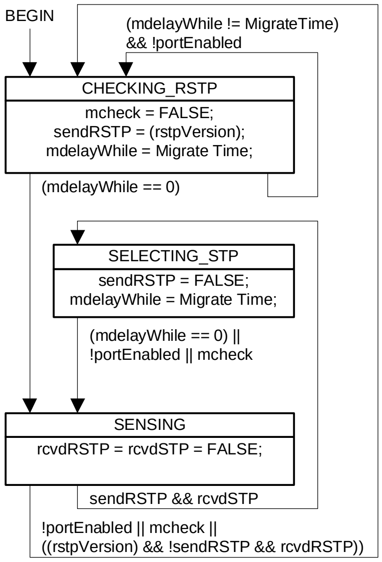

BEGIN
(mdelayWhile != MigrateTime)
&& !portEnabled
CHECKING_RSTP
mcheck = FALSE;
sendRSTP = (rstpVersion);
mdelayWhile = Migrate Time;
(mdelayWhile == 0)
SELECTING_STP
sendRSTP = FALSE;
mdelayWhile = Migrate Time;
(mdelayWhile == 0) ||
!portEnabled || mcheck
SENSING
rcvdRSTP = rcvdSTP = FALSE;
sendRSTP && rcvdSTP
!portEnabled || mcheck ||
((rstpVersion) && !sendRSTP && rcvdRSTP))

Figure 17-15—Port protocol migration state machine

17.25 Bridge Detection state machine

The Bridge Detection state machine shall implement the function specified by the state diagram in Figure 17-16, the definitions in 17.16, 17.13, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19.

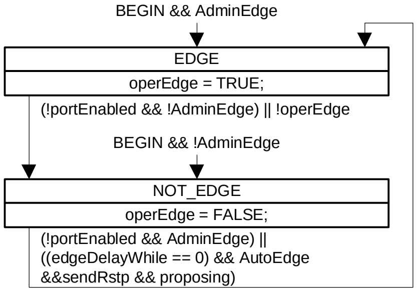
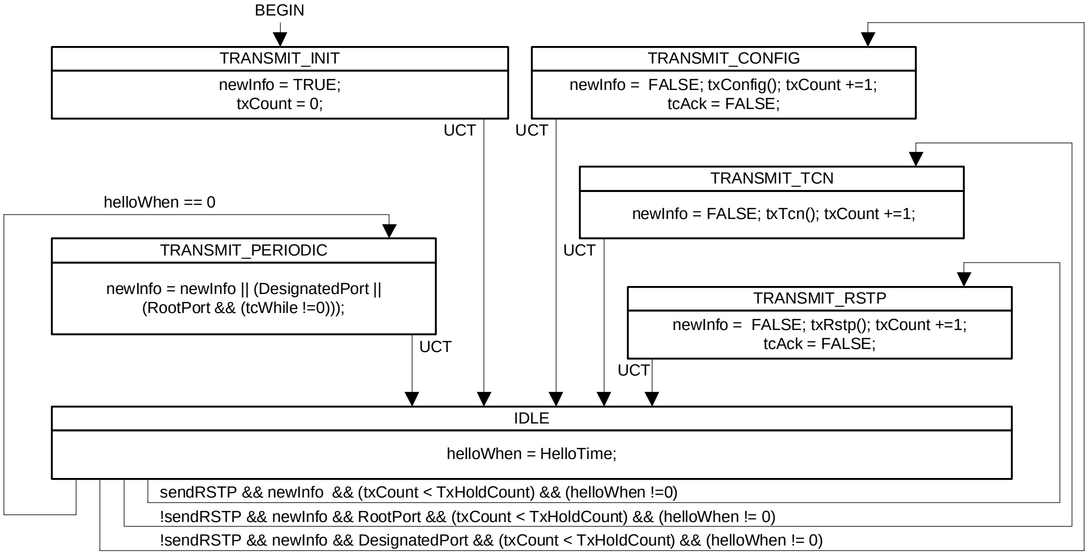


Figure 17-16—Bridge Detection state machine

17.26 Port Transmit state machine

The Port Transmit state machine shall implement the function specified by the state diagram contained in Figure 17-17, the definitions in 17.13, 17.16, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19. It transmits BPDUs at regular intervals and when newInfo (17.19.16) is set, using sendRSTP (17.19.38) to determine their Type, and txCount (17.19.44) and TxHoldCount (17.13.12) to rate limit transmission. At least one BPDU per HelloTime (17.13.6) interval, and not more than (TxHoldCount + 1) BPDUs in one second, are transmitted.


BEGIN
TRANSMIT_INIT
newInfo = TRUE;
txCount = 0;
UCT
TRANSMIT_CONFIG
newInfo = FALSE; txConfig(); txCount +=1;
tcAck = FALSE;
UCT
TRANSMIT_TCN
newInfo = FALSE; txTcn(); txCount +=1;
UCT
helloWhen == 0
TRANSMIT_PERIODIC
newInfo = newInfo || (DesignatedPort ||
(RootPort && (tcWhile !=0)));
UCT
TRANSMIT_RSTP
newInfo = FALSE; txRstp(); txCount +=1;
tcAck = FALSE;
UCT
IDLE
helloWhen = HelloTime;
sendRSTP && newInfo && (txCount < TxHoldCount) && (helloWhen !=0)
!sendRSTP && newInfo && RootPort && (txCount < TxHoldCount) && (helloWhen != 0)
!sendRSTP && newInfo && DesignatedPort && (txCount < TxHoldCount) && (helloWhen != 0)

All transtions, except UCT, are qualified by "&& selected &&!updtInfo".

Figure 17-17—Port Transmit state machine

17.27 Port Information state machine

The Port Information state machine shall implement the function specified by the state diagram in Figure 17-18, the definitions in 17.13, 17.16, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19.

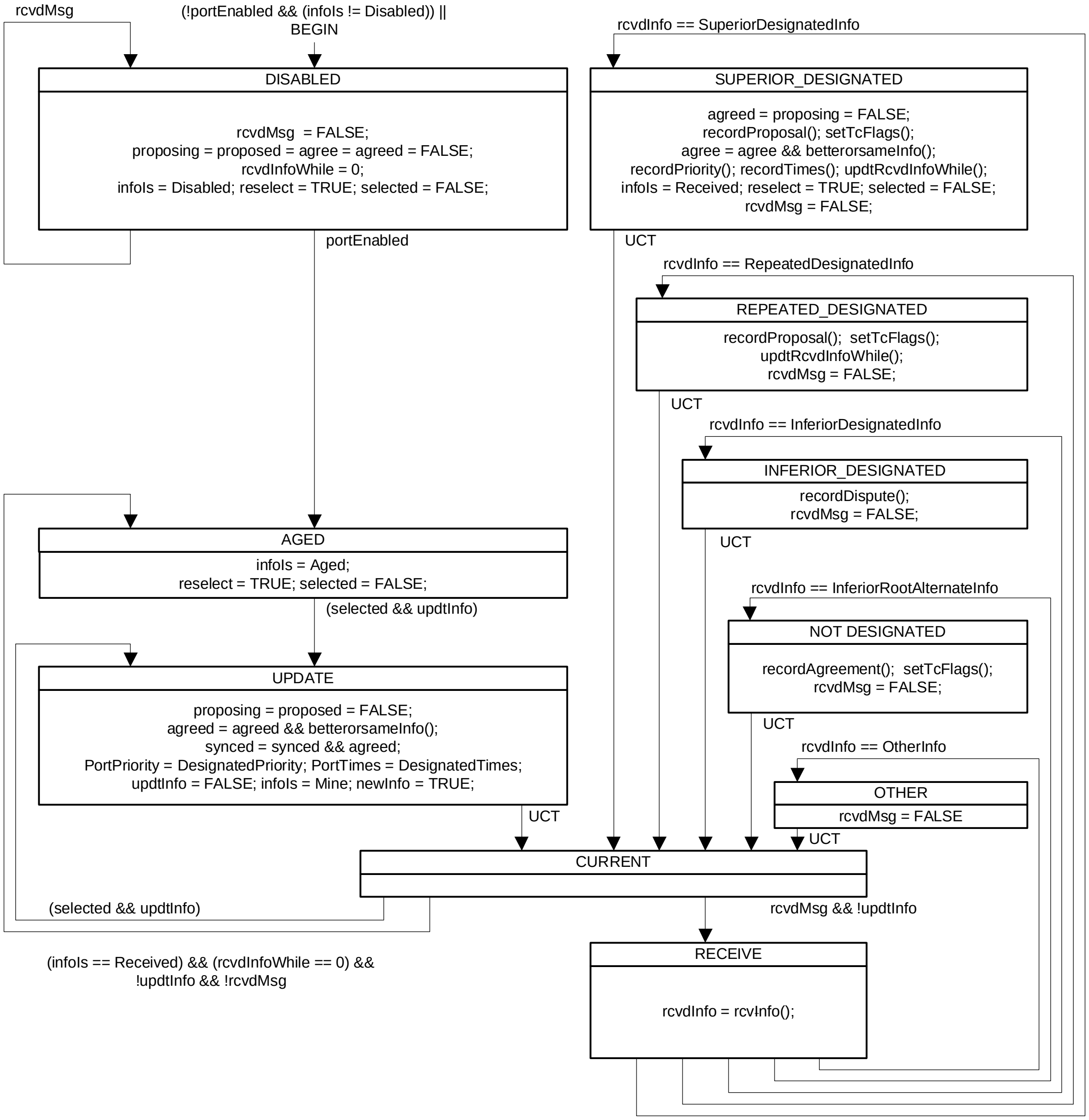

rcvdMsg
(!portEnabled && (infoIs != Disabled)) ||
BEGIN
rcvdInfo == SuperiorDesignatedInfo
DISABLED
rcvdMsg = FALSE;
proposing = proposed = agree = agreed = FALSE;
rcvdInfoWhile = 0;
infoIs = Disabled; reselect = TRUE; selected = FALSE;
SUPERIOR_DESIGNATED
agreed = proposing = FALSE;
recordProposal(); setTcFlags();
agree = agree && betterorsameInfo();
recordPriority(); recordTimes(); updtRcvdInfoWhile();
infoIs = Received; reselect = TRUE; selected = FALSE;
rcvdMsg = FALSE;
portEnabled
UCT
rcvdInfo == RepeatedDesignatedInfo
REPEATED_DESIGNATED
recordProposal(); setTcFlags();
updtRcvdInfoWhile();
rcvdMsg = FALSE;
UCT
rcvdInfo == InferiorDesignatedInfo
INFERIOR_DESIGNATED
recordDispute();
rcvdMsg = FALSE;
UCT
AGED
infoIs = Aged;
reselect = TRUE; selected = FALSE;
(selected && updtInfo)
rcvdInfo == InferiorRootAlternateInfo
NOT DESIGNATED
recordAgreement(); setTcFlags();
rcvdMsg = FALSE;
UPDATE
proposing = proposed = FALSE;
agreed = agreed && betterorsameInfo();
synced = synced && agreed;
PortPriority = DesignatedPriority; PortTimes = DesignatedTimes;
updtInfo = FALSE; infoIs = Mine; newInfo = TRUE;
UCT
rcvdInfo == OtherInfo
OTHER
rcvdMsg = FALSE
UCT
UCT
CURRENT
(selected && updtInfo)
rcvdMsg && !updtInfo
(infoIs == Received) && (rcvdInfoWhile == 0) &&
!updtInfo && !rcvdMsg
RECEIVE
rcvdInfo = rcvInfo();

Figure 17-18—Port Information state machine

This state machine is responsible for updating and recording the source (infoIs, 17.19.10) of the Spanning Tree information (portPriority 17.19.21, portTimes 17.19.22) used to test the information conveyed (msgPriority, 17.19.14; msgTimes, 17.19.15) by received Configuration Messages. If new, superior, information arrives on the port, or the existing information is aged out, it sets the reselect variable to request the Port Role Selection state machine to update the spanning tree priority vectors held by the Bridge and the Bridge's Port Roles.

17.28 Port Role Selection state machine

The Port Role Selection state machine shall implement the function specified by the state diagram in Figure 17-19, the definitions in 17.13, 17.16, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19. It selects roles for all Bridge Ports.

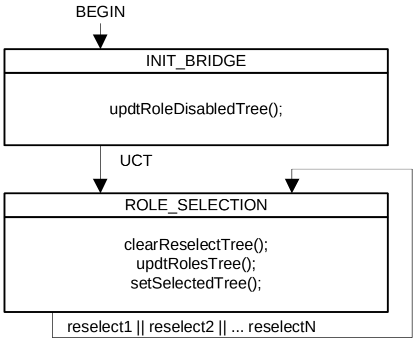


Figure 17-19—Port Role Selection state machine

On initialization all Bridge Ports are assigned the Disabled Port Role. Whenever any Bridge Port’s reselect variable (17.19.34) is set by the Port Information state machine (17.27), spanning tree information including the designatedPriority (17.19.4) and designatedTimes (17.19.5) for each Port is recomputed and its Port Role (selectedRole, 17.19.37) updated by the updtRolesTree() procedure (17.21.25). The reselect variables are cleared before computation starts so that recomputation will take place if new information becomes available while the computation is in progress.

17.29 Port Role Transitions state machine

The Port Role Transitions state machine shall implement the function specified by the state diagrams in Figure 17-22 and Figure 17-23, the definitions in 17.13, 17.16, 17.20, and 17.21, and the variable declarations in 17.17, 17.18, and 17.19. All four figures represent parts of the same state machine, so any global transition is a possible exit transition from any of the states shown.

Port Roles (17.7) are assigned by the Port Role Selection (17.28) state machine, which updates selectedRole (17.19.37). If the Port Role Transitions’ role variable (17.19.35) is not equal to selectedRole, a global transition to the part of the machine that handles the new role takes place.

Two variables, learn and forward (see 17.19.8, 17.19.11), are used by this state machine to request the Port State Transitions (17.30) machine to change the Port State (7.4). Two further variables, learning and forwarding (17.19.9, 17.19.12), indicate when the Port State transition has actually occurred. State transitions in the Port Role Transitions machine that depend upon the actual Port State are qualified by the current value of the learning or forwarding variables. State transitions that request changes in Port State are qualified by the current value of the learn or forward variables, to avoid repeating a request to change to the same Port State.

17.29.1 Disabled Port states

Figure 17-20 shows initialization of the Port Role Transitions state machine and the states associated with the Disabled Port role.

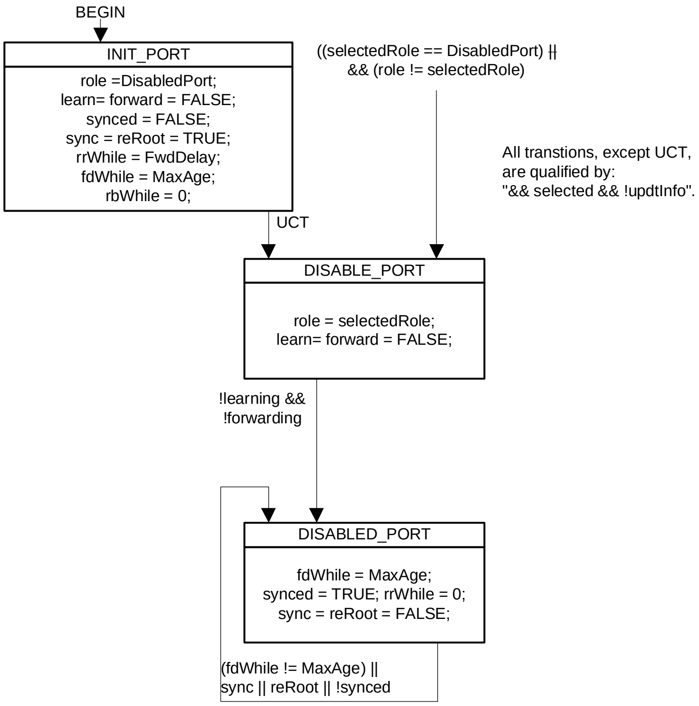


Figure 17-20—Disabled Port role transitions

NOTE—The Port Role Selection state machine will set selectedRole to DisabledPort on initialization.

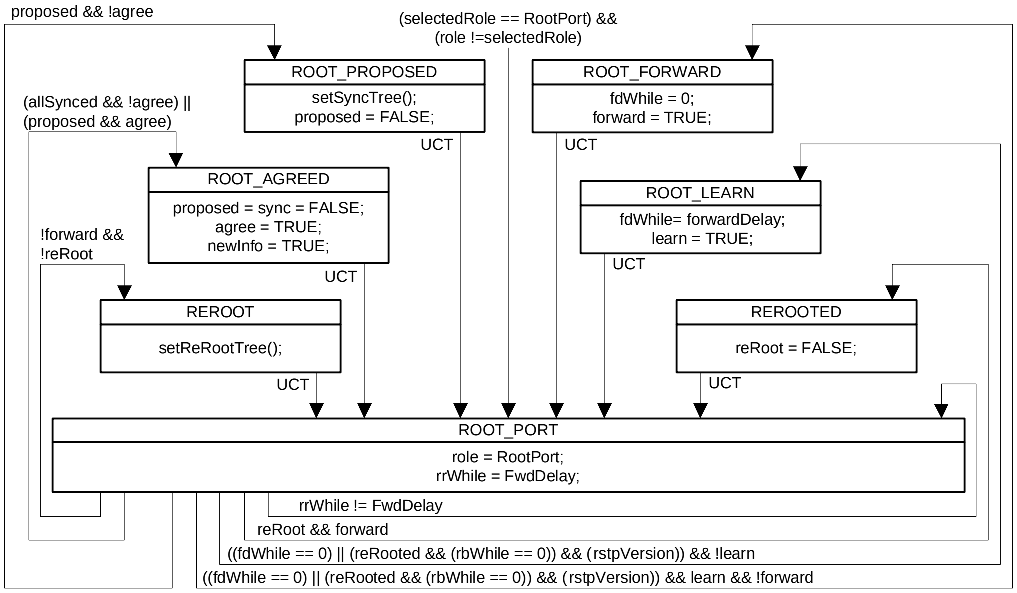
17.29.2 Root Port states

Figure 17-21 shows the states associated with the Root Port role.

A transition to this role does not necessarily modify the Port State. In particular a Designated Port that was Forwarding or awaiting expiry of the fdWhile timer (17.17.2) to transition to Forwarding does not necessarily revert to Discarding or modify fdWhile. However, if it is not Forwarding, it will use the setReRootTree() procedure (17.21.15) to set reRoot (17.19.33) for all Bridge Ports, instructing all recent roots to transition to Discarding. A Root Port maintains the rrWhile timer (17.17.7) at FwdDelay (17.20.6), once it is no longer a Root Port rrWhile is allowed to expire, and will be set to zero if the Port becomes Discarding (Figure 17-20, Figure 17-22, Figure 17-23). Once all recent roots have been retired (reRooted, 17.20.10) the Root Port can transition to Learning and to Forwarding.

A Root Port's transition to Forwarding is delayed if it was recently a Backup Port, i.e., rbWhile (17.17.5) is running. All rapid transitions are disabled and forwarding delays imposed if rstpVersion (17.20.11) is FALSE.

If all the Bridge's Ports are synchronized with its spanning tree priority information, an Agreement (17.10) is transmitted through the Root Port, and receipt of a Proposal will cause the Root Port to request the other Ports to synchronize their state.

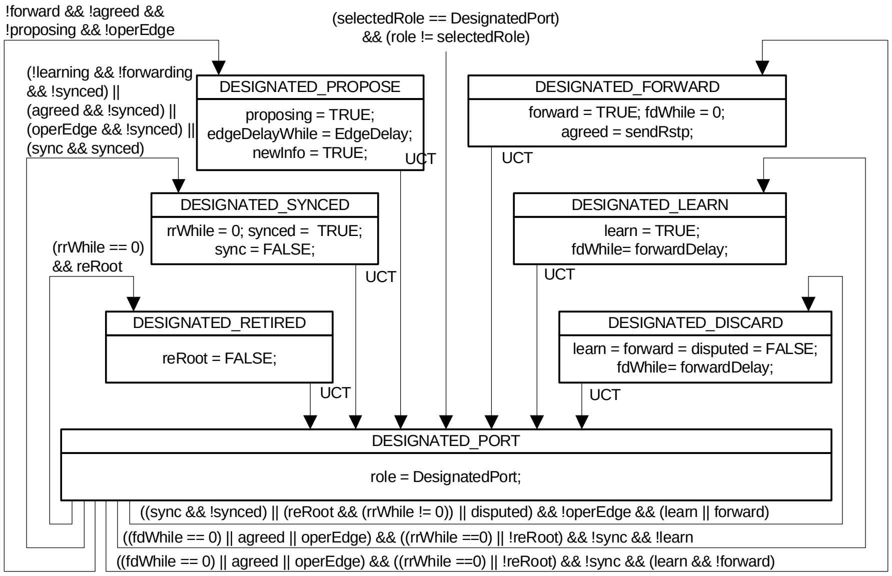


All transitions, except UCT, are qualified by "&& selected && !updtInfo".

Figure 17-21—Root Port role transitions

17.29.3 Designated Port states

Figure 17-22 shows the states associated with the Designated Port role.


!forward && !agreed &&
!proposing && !operEdge
(selectedRole == DesignatedPort)
&& (role != selectedRole)
(!learning && !forwarding
&& !synced) ||
(agreed && !synced) ||
(operEdge && !synced) ||
(sync && synced)
DESIGNATED_PROPOSE
proposing = TRUE;
edgeDelayWhile = EdgeDelay;
newInfo = TRUE;
UCT
DESIGNATED_FORWARD
forward = TRUE; fdWhile = 0;
agreed = sendRstp;
UCT
DESIGNATED_SYNCED
rrWhile = 0; synced = TRUE;
sync = FALSE;
UCT
DESIGNATED_LEARN
learn = TRUE;
fdWhile= forwardDelay;
UCT
(rrWhile == 0)
&& reRoot
DESIGNATED_RETIRED
reRoot = FALSE;
UCT
DESIGNATED_DISCARD
learn = forward = disputed = FALSE;
fdWhile= forwardDelay;
UCT
DESIGNATED_PORT
role = DesignatedPort;
((sync && !synced) || (reRoot && (rrWhile != 0)) || disputed) && !operEdge && (learn || forward)
((fdWhile == 0) || agreed || operEdge) && ((rrWhile ==0) || !reRoot) && !sync && !learn
((fdWhile == 0) || agreed || operEdge) && ((rrWhile ==0) || !reRoot) && !sync && (learn && !forward)

All transitions, except UCT, are qualified by "&& selected && !updtInfo".

Figure 17-22—Designated port role transitions

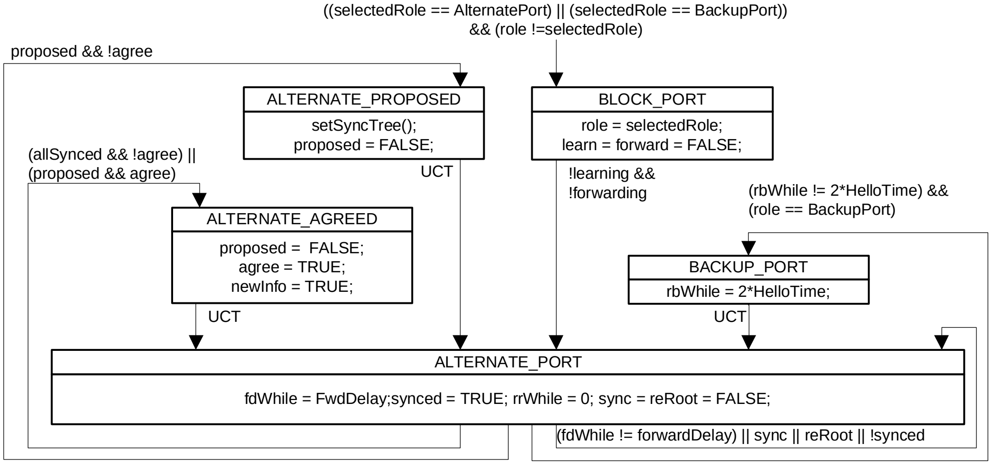
A transition to this role does not necessarily modify the Port State. A Designated Port that was Forwarding or awaiting expiry of the fdWhile timer (17.17.2) will not necessarily change state or modify fdWhile. However, a new Root Port that is not already Forwarding will set reRoot to cause recent roots to revert to Discarding. If this port is Discarding or not a recent root it will clear reRoot.

A Root or Alternate Port that has received a Proposal (17.10) can set sync (17.19.39) for this Port, causing it to synchronize its Port State with spanning tree information, transitioning to Discarding if it has not received a prior Agreement. Once synchronized, this port will set synced (17.19.40).

If the Port Receive state machine (17.23) receives an inferior RST BPDU from a Port that believes itself to be a Designated Port and is Learning or Forwarding it will set disputed (17.19.6), causing this state machine to transition a Designated Port to Discarding.

17.29.4 Alternate and Backup Port states

Figure 17-22 shows the states associated with the Alternate and Backup Port roles.



All transitions, except UCT, are qualified by "&& selected && !updtInfo".

Figure 17-23—Alternate and Backup Port role transitions

17.30 Port State Transition state machine

The Port State Transition state machine shall implement the function specified by the state diagram in Figure 17-24, the definitions in 17.13, 17.16, 17.20, and 17.21, and variable declarations in 17.17, 17.18, and 17.19.

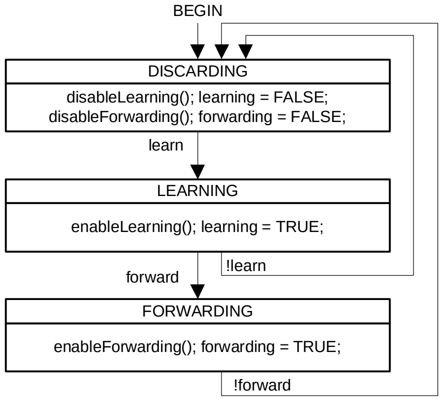

BEGIN
DISCARDING
disableLearning(); learning = FALSE;
disableForwarding(); forwarding = FALSE;
learn
LEARNING
enableLearning(); learning = TRUE;
!learn
forward
FORWARDING
enableForwarding(); forwarding = TRUE;
!forward

Figure 17-24—Port State Transition state machine

This state machine models changes in the Port State (7.4). The Port Role Transitions state machine requests changes by setting the learn and forward variables; the Port State Transitions machine updates the learning and forwarding variables as the actual Port State changes. The disableLearning(), disableForwarding(), enableLearning(), and enableForwarding() procedures model the system-dependent actions and delays that take place; these procedures do not complete until the desired behavior has been achieved.

17.31 Topology Change state machine

The Topology Change state machine shall implement the function specified by the state diagram in Figure 17-25, the definitions in 17.13, 17.16, 17.20, and 17.21, and variable declarations in 17.17, 17.18, and 17.19.

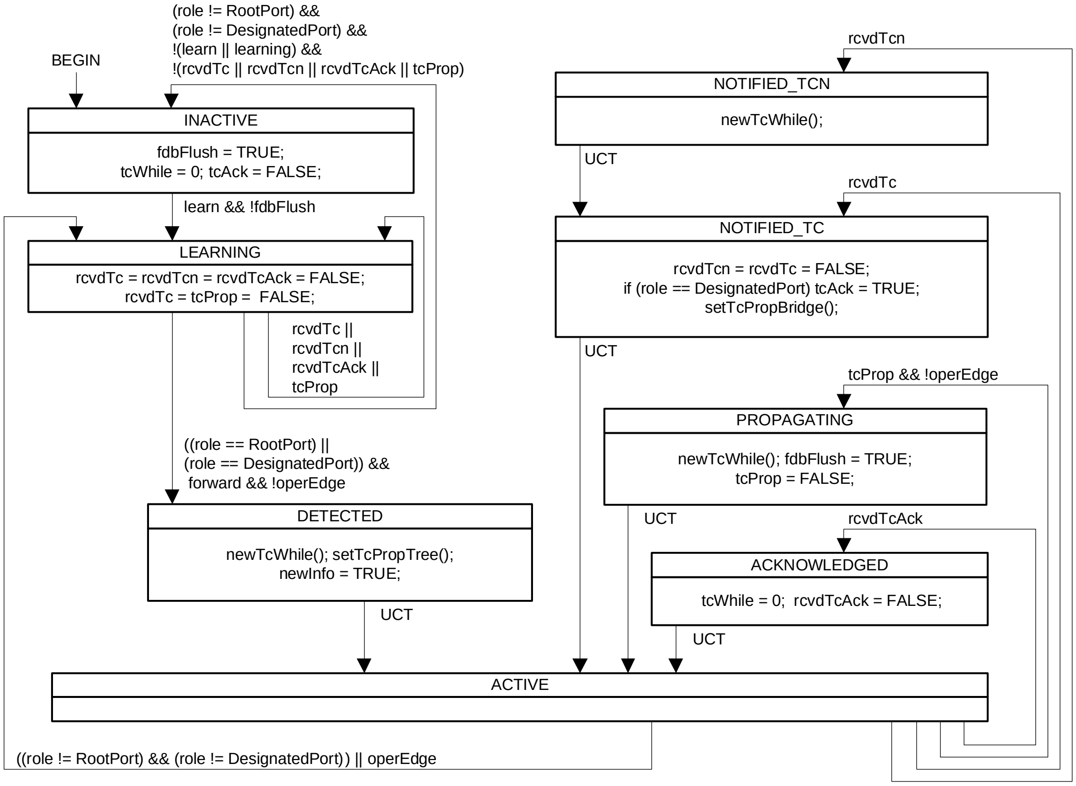

BEGIN
(role != RootPort) &&
(role != DesignatedPort) &&
!(learn || learning) &&
!(rcvdTc || rcvdTcn || rcvdTcAck || tcProp)
INACTIVE
fdbFlush = TRUE;
tcWhile = 0; tcAck = FALSE;
learn && !fdbFlush
LEARNING
rcvdTc = rcvdTcn = rcvdTcAck = FALSE;
rcvdTc = tcProp = FALSE;
rcvdTc ||
rcvdTcn ||
rcvdTcAck ||
tcProp
((role == RootPort) ||
(role == DesignatedPort)) &&
forward && !operEdge
DETECTED
newTcWhile(); setTcPropTree();
newInfo = TRUE;
UCT
rcvdTcn
NOTIFIED_TCN
newTcWhile();
UCT
rcvdTc
NOTIFIED_TC
rcvdTcn = rcvdTc = FALSE;
if (role == DesignatedPort) tcAck = TRUE;
setTcPropBridge();
UCT
tcProp && !operEdge
PROPAGATING
newTcWhile(); fdbFlush = TRUE;
tcProp = FALSE;
UCT
rcvdTcAck
ACKNOWLEDGED
tcWhile = 0; rcvdTcAck = FALSE;
UCT
ACTIVE
((role != RootPort) && (role != DesignatedPort)) || operEdge

Figure 17-25—Topology Change state machine

This state machine is responsible for topology change detection, notification, and propagation, and for instructing the Filtering Database to remove Dynamic Filtering Entries for certain ports (17.11).

17.32 RSTP performance requirements

This subclause places requirements on the performance of the Spanning Tree Protocol Entities of Bridges in a Bridged Local Area Network to ensure that RSTP operates correctly.

The delay between the occurrence of an external event and the action or actions mandated by the RSTP specification as a consequence of the event shall not exceed the Maximum RSTP processing delay specified in Table 17-5.

External events subject to this provision of this specification shall include the following:

a) Transmission of a BPDU by another Bridge on a LAN to which the Bridge is attached.

Specified actions subject to this provision shall include the following:

b) Transmissions of BPDUs on all Ports mandated by the RSTP specification, with the exception of transmissions delayed by the Port Transmit state machine (17.26) to enforce transmit rate limits.
c) Ceasing to learn or forward frames.

The delay between internal timer related events and the transmission of all BPDUs on Ports mandated by the RSTP specification as a consequence shall not exceed the Maximum BPDU transmission delay specified in Table 17-5. Timer events subject to this provision of this specification shall include the following:

d) Decrementing of txCount by the Port Transmit machine, thus allowing a BPDU transmission if transmit rate limits were being enforced.
e) Expiry of the helloWhen timer (17.17.3).

Table 17-5—Transmission and reception delays

Parameter	Absolute maximum value
Maximum RSTP processing delay	1.0 second
Maximum BPDU transmission delay	0.2 second

18. Bridge Detection state machine

In IEEE Std 802.1t-2001, this clause specified a Bridge Detection state machine that checked the validity of an administrative setting of a Bridge Port as an Edge Port. The functionality of this state machine has been superseded by the RSTP Bridge Detection state machine in Clause 17.

Annex A

(normative)

PICS Proforma[10]

A.1 Introduction

The supplier of a protocol implementation that is claimed to conform to this standard 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

a) By the protocol implementor, as a checklist to reduce the risk of failure to conform to the standard through oversight.
b) 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.
c) 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 PICSs).
d) 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
O	optional
O.n	optional, but support of at least one of the group of options labelled by the same numeral *n* is required
X	prohibited
pred:	conditional-item symbol, including predicate identification: see A.3.4
¬	logical negation, applied to a conditional item’s predicate

A.2.2 General abbreviations

N/A	not applicable
PICS	Protocol Implementation Conformance Statement

[10] *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.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 several 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 or No), or by entering a value or a set or range of values. (Note that there are some items where two or more choices from a set of possible answers can apply; all relevant choices are to be marked.)

Each item is identified by an item reference in the first column. The second column contains the question to be answered; the third column records the status of the item—whether support is mandatory, optional, or conditional; see also A.3.4. The fourth column contains the reference or references to the material that specifies the item in the main body of this standard, and the fifth column provides the space for the answers.

A supplier may also provide (or be required to provide) further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labelled *Ai* or *Xi*, respectively, for cross-referencing purposes, where *i* is any unambiguous identification for the item (e.g., simply a numeral). There are no other restrictions on its format and presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformation 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, in case that 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 will be supplied, and a PICS can be considered complete without any such information. Examples 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 information about aspects of the implementation that are outside the scope of this standard but that have a bearing upon the answers to some items.

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 occasionally happen that a supplier will wish to answer an item with mandatory status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No pre-printed answer will be found in the Support column for this: instead, the supplier shall write the missing answer into the Support column, together with an *Xi* reference to an item of Exception Information, and shall provide the appropriate rationale in the Exception item itself.

An implementation for which an Exception item is required in this way does not conform to this standard.

NOTE—A possible reason for the situation described previously is that a defect in this standard 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 both the applicability of the item itself, and its status (if it does apply)—mandatory or optional—are dependent upon whether or not certain other items are supported.

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 a conditional symbol 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, and S is a status symbol, M or 0.

If the value of the predicate is true (see A.3.4.2), the conditional item is applicable, and its status is indicated by the status symbol following the predicate: the answer column is to be marked in the usual way. If the value of the predicate is false, the “Not Applicable” (N/A) answer is to be marked.

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 as a boolean expression constructed by combining item-references using the boolean operator OR: the value of the predicate is true if one or more of the items is marked as supported.
c) A predicate-name, for a predicate defined as a boolean expression constructed by combining item-references using the boolean operator AND: the value of the predicate is true if all of the items are marked as supported
d) The logical negation symbol “¬” 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 “¬” symbol is false, and vice versa.

Each item whose reference is used in a predicate or predicate definition, or in a preliminary question for grouped conditional items, is indicated by an asterisk in the Item column.

A.4 PICS proforma for IEEE Std 802.1D

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 names	
NOTE 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. NOTE 2—The terms Name and Version should be interpreted appropriately to correspond with a supplier's terminology (e.g., Type, Series, Model).	

A.4.2 Protocol summary, IEEE Std 802.1D

Identification of protocol specification	IEEE Std 802.1D, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges
Identification of amendments and corrigenda to the PICS proforma that have been completed as part of the PICS	Amd. : Corr. : Amd. : Corr. :
Have any Exception items been required? (See A.3.3: the answer Yes means that the implementation does not conform to IEEE Std 802.1D.)	No [ ] Yes [ ]

Date of Statement	

A.5 Major Capabilities

Item	Feature	Status	References	Support
MAC	Do the implementations of MAC Technologies and support of the MAC Internal Sublayer Service conform to MAC standards as specified in 6.4 and 6.5? (If support of a specific MAC technology is claimed any PICS Proforma(s) required by the Standard specifying that technology shall also be completed.)	M	6.4, 6.5. A.6	Yes [ ]
LLC	Is a class of LLC supporting Type 1 operations supported on all Bridge Ports in conformance with IEEE Std 802.2? (The PICS Proforma required by IEEE Std 802.2 shall also be completed.)	M	7.2, 7.3, 7.12. IEEE Std 802.2	Yes [ ]
RLY	Does the implementation relay and filter frames as specified?	M	7.1, 7.5, 7.6, and 7.7. A.7	Yes [ ]
BFS	Does the implementation maintain the information required to make frame filtering decisions and support Basic Filtering Services?	M	7.1, 7.5, 7.8, and 7.9. A.8	Yes [ ]
ADDR	Does the implementation conform to the provisions for addressing?	M	7.12 A.9	Yes [ ]
RSTP	Is the Rapid Spanning Tree Protocol implemented?	M	17 A.10	Yes [ ]
BPDU	Are transmitted BPDUs encoded and received BPDUs validated as specified?	M	9, 17.21.19, 17.21.20, and17.21.21. A.11	Yes [ ]
IMP	Are the required implementation parameters included in this completed PICS?	M	7.9 A.12	Yes [ ]
PERF	Are the required performance parameters included in this completed PICS? (Operation of the Bridge within the specified parameters shall not violate any of the other conformance provisions of this standard.)	M	16 A.13	Yes [ ]
MGT	Is management of the Bridge supported?	O	14 A.14	Yes [ ] No [ ]
RMGT	Is a remote management protocol supported?	**MGT:**O	5.2 A.15	Yes [ ] No [ ]
TC	Are multiple Traffic Classes supported for relaying frames?	O	7.7.3, 7.7.4. A.16	Yes [ ] No [ ]
EFS	Are Extended Filtering Services supported for relaying and filtering frames?	O	7.12 A.17	Yes [ ] No [ ]
GMRP	Is the GARP Multicast Registration Protocol (GMRP) implemented?	**EFS:**M	10 A.18	Yes [ ] N/A[ ]
GARP	Is the Generic Attribute Registration Protocol (GARP) implemented in support of the GMRP Application?	**GMRP:**M	12 A.18	Yes [ ] N/A[ ]

A.6 Media Access Control Methods

Item	Feature	Status	References	Support
	Which Media Access Control methods are implemented in conformance with the relevant MAC Standards?		6.4, 6.5	
MAC-802.3	CSMA/CD, IEEE Std 802.3	O.1		Yes [ ] No [ ]
MAC-802.5	Token Ring, IEEE Std 802.5	O.1		Yes [ ] No [ ]
MAC-9314-2	FDDI, ISO 9314-2	O.1		Yes [ ] No [ ]
MAC-802.11	Wireless LAN, IEEE Std 802.11	O.1		Yes [ ] No [ ]
MAC-1	Has a PICS been completed for each of the Media Access Control methods implemented as required by the relevant MAC Standards?	M		Yes[ ]
MAC-2	Do all the Media Access Control methods implemented support the MAC Internal Sublayer Service as specified.	M	6.4, 6.5	Yes [ ]
MAC-3	Are the adminPointToPointMAC and operPointToPointMAC parameters implemented on all Ports?	M	6.4, 6.5	Yes [ ]
MAC-4	Does the implementation support the use of the adminEdgePort and operEdgePort parameters on any Ports?	O	6.4.2	Yes [ ] No [ ]
MAC-4a	State which Bridge Ports support the adminEdgePort and operEdgePort parameters			Ports_______
MAC-5	Is the user_priority of received frames set to the Default User Priority where specified for the MAC?	M	6.5.1, 6.5.4	Yes[ ]
MAC-6	Can the Default User Priority be set for each Port	O	6.5.1, 6.5.4	Yes [ ] No [ ]
MAC-7	Can the Default User Priority be set to any of 0–7?	**MAC-6:**M	6.5.1, 6.5.4	Yes[ ]
MAC-8	Is an M_UNITDATA.indication generated by the FDDI MAC entity for a Port on receipt of frame transmitted by that entity?	FDDI:X	6.5.3, ISO 9314-2	No [ ] N/A[ ]
MAC-9	Is only Asynchronous service used on FDDI rings?	FDDI:M	ISO 9314-2 Clause 8.1.4	Yes [ ] N/A[ ]
MAC-10	Is the C indicator set on receipt of a frame for forwarding from an FDDI ring?	FDDI:O.2	6.5.3, ISO 9314-2 Clause 7.3.8	Yes [ ] No [ ] N/A[ ]
MAC-11	Is the C indicator unaltered on receipt of a frame for forwarding from an FDDI ring?	FDDI:O.2	6.5.3, ISO 9314-2 Clause 7.3.8	Yes [ ] No [ ] N/A[ ]

Predicates:
FDDI = MAC-9314-2[Yes]

A.7 Relay and filtering of frames

Item	Feature	Status	References	Support
RLY-1	Are received frames with media access method errors discarded?	M	6.4, 7.5	Yes[]
RLY-2	Are user data frames the only type of frame relayed?	M	7.5, 7.6	Yes [ ]
RLY-3	Is the user priority of each frame relayed regenerated as specified?	M	7.5.1, 7.7.5	Yes [ ]
RLY-4	Are the default values of the User Priority Regeneration Table as specified for each Port?	M	7.5.1, Table 7-1	Yes [ ]
RLY-5	Can the User Priority Regeneration Table be modified?	O	7.5.1, Table 7-1	Yes [ ]
RLY-6	Can the entries in the User Priority Regeneration Table be set independently for each user priority and Port and to any of the full range of values?	**RLY-5:** M	7.5.1, Table 7-1	Yes [ ]
RLY-7	Are frames transmitted by an LLC User attached at a Bridge Port also submitted for relay?	M	7.6	Yes [ ]
RLY-8	Are correctly received user data frames relayed subject to the conditions imposed by the Forwarding Process?	M	7.7, 7.7.1, 7.7.2, 7.7.2, 7.9, Tables 7-6, 7-7, 7-8	Yes [ ]
RLY-9	Is the order of relayed frames preserved as required by the forwarding process?	M	7.7.3	Yes [ ]
RLY-10	Is a relayed frame submitted to a MAC Entity for transmission only once?	M	7.7.3	Yes [ ]
RLY-11	Is a maximum bridge transit delay enforced for relayed frames?	M	7.7.3, Table 7-3	Yes [ ]
RLY-12	Are queued frames discarded if a Port leaves the Forwarding State?	M	7.7.3	Yes [ ]
RLY-13	Is the default algorithm for selecting frames for transmission supported?	M	7.7.4	Yes [ ]
RLY-14	Is the access priority of each transmitted frame as specified for each media access method?	M	7.7.5, Table 7-4	Yes [ ]
RLY-15	Is the FCS of frames relayed between Ports of the same MAC type preserved?	O	7.7.6	Yes [ ] No [ ]
RLY-16	Is the undetected frame error rate greater than that achievable by preserving the FCS?	X	7.7.6	No [ ]

A.8 Basic Filtering Services

Item	Feature	Status	References	Support
BFS-1	Are correctly received user data frames submitted to the Learning Process?	M	7.5	Yes [ ]
BFS-2	Are correctly received frames of types other than user data frames submitted to the Learning Process?	O	7.5	Yes [ ] No [ ]
BFS-3	Does the Filtering Database support creation and update of Dynamic Filtering Entries by the Learning Process?	M	7.8, 7.9, 7.9.2	Yes [ ]
BFS-4	Are Dynamic Filtering Entries created and updated if and only if the Port State permits?	M	7.8, 7.9.2	Yes [ ]
BFS-5	Are Dynamic Filtering Entries created on receipt of frames with a group source address?	X	7.8, 7.9.2	No [ ]
BFS-6	Can a Dynamic Filtering Entry be created that conflicts with an existing Static Filtering Entry?	X	7.8, 7.9, 7.9.1, 7.9.2	No [ ]
BFS-7	Are existing Dynamic Filtering Entries removed to allow creation of a new entry if the Filtering Database is full?	O	7.8, 7.9.2	Yes [ ] No [ ]
BFS-8	Does the Filtering Database contain Static Filtering Entries?	M	7.9.1	Yes [ ]
BFS-9	Are Static Filtering Entries aged out?	X	7.9	No [ ]
BFS-10	Can Static Filtering Entries be created, modified, and deleted by management?	O	7.9	Yes [ ] No [ ]
BFS-11	Can Static Filtering Entries be made for individual and Group MAC Addresses?	**10:**M	7.9.1	Yes [ ] N/A[ ]
BFS-12	Can a Static Filtering Entry be made for the broadcast MAC Address?	**10:**M	7.9.1	Yes [ ] N/A[ ]
BFS-13	Can a Static Filtering Entry specify a forwarding Port Map?	**10:**M	7.9.1	Yes [ ] N/A[ ]
BFS-14	Can a Static Filtering Entry specify a filtering Port Map?	**10:**M	7.9.1	Yes [ ] N/A[ ]
BFS-15	Does the creation of a Static Filtering Entry remove any conflicting information in a Dynamic Filtering Entry for the same address?	M	7.9.1, 7.9.2	Yes [ ]
BFS-16	Can a separate Static Filtering Entry with a Port Map be created for each inbound Port?	O	7.9.1	Yes [ ] No [ ]
BFS-17	Are Dynamic Filtering Entries aged out of the Filtering Database if not updated?	M	7.9.2	Yes [ ]
BFS-18	Can more than one Dynamic Filtering Entry be created for the same MAC Address?	X	7.9.2	No [ ]
BFS-19	Can the Bridge be configured to use the recommended default Ageing Time?	O	7.9.2, Table 7-5	Yes [ ] No [ ]
BFS-20	Can the Bridge be configured to use any value in the range specified for Ageing Time?	O	7.9.2, Table 7-5	Yes [ ] No [ ]
BFS-21	Is the Filtering Database initialized with the entries contained in the Permanent Database?	M	7.9.6	Yes [ ]

A.9 Addressing

Item	Feature	Status	References	Support
ADDR-1	Does each Port have a separate MAC Address?	M	7.12.2	Yes [ ]
ADDR-2	Are frames addressed to a MAC Address for a Port and received from or relayed to the attached LAN submitted to LLC Service User for the destination LLC Address?	M	7.5, 7.12.2	Yes [ ]
ADDR-3	Are all BPDUs and GARP PDUs transmitted using the Bridge Spanning Tree Protocol LLC Address?	M	7.12.3, Table 7-9	Yes [ ]
ADDR-4	Are PDUs addressed to the Bridge Spanning Tree Protocol Address with an unknown Protocol Identifier discarded on receipt	M		Yes [ ]
ADDR-5	Are all BPDUs transmitted to the Bridge Group Address?	M	7.12.3, Table 7-10	Yes [ ]
ADDR-6	Are all GARP PDUs transmitted to the Group Address assigned for the GARP Application?	M	7.12.3, Table 12-1	Yes [ ]
ADDR-7	Is it possible to create entries in the Permanent or Filtering Databases for unsupported GARP application addresses or delete or modify entries for supported application addresses?	X	7.12.3	No [ ]
ADDR-8	Is the source MAC address of BPDUs and GARP PDUs for GARP Applications supported by the Bridge the address of the transmitting Port?	M	7.12.3	Yes [ ]
ADDR-9	Is Bridge Management accessible through a Port using the MAC Address of the Port?	**MGT:**O	7.12.4	Yes [ ] No [ ]
ADDR-10	Is a 48-bit Universally Administered MAC Address assigned to each Bridge as its Bridge Address?	M	7.12.5	Yes [ ]
ADDR-11	Is the Bridge Address the Address of a Port?	O	7.12.5	Yes [ ] No [ ]
ADDR-12	Is the Bridge Address the Address of Port 1?	**ADDR-11:** O	7.12.5	Yes [ ] No [ ]
ADDR-13	Are frames addressed to any of the Reserved Addresses relayed by the Bridge?	X	7.12.6	No [ ]
ADDR-14	Is it possible to delete or modify entries in the Permanent and Filtering Databases for the Reserved Addresses?	X	7.12.6	No [ ]

A.10 Rapid Spanning Tree Protocol

Item	Feature	Status	References	Support
RSTP-1	Does each Bridge have a unique identifier based on the Bridge Address, and a unique identifier for each Port?	M	17.2	Yes [ ]
RSTP-2	Can each Port be configured as an edge port by setting the adminEdgePort parameter?	O	17.3	Yes [ ] No [ ]
RSTP-3	Can each Port be configured to automatically determine if it an edge port by setting the autoEdgePort parameter?	O	17.3	Yes [ ] No [ ]
RSTP-4	Are learned MAC addresses transferred from a retiring Root Port to a new Root Port?	O	17.11	Yes [ ] No [ ]
RSTP-5	Is the Spanning Tree Protocol Entity reinitialized if the Force Protocol Version parameter is modified?	M	17.13	Yes [ ]
RSTP-6	Are spanning tree priority vectors and Port Role assignments recomputed if the Bridge Identifier Priority, Port Identifier Priority, or Port Path Costs change?	M	17.13	Yes [ ]
RSTP-7	Is the txCount variable for a Port set to zero if the Port’s Transmit Hold Count is modified?	M	17.13	Yes [ ]
RSTP-8	Are the recommended default values of Migrate Time, Bridge Hello Time, Bridge Max Age, Bridge Forward Delay, and Transmit Hold Count used?	O	17.14	Yes [ ] No [ ]
RSTP-9	Can the Bridge Max Age, Bridge Forward Delay, and Transmit Hold Count parameters be set?	O	17.13, 17.14	Yes [ ] No [ ]
RSTP-10	Can Bridge Max Age, Bridge Forward Delay, Transmit Hold Count be set to any value in the permitted range?	**RSTP-9:** M	17.2, 17.15, Table 17-1	Yes [ ] N/A[ ]
RSTP-11	Are the relationships between Bridge Hello Time, Bridge Max Age, and Bridge Forward Delay enforced?	**RSTP-9:** M	17.14	Yes [ ] N/A[ ]
RSTP-12	Are the recommended values of Bridge Identifier Priority, Port Path Costs, and Port Identifier Priorities used?	O	17.14	Yes [ ] No [ ]
RSTP-13	Can the Bridge Identifier Priority, Port Path Costs, and Port Identifier Priorities be set?	O	17.1, 17.3.1, 17.13, 17.14, 17.18, 17.19	Yes [ ] No [ ]
RSTP-14	Can the Bridge Identifier Priority and Port Identifier Priorities be set to any of the values in the ranges specified?	**RSTP-13:** M	17.14	Yes [ ] N/A[ ]
RSTP-15	Can the Port Path Cost for each Port be set to any of the values in the specified range?	**RSTP-13:** M	17.14	Yes [ ] N/A[ ]
RSTP-16	Are Port Path Costs changed automatically by default if port speeds change?	X	17.14	No [ ]
RSTP-17	Is one instance of the Port Role Selection state machine implemented for the Bridge; one instance of each of the Port Timers, Port Receive, Port Protocol Migration, Bridge Detection, Port Transmit, Port Information, Port Role Transition, Port State Transition, and Topology Change state machines implemented per Port; and the referenced definitions and declarations followed for all machines?	M	17.15, 17.28, 17.22, 17.23, 17.24, 17.25, 17.26, 17.27, 17.29, 17.29, 17.30, 17.31	Yes [ ]
RSTP-18	Is it possible to set each Port Protocol Migration state machine’s mcheck variable?	O	17.19.13	Yes [ ] No [ ]
RSTP-19	Is a single instance of each of the timer variables implemented per Port?	M	17.22	Yes [ ]
RSTP-20	Are the values for maximum RSTP processing delay and maximum BPDU transmission delay ever exceeded for any of the specified external events, actions, internal events, or transmissions?	X	17.32, Table 17-5	No [ ]

A.11 BPDU Encoding

Item	Feature	Status	References	Support
BPDU-1	Do all BPDUs contain an integral number of octets?	M	9.1.1	Yes [ ]
BPDU-2	Are all the following BPDU parameter types encoded as specified?	M	9.1.1, 9.2	Yes [ ]
	Protocol Identifiers		9.2.1	
	Protocol Version Identifiers		9.2.2	
	BPDU Types		9.2.3	
	Flags		9.2.4	
	Bridge Identifiers		9.2.5	
	Root Path Cost		9.2.6	
	Port Identifiers		9.2.7	
	Timer Values		9.2.8	
BPDU-3	Do Configuration BPDUs have the format, parameters, and parameter values specified?	M	9.3.1, 17.21.19	Yes [ ]
BPDU-4	Do Topology Change Notification BPDUs have the format, parameters, and parameter values specified?	M	9.3.2, 17.21.21	Yes [ ]
BPDU-5	Do Rapid Spanning Tree BPDUs have the format, parameters, and parameter values specified?	M	9.3.3, 17.21.20	Yes [ ]
BPDU-6	Are received BPDUs validated as and only as specified?	M	9.3.4	Yes [ ]
BPDU-7	Does the implementation process BPDUs of prior and possible later protocol versions as specified?	M	9.3.4	Yes [ ]

A.12 Implementation Parameters

Item	Feature	Status	References	Support
IMP-1	State the Filtering Database Size.	M	7.9	____ entries
IMP-2	State the Permanent Database Size.	M	7.9	____ entries

A.13 Performance

Item	Feature	Status	References	Support
PERF-1	Specify a Guaranteed Port Filtering Rate, and the associated measurement interval T_F, for each Bridge Port in the format specified below.	M	16.1	
PERF-2	Specify a Guaranteed Bridge Relaying Rate, and the associated measurement interval T_R, in the format specified below. Supplementary information shall clearly identify the Ports.	M	16.2	

Guaranteed Bridge Relaying Rate	T_R
_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)

Port number(s) or other identification	Guaranteed port filtering rate (specify for all ports)	T_F (specify for all ports)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)
	_ _ _ _ _ _ _ _ _ _ _ frames per second	_ _ _ _ _ _ second(s)

A.14 Bridge management

Item	Feature	Status	References	Support
	Are each of the following management operations supported?			
MGT-1	Discover Bridge	**MGT:**M	14.4.1.1	Yes [ ]
MGT-2	Read Bridge	**MGT:**M	14.4.1.2	Yes [ ]
MGT-3	Set Bridge Name	**MGT:**M	14.4.1.3	Yes [ ]
MGT-4	Reset Bridge	**MGT:**M	14.4.1.4	Yes [ ]
MGT-5	Read Port	**MGT:**M	14.4.2.1	Yes [ ]
MGT-6	Set Port Name	**MGT:**M	14.4.2.2	Yes [ ]
MGT-7	Read Forwarding Port Counters	**MGT:**M	14.6.1.1	Yes [ ]
MGT-8	Read Port Default User Priority	**MGT:** M	14.6.2.1	Yes [ ]
MGT-9	Set Port Default User Priority	**MGT AND MAC-6:** M	14.6.2.2	Yes [ ] N/A[ ]
MGT-10	Read Port User Priority Regeneration Table	**MGT AND RLY-5:** M	14.6.2.3	Yes [ ] N/A[ ]
MGT-11	Set Port User Priority Regeneration Table	**MGT AND RLY-5:** M	14.6.2.4	Yes [ ] N/A[ ]
MGT-12	Read Port Traffic Class Table	**MGT AND TC:** M	14.6.3.1	Yes [ ] N/A[ ]
MGT-13	Set Port Traffic Class Table	**MGT AND TC-3:** M	14.6.3.2	Yes [ ] N/A[ ]
MGT-14	Read Outbound Access Priority Table	**MGT :** M	14.6.3.3	Yes [ ]
MGT-15	Read Filtering Database	**MGT:**M	14.7.1.1	Yes [ ]
MGT-16	Set Filtering Database Ageing Time	**MGT:**M	14.7.1.2	Yes [ ]
MGT-17	Read Permanent Database	**MGT:**M	14.7.5.1	Yes [ ]
MGT-18	Create Filtering Entry	**MGT:**M	14.7.6.1	Yes [ ]
MGT-19	Delete Filtering Entry	**MGT:**M	14.7.6.2	Yes [ ]
MGT-20	Read Filtering Entry	**MGT:**M	14.7.6.3	Yes [ ]
MGT-21	Read Filtering Entry Range	**MGT:**M	14.7.6.4	Yes [ ]
MGT-22	Read Spanning Tree Protocol Parameters	**MGT:**M	14.8.1.1	Yes [ ]
MGT-23	Set Spanning Tree Protocol Parameters	**MGT:**M	14.8.1.2	Yes [ ]
MGT-24	Read Port Parameters	**MGT:**M	14.8.2.1	Yes [ ]
MGT-25	Force Port State	**MGT:**M	14.8.2.2	Yes [ ]
MGT-26	Set Port Parameters	**MGT:**M	14.8.2.3	Yes [ ]
MGT-27	Force BPDU Migration Check	**MGT AND RSTP:** M	14.8.2.4	Yes [ ] N/A[ ]
MGT-28	Read GARP Timers	**MGT AND GARP:** M	14.9.1.1	Yes [ ] N/A[ ]
MGT-29	Set GARP Timers	**MGT AND GARP:** M	14.9.1.2	Yes [ ] N/A[ ]
MGT-30	Read GARP Applicant Controls	**MGT AND GARP:** M	14.9.2.1	Yes [ ] N/A[ ]
MGT-31	Set GARP Applicant Controls	**MGT AND GARP:** M	14.9.2.2	Yes [ ] N/A[ ]
MGT-32	Read GARP State	**MGT AND GARP:** M	14.9.3.1	Yes [ ] N/A[ ]

A.15 Remote Management

Item	Feature	Status	References	Support
RMGT-1	What Management Protocol standard(s) or specification(s) are supported?	RMGT:M	5.2	
RMGT-2	What standard(s) or specifications for Managed Objects and Encodings are supported?	RMGT:M	5.2	

A.16 Expedited Traffic Classes

Item	Feature	Status	References	Support
TC-1	Does the implementation provide more than one transmission queue for (a) Bridge Port(s)?	**TC:** M	7.7.3	Yes [ ] N/A[ ]
TC-2	Is the recommended mapping of user_priority to traffic classes supported for each Port?	**TC:** O	7.7.3	Yes [ ] No [ ]
TC-3	Can the traffic class tables be managed?	**TC:** O	7.7.3, Table 7-2	Yes [ ] No [ ]
TC-4	Is the default algorithm for selecting frames for transmission supported?	M	7.7.4	Yes [ ]
TC-5	Are additional algorithms for selecting frames for transmission supported?	O	7.7.4	Yes [ ] No [ ]

A.17 Extended Filtering Services

Item	Feature	Status	References	Support
EFS-1	Can Group Registration Entries be created, updated and removed from the Filtering Database by GMRP?	EFS:M	7.9, 7.9.3, 10	Yes [ ] N/A[ ]
EFS-2	Can a Static Filtering Entry be created with an address specification that represents a Group Address, or All Group Addresses, or All Unregistered Group Addresses, and with a control element for each Port that specifies unconditional forwarding, or unconditional filtering, or the use of dynamic or default group filtering information?	EFS:M	7.9.1	Yes [ ] N/A[ ]
EFS-3	Can a Static Filtering Entry be created with an address specification that represents an Individual Address and with a control element for each Port that specifies unconditional forwarding, or unconditional filtering?	M	7.9.1	Yes [ ]
EFS-4	Can a Static Filtering Entry be created with an address specification that represents an Individual Address and with a control element for each Port that specifies unconditional forwarding, or unconditional filtering, or the use of dynamic filtering information?	EFS:O	7.9.1	Yes [ ] No [ ] N/A[ ]

A.18 GMRP

Item	Feature	Status	References	Support
GMRP-1	Does GMRP operate within the Base Spanning Tree Context, with the GIP Context identifier of 0, and propagate registration information only on the active topology?	**GMRP:**M	10.3.1.1, 10.4.1, 12.2.3, 12.2.4	Yes [ ] N/A[ ]
GMRP-2	Is the GMRP Address used as the destination MAC Address in all GMRP protocol exchanges?	**GMRP:**M	10.3.1.2, 10.4.1, Table 12-1	Yes [ ]
GMRP-3	Do the PDUs exchanged by the GARP state machines use the PDU formats, attribute types, and value encodings defined for GMRP?	**GMRP:**M	10.3.1, 10.4.1, 12.3, 12.4, 12.10	Yes [ ]
GMRP-4	Are GMRP PDUs and the messages they contain processed in the order received?	**GMRP:**M	12.10	Yes [ ]
GMRP-5	Do values of the Group Attribute type include individual MAC Addresses?	**GMRP:**X	10.3.1.4	No [ ]
GMRP-6	Does the GMRP application operate as defined?	**GMRP:**M	10, 10.3, 10.4.1	Yes [ ]
GMRP-7	Can the Static Filtering Entry that specifies All Groups with Registration Fixed for all Ports be deleted from the Permanent Database?	**GMRP:**O	10.3.2.3	Yes [ ] No [ ]
GMRP-8	Is the use of the Restricted Group Registration parameter supported for each Port?	**GMRP:**O	10.3.2.2, 10.3.2.3	Yes [ ] No [ ]
GMRP-9	Is the creation or modification of Dynamic Group Registration Entries restricted as specified if the Restricted Group Registration control is TRUE?	**GMRP-8:**O	10.3.2.2, 10.3.2.3	Yes [ ] No [ ]
GMRP-10	Is the Restricted Group Registration control FALSE for all Ports?	¬**GMRP-8:**O	10.3.2.3	Yes [ ] No [ ]
GMRP-11	Does the implementation support the operation of the GARP Applicant, Registrar, and Leave All state machines?	**GMRP:**M	10.4.1, 12.7, 13	Yes [ ]

A.19 GARP

Item	Feature	Status	References	Support
GARP-1	Does the GARP Entity transmit PDUs to or process PDUs from any port that is not MAC_Operational or is not authorized?	**GARP:**X	12.1	No [ ]
GARP-2	Are GARP PDUs destined for Applications that the Bridge supports relayed by the Bridge?	**GARP:**X	7.12.3, 12.4	No [ ]
GARP-3	Are all GARP PDUs destined for Applications that the Bridge does not support relayed by the Bridge?	M	7.12.3, 12.4, 12.10	Yes [ ]
GARP-4	Do GARP protocol exchanges use LLC Type 1 procedures, and the Bridge Spanning Tree Protocol LLC address?	**GARP:**M	12.3, 12.4, Table 7-9	Yes [ ]
GARP-5	Are received GARP PDUs that are not well formed for the GARP Applications supported, discarded?	**GARP:**M	12.3, 12.4, 12.9.1, 12.10	Yes [ ]
GARP-6	Are information items that are received in well formed PDUs but not understood, individually discarded?	**GARP:**M	12.9.1, 12.10.3	Yes [ ]
GARP-7	Are the state machines, administrative controls, and procedures required by each supported application implemented as specified?	**GARP:**M	12.7, 12.8, 12.9	Yes [ ]
GARP-8	Are the generic elements of GARP PDUs formatted for transmission and processed on reception by each GARP Application as specified?	**GARP:**M	12.10	Yes [ ]
GARP-9	Is the resolution of GARP timers as specified?	**GARP:**M	12.11.2	Yes [ ]

Annex B

(informative)

Calculating spanning tree parameters

In IEEE Std 802.1D, 1998 Edition, and earlier editions, this annex described the calculation of recommended values and ranges of performance parameters for the Spanning Tree Protocol (STP) specified in Clause 8 of those editions. The Rapid Spanning Tree Protocol (RSTP) is not sensitive to the values of these parameters. The ranges of RSTP performance parameters included in Table 17-1 use the values previously calculated for STP to facilitate interoperability with STP Bridges implemented according to prior editions of this standard.

Annex C

(normative)

Source-routing transparent bridge operation

C.1 Overview

A Source-Routing Transparent (SRT) Bridge is a MAC Bridge that performs source routing when frames are received with routing information (RII=1), and that performs Transparent Bridging when frames are received without routing information (RII=0). This annex specifies the protocols for the operation of source routing in an SRT Bridge. Source-routing frame formats and Bridge operations will facilitate end-station source routing of frames. Source-routing operation by end stations is specified in IEEE Std 802.2. Source routing provides the following:

a) Greater utilization of resources by providing multiple routes through the Bridged Local Area Network.
b) Greater individual control of frames through the Bridged Local Area Network.

C.1.1 Scope

For the purpose of compatible interconnection of data-processing equipment using source routing on a Bridged Local Area Network, this standard specifies the operation of MAC Bridges supporting source routing. To this end it

a) Defines the principles of operation of the MAC Bridge in providing source routing, in terms of the support and preservation of the MAC Service, and the maintenance of QoS.
b) Specifies the enhanced MAC Internal Sublayer Service as it pertains to the SRT Bridge.
c) Augments the architectural model of the internal operation of a Bridge.
d) Establishes the requirements for Bridge Management of the source-routing function in the Bridged Local Area Network, identifying the basic managed objects and management operations.
e) Specifies the requirements to be satisfied by equipment claiming conformance to this standard.

Items for future study are as follows:

— Performance requirements, and recommended default values and applicable ranges for the operational parameters of a Bridge.

In addition, SRT operation is defined for only those MACs that have defined the routing information field. As of this publication, they include the following:

— Token Ring (IEEE Std 802.5).
— FDDI MAC (ISO 9314-2).

MAC-specific matters are discussed in C.2.5.

C.1.2 Definitions

C.1.2.1 ARE Rd limit

This limit represents the maximum allowable number of route descriptors in an All Routes Explorer frame. This limit is implemented in Bridges to limit the number of Bridges traversed by All Routes Explorer frames between end stations during the route determination process.

C.1.2.2 Explorer frame

A frame to which the SRT Bridge adds routing information as it forwards the frame. The explorer frame is used to discover routes. There is an All Routes Explorer, which is intended to be forwarded by every Port; and a Spanning Tree Explorer frame, which is forwarded only by Ports designated to forward them by the spanning tree protocol.

C.1.2.3 LAN-in ID (LIN)

The identifier of the LAN from which an SRT Bridge receives a frame.

C.1.2.4 LAN-out ID (LOUT)

The identifier of the LAN to which an SRT Bridge transmits a frame.

C.1.2.5 parallel bridges

Two or more Bridges connecting the same LANs.

C.1.2.6 route

A path through a series of LANs and SRT Bridges.

C.1.2.7 route control

The first two-octet field in the routing information field of a source-routed frame. This field indicates the type and characteristics of the frame to be source routed.

C.1.2.8 route descriptor

A two-octet field in the routing information field that designates a LAN ID and Bridge number.

C.1.2.9 route discovery

The process of obtaining a route to a destination station.

C.1.2.10 routing information

A route control field and a sequence of route descriptors consisting of LAN IDs and Bridge numbers, indicating a route through the network between two stations.

C.1.2.11 source routing

A bridging mechanism to route frames through a multi-LAN network by specifying in the frame a route it will traverse.

C.1.2.12 spanning tree

A topology of Bridges such that there is one, and only one, data route between any two end stations.

C.1.2.13 STE Rd Limit

This limit represents the maximum allowable number of route descriptors in a Spanning Tree Explorer frame. This limit is implemented in Bridges to limit the number of Bridges traversed by Spanning Tree Explorer frames.

C.1.2.14 Transparent Bridging

A bridging mechanism, which is transparent to end stations, that interconnects LANs by Bridges designated to forward frames through participation in a Spanning Tree Algorithm.

C.1.2.15 Abbreviations

The following abbreviations are specific to this standard among the family of IEEE 802 standards:

ARE	All Routes Explorer: RII=1, RT=10x
ARERdLim	ARE Route Descriptor Limit
BN	Bridge Number
BPP	Bridge Port Pair
LIN	LAN-in ID
LOUT	LAN-out ID
LTH	Length Field
MSDU	MAC Service Data Unit
NSR	Non-Source Routed: RII=0, no RI Field
RC	Routing Control
RD	Route Descriptor
RI	Routing Information
RII	Routing-Information Indicator
RT	Routing Type
SRF	Specifically Routed Frame: RII=1, RT=0xx
SRT	Source Routing Transparent (Bridge)
STE	Spanning Tree Explorer: RII=1, RT=11x
STERdLim	STE Route Descriptor Limit

C.1.3 Conformance

A MAC Bridge that claims conformance to this annex

a) Shall conform to the main body of this standard, augmented by this annex, and represent a superset of the functions described in the main body of this standard for frames with routing information.
b) Shall conform to the static and dynamic requirements given in C.1.3.1 and C.1.3.2.
c) Furthermore, the supplier of an implementation that is claimed to conform to this standard shall complete a copy of the PICS proforma provided in Annex D (normative), and shall provide the information necessary to identify both the supplier and the implementation.

C.1.3.1 Static conformance requirements

An implementation claiming conformance to this standard

a) Shall conform with the MAC Sublayer and Physical Layer of the LANs it interconnects.
b) Shall conform to the static characteristics of the SRT Bridge as described in C.3.

C.1.3.2 Dynamic conformance requirements

An implementation claiming conformance to this standard

a) Shall exhibit external behavior corresponding to the Bridging operations for the handling of source-routed frames as described in C.3.
b) Shall report errors as described in C.3.

C.2 Support of the MAC Service

C.2.1 Support of the MAC Service

The MAC Service provided to end stations attached to a Bridged Local Area Network is supported by the Bridges in that network as described in 6.2.

C.2.2 Preservation of the MAC Service

SRT Bridges preserve the MAC Service as described in 6.2.

C.2.3 Quality of service maintenance

The quality of the MAC Service supported by an SRT Bridge is not significantly inferior to that provided by a single LAN. A full set of QoS parameters is included in 6.3. Parameters that are affected by the source-routing capability are listed as follows:

a) Frame misordering
b) Frame duplication
c) Undetected frame error rate
d) Maximum Service Data Unit Size

Frames without routing information are referred to as Non-Source-Routed Frames (NSR). Source routing introduces three new types of frames. Specifically, Source-Routed Frames (SRFs) are the primary mechanism for transporting data through the source-routed network. Spanning Tree Explorer frames (STE) are sent on the portion of the spanning tree interconnected by SRT Bridges; they collect the spanning tree routing information as they travel. STE frames can be used for management and route determination but are not required for data transmission. All Routes Explorer Frames (ARE) are used explicitly during route discovery.

C.2.3.1 Frame misordering

Source routing allows multiple routes to exist and be used simultaneously between a given pair of stations. If frames are sent alternately on different routes, they might arrive in a different order. However, misordering does not occur along any particular route within each type. Bridges receiving frames of the same type (e.g., SRFs) will transmit them in a First In First Out (FIFO) manner. Regarding the specific types of frames,

a) NSR frames cannot be misordered (see the main body of this standard) because they follow the unique path of the spanning tree.

b) STE frames cannot be misordered (see the main body of this standard) because they follow the unique path of the spanning tree.

c) ARE frames are used as control frames. Misordering does not apply for these frames since they are sent on multiple routes by definition.

d) SRFs will not be misordered provided the sender uses one and only one source route to deliver the sequence of frames; frames that use different routes between the same pair of Ports are susceptible to frame misordering.

C.2.3.2 Frame duplication

a) NSR a nd STE frames cannot be duplicated (see the main body of this standard).

b) ARE frames are used as control frames; the receiving side must accept multiple ARE frames (one for each route).

c) SRFs cannot be duplicated because frames are forwarded only by the Bridge sequence specified in the routing information field. Furthermore, Bridges prevent frames with duplicate LAN numbers in the routing sequence from being forwarded.

C.2.3.3 Undetected frame error rate

The service provided by the MAC Sublayer introduces a very low undetected frame error rate in transmitted frames, see 6.3.7. In particular, the following should be noted:

a) The rate of undetected errors for NSR and SRFs is unchanged.

b) ARE and STE frames collect routing information as they travel to their destination; therefore, the FCS is recalculated at each intermediate Bridge. These frames can have a higher undetected frame error rate, but they are not intended to be used for data transport.

C.2.3.4 Maximum service data unit size supported

For a complete description of the Maximum Service Data Unit Size, see 6.3.8. The SRT Bridge meets all requirements associated with Maximum Service Data Unit Size. The Maximum Service Data Unit Size supported by a Bridge between two LANs is the smaller of that supported by the LANs. No attempt is made by a Bridge to relay a frame to a LAN that does not support the size of Service Data Unit conveyed by that frame.

Source routing reduces Bridge receipt of frames of unacceptable sizes since the sender of the data can determine the maximum size of the data unit on the chosen route during route discovery.

C.2.4 Internal sublayer service

The MAC entities provide an additional enhanced service with a new set of primitives in addition to the Internal Sublayer Service specified in 6.4.

The Internal Sublayer Service excludes procedures whose operation is confined to that of the individual LANs. The unitdata primitives that describe this service follow. The parameters and their meaning are identical to those in 6.4 except for the addition of the following parameter:

routing_information

C.2.4.1 Interactions

The following primitives are defined for the MAC relay to request the MAC to transmit a PDU.

M_UNITDATA.request
M_UNITDATA.indication

C.2.4.2 Detailed service specification

All primitives are specified as examples only. Each service names the particular primitive and the required information that is passed between MAC and the MAC relay function. Only those primitives not included in 6.4 are defined here.

C.2.4.2.1 M_UNITDATA.indication

Each MAC data indication primitive invoked corresponds to the receipt of a frame from an individual LAN.

Semantics of the service primitive

```
M_UNITDATA.indication       (
                            frame_type,
                            destination_address,
                            source_address,
                            routing_information,
                            mac_service_data_unit,
                            user_ priority,
                            frame_check_sequence
                            )
```

routing_information. The routing_information parameter specifies the routing information field of the received frame.

When generated. This primitive is generated by the MAC entity to the MAC relay function to indicate the arrival of a frame to be forwarded. Such frames are reported only if they are validly formed (as defined by the respective MACs).

Effect of receipt. The receipt of this primitive causes the MAC relay function to process the frame according to the forwarding logic specified in C.3.7.

C.2.4.2.2 M_UNITDATA.request

A data request primitive is invoked to transmit a frame to an individual LAN.

Semantics of the service primitive

```
M_UNITDATA.request      (
                        frame_type,
                        destination_address,
                        source_address,
                        routing_information,
                        mac_service_data_unit,
                        user_priority,
                        access priority,
                        frame check sequence
```

routing_information. The routing_information parameter specifies the routing information field of the frame to be forwarded as modified by the Bridge function.

NOTE—If the Frame Check Sequence (FCS) is supplied, then it may be forwarded without recalculation; otherwise, it should be calculated.

When generated. This primitive is generated by the MAC relay function to the MAC entity when the MAC relay passes a frame to the MAC entity for forwarding.

Effect of receipt. The receipt of this primitive causes the MAC entity to append all the appropriate fields, and pass the properly formed frame to the lower layers of the protocol for transfer to the peer MAC entity or entities.

C.2.5 Support of the internal sublayer service

This subclause specifies the mapping of the Internal Sublayer Service to MAC Protocol and Procedures, and the encoding of the parameters of the service in MAC frames.

C.2.5.1 Support of token ring

The Token Ring Access Method is specified in IEEE Std 802.5. Clause 3 of that standard specifies Formats and Facilities, and Clause 4 specifies Token Ring protocols.

On receipt of an M_UNITDATA.request primitive, the local MAC Entity composes a frame using the parameters supplied as specified below, appending the frame control, destination address, source address, routing information, and frame check sequence fields to the user data. The Routing Information Indicator bit is set to one to indicate the presence of the routing information field. The frame is enqueued for transmission. On transmission, the starting delimiter, access control field, ending delimiter, and frame status fields are added.

On receipt of a valid frame with a frame_type of user_data_frame, with the Routing Information Indicator bit set to one, an M_UNITDATA.indication primitive is generated, with parameters derived from the frame fields as specified below.

NOTE—On receipt of a valid MAC frame that was not transmitted by the Bridge Port’s local MAC entity with the Routing Information Indicator bit set to zero, an M_UNITDATA.indication primitive is generated as required by 6.5.2, and the frame is handled by the Transparent Bridging function.

The frame_type, destination_address, source_address, mac_service_data_unit, user_priority, and frame_check_sequence parameter are encoded as specified in 6.5.2.

The Routing Information Indicator bit is encoded in the source address field. It occupies the same position in the source address field as does the Group Address indicator bit in the destination address field.

If an M_UNITDATA.request primitive is not accompanied by the frame_check_sequence parameter, it is calculated in accordance with 3.2.7 of IEEE Std 802.5.

For the setting of the A and C bits, refer to 6.5.2.

In order to support the MAC Internal Sublayer Service, a Token Ring Bridge must be capable of recognizing and removing frames transmitted by itself, even though they can carry a source address different than that of the Bridge Port that transmitted them.

C.2.5.2 Support of FDDI

On receipt of a valid frame (ISO 9314-2) that was not transmitted by the Bridge Port's local MAC entity, an M_UNITDATA.indication primitive is generated. The parameters associated with the primitive are derived from the frame's fields as specified in 6.5.3.

On receipt of an M_UNITDATA.request primitive, the local MAC entity composes a frame using the parameters supplied as specified in 6.5.3.

C.3 Principles of operation

This clause uses the principles and model of operation of an SRT Bridge. It provides the following:

a) An explanation of the principal elements of source-routing Bridge operation and a list of the supporting functions
b) An architectural model that governs the provision of these functions
c) A model of the operation of the SRT Bridge in terms of the operation of processes and entities that support the functions
d) Details concerning the additional addressing requirements

C.3.1 Source-routing bridge operation

The principal elements of source-routing operation are as follows:

a) Relay of source-routed data frames
b) Relay and processing of frames to support the transfer and acquisition of routing information
c) Management of the above

C.3.1.1 Relay of data frames

A MAC Bridge relays individual MAC user data frames between the separate MACs of the LANs connected to its Ports. The order of frames of given user_priority and frame_type received on one Port and transmitted on another is preserved.

The functions that support the relay of source-routed data frames and maintain the QoS supported by the Bridge are as follows:

a) Frame reception
b) Discard of received frame in error
c) Selection of source-routed data frames
d) Frame discard if transmittable service data unit size exceeded (C.2.3.4)
e) Modification of routing information
f) Frame discard when maximum Bridge transit delay is exceeded
g) Selection of outbound access priority
h) Frame transmission

C.3.1.2 Dissemination of routing information

The functions that support the relay and processing of frames to support the transfer and acquisition of routing information are as follows:

a) Handling of All Routes Explorer (ARE) frames
b) Handling of Spanning Tree Explorer (STE) frames

C.3.1.3 Bridge management

The functions that support Bridge Management control and monitor the provision of the functions in the subclauses above are specified in C.4.

C.3.2 Bridge architecture

Each Bridge Port receives and transmits frames to and from the LAN to which it is attached using the services provided by the individual MAC entity associated with that port as described in 7.2. The MAC relay entity handles the MAC-independent functions of relaying frames between Bridge Ports. If the received frame is not source routed (RII = 0), then the Bridge frame is forwarded or discarded using the logic described in Clause 7 (Transparent Bridging logic). If the received frame is source routed (RII = 1), then the frame is handled using the logic as described in C.3.3 (see Figure C-1).

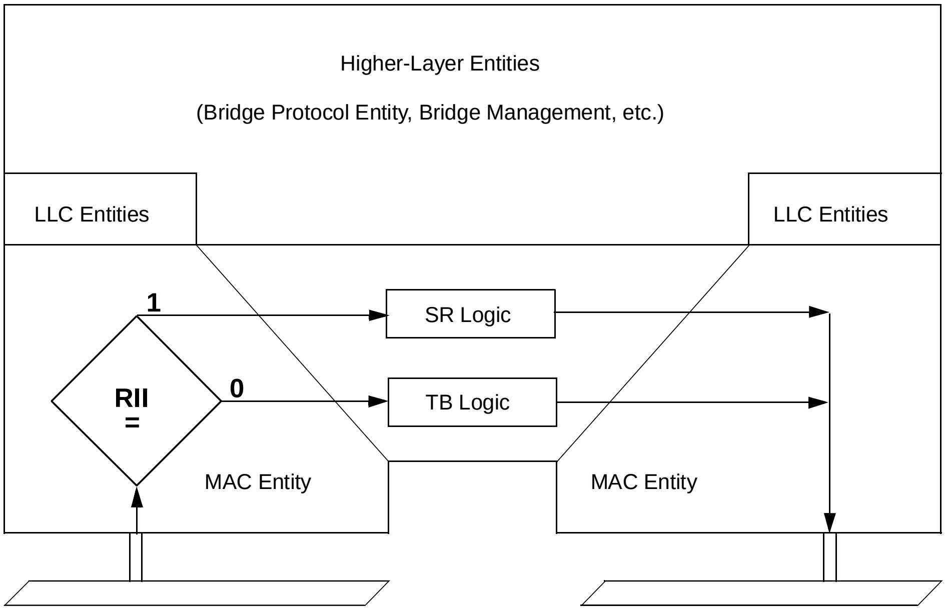


Figure C-1—SRT Bridge operation logic

In SRT bridging, the port serves a specific LAN, and that LAN is assigned a unique LAN_ID. Because SRT allows multiple Bridge paths between two LANs, a Bridge Number (BN) is assigned to each Bridge path such that any two Bridge paths between the same two LANs must have a different Bridge number. From this assignment, each path between LANs can be uniquely identified by the LAN_IDs and Bridge number. A path through the Bridge is designated in the RI field by the designation of LIN, BN, LOUT where LIN is the LAN_ID of the receiving Port and LOUT is the LAN_ID of the forwarding Port.

A bridging relationship (Bridge path) therefore exists between each unique pair of Ports in a Bridge. The term Bridge Port Pair (BPP) will be used to represent this pairing of Ports forming a Bridge path, and each BPP will be designated by both Port_Numbers and their respective BN. The Bridge path can now be specified by the set of LAN_IDs and BN assigned to that pair of Ports.

Each Port has the following attributes:

a) Port_Number
b) LAN_ID
c) Largest MSDU

d) SRT port type
e) RD Limits

With this Bridge architecture approach, it is sufficient to describe the Bridge frame handling by describing it on a Bridge Port Pair basis as it is done in C.3.7.

C.3.3 Bridge operation

The use, by the MAC relay entity, of the Internal Sublayer Service provided by the individual MAC entities associated with each Bridge Port is specified in C.3.5 and C.3.6. Bridge operations on frames without routing information (RII=0) are described in 7.3. The Bridge Protocol Entity remains unchanged. The Bridge Management Entity is enhanced to include SR managed objects as described in C.4. The MAC Relay Entity is enhanced to handle source-routed frames as well as non-source-routed frames. The source-routing forwarding process forwards received source-routed frames to other Bridge Ports on the basis of information contained in the frame and on the state of the Bridge Ports (C.3.7). Source addresses of source-routed frames (with RII=1) are not processed by the Transparent Bridge function. This is because it is not possible to associate a spanning tree Port with an address on a source-routed frame since source-routing paths do not always coincide with spanning tree paths.

C.3.3.1 Source-routing function overview

For a source-routed frame, a forwarding decision is based on information contained in a routing information field in the body of the frame, if such a field is present. If it is not present, the forwarding decision is based on the destination address field of the frame (by the Transparent Bridging logic).

After a route has been determined, the station that originates a frame designates the route that the frame will travel by embedding a description of the route in the routing information (RI) field of the transmitted frame. Bridges copy and forward these frames from one LAN to another without altering the frame’s content provided that the MAC types are the same. If the MAC types differ, then the frame’s content will be altered and the FCS will be regenerated. If data is sent in a frame in which the routing information is modified (e.g., an All Routes Explorer frame), the FCS will be recalculated and the user data integrity may be compromised. Source routing provides the capability for frames containing user data to traverse like LANs without regeneration of the FCS.

Stations initially obtain the route to a given destination station by a process called *route discovery*. This process allows the station to dynamically discover a route to the destination station, as needed. Each Bridge that forwards a route discovery (explorer) frame indicates in the RI field the Bridge’s configuration (LIN, BN, LOUT) and the largest frame size the Bridge can support (through the particular LOUT), if it is smaller than the size already specified, and regenerates the FCS. In this way, the routing information is built by the Bridges as the explorer frame is forwarded from LAN to LAN. When the frame reaches the destination station, the RI field indicates the route the frame traveled from source to destination and the maximum frame size supported along the entire route.

Figure C-2 depicts the elements of the source-routing function.

C.3.3.2 Source-routing information field

The structure of the routing information field is defined in Figure C-3.

The RI field is transmitted on the medium according to the usual practice for each MAC’s treatment of data. For temporal understanding of the RI field, Figure C-4 is equivalent to the definition in Figure C-3.

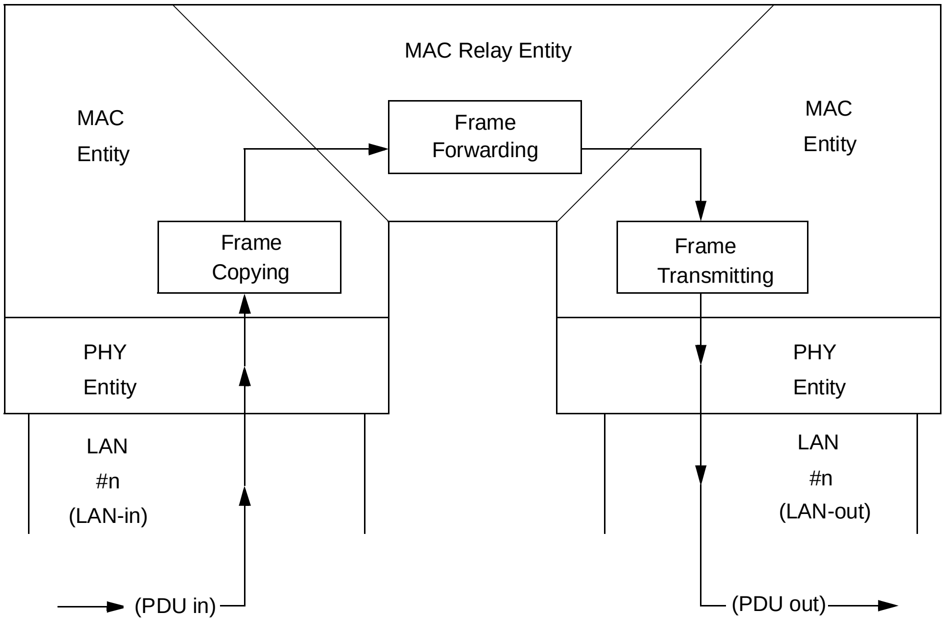
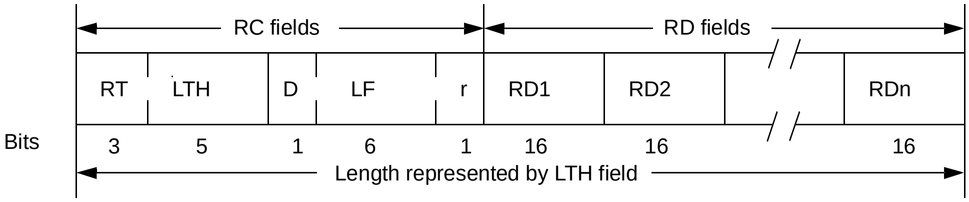


Figure C-2—Elements of a Source-Routed Bridge



RC = Routing Control (16 bits)
RDn = Route Descriptor n (16 bits)
RT = Routing Type (3 bits)
r = reserved (1 bit)
LTH = Length (5 bits)
D = Direction bit (1 bit)
LF = Largest Frame (6 bits)

Figure C-3—Structure of the routing information field

Routing Type (RT) field. This field indicates whether the frame is to be forwarded through the network either along a single route, or multiple routes through multiple interconnected LANs.

Specifically Routed frame (RT=0XX). If the most significant RT bit is set to 0, the RD fields contain a specific route through the network. The XX bits are preserved in transmission throughout the network.

All Routes explorer frame (RT=10X). If the RT bits are set to 10X, indicating an All Routes Explorer frame, the frame will be routed along every route in the network allowed by the ARE forwarding decision. This type will result in as many frames arriving at the destination station, as there are different routes from the source to that station. It is originated by an end station with no route descriptors. Route descriptors are added to the frame by SRT Bridges as they forward the frame. The X bit is preserved in transmission throughout the Bridged Local Area Network.

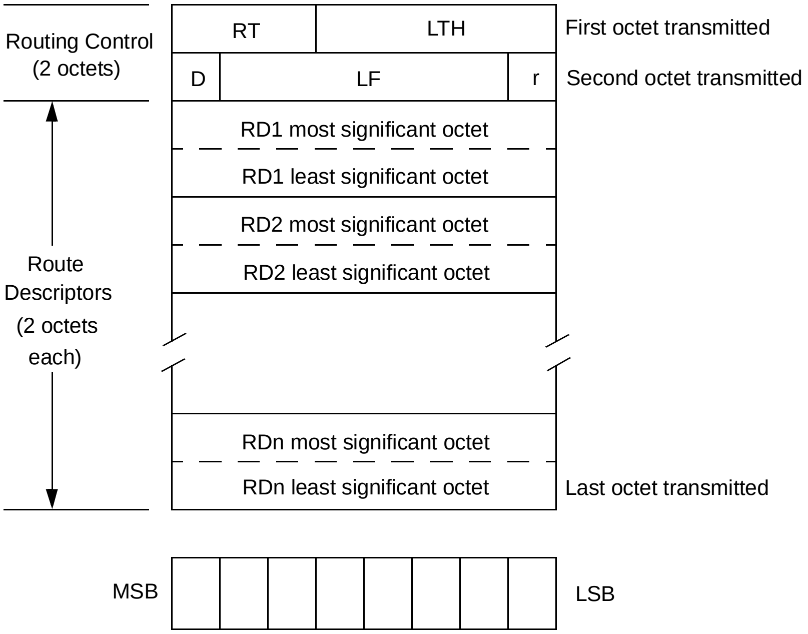

Routing Control
(2 octets)
RT
LTH
First octet transmitted
D
LF
r
Second octet transmitted
RD1 most significant octet
RD1 least significant octet
RD2 most significant octet
RD2 least significant octet
Route
Descriptors
(2 octets
each)
RDn most significant octet
RDn least significant octet
Last octet transmitted
MSB
LSB

Figure C-4—Routing information field

Spanning Tree Explorer frame (RT=11X). If the RT bits are set to 11X, indicating a Spanning Tree Explorer frame, only SRT Bridges with Ports in the Transparent Bridging forwarding state relay the frame from one LAN to another with the result that the frame will be forwarded along the spanning tree and appear no more than once on every LAN in the network. It is originated by an end station with no route descriptors. Route descriptors are added to the frame by SRT Bridges as they forward the frame. The X bit is preserved in transmission throughout the Bridged Local Area Network.

Length Bits (LTH). These five bits indicate the length (in octets) of the RI field. Length field values will be even values between 2 and 30 inclusive.

Direction Bit (D). If the D bit is 0, the frame traverses the LANs in the order in which they are specified in the routing information field (RD1 to RD2 to... to RDn). Conversely, if the D bit is set to 1, the frame will traverse the LANs in the reverse order (RDn to RDn-1 to... to RD1). The D bit is meaningful only for SRFs. For STE and ARE frames, the D bit shall be 0.

Largest Frame Bits (LF). The LF bits indicate the largest size of the MAC Service Data Unit (MAC information field) that may be transmitted between two communicating stations on a specific route. The LF bits are meaningful only for STE and ARE frames. For SRFs, the LF bits are ignored and shall not be altered by the Bridge. A station originating an explorer frame sets the LF bits to the maximum frame size it can handle. Forwarding Bridges shall set the LF bits to indicate the largest value that does not exceed the minimum of

a) The indicated value of the received LF bits
b) The largest MSDU size supported by the Bridge
c) The largest MSDU size supported by the Port from which the frame was received
d) The largest MSDU size supported by the Port on which the frame is to be transmitted

The destination station may further reduce the LF value to indicate its maximum frame capacity. Base LF bit encoding values and rationale are given in Figure C-6.

LF bit encodings are composed of a 3-bit base encoding and a 3-bit extended encoding (see Figure C-5). The SRT Bridge shall contain an LF mode indicator to allow for the selection of base or extended LF bits. When the LF mode indicator is set to *base mode*, the Bridge shall set the LF bits in explorer frames in accordance with the largest frame base values (see Table C-1). When the LF mode indicator is set to *extended mode*, the Bridge shall set the LF bits in explorer frames in accordance with the largest frame extended values (Table C-2).

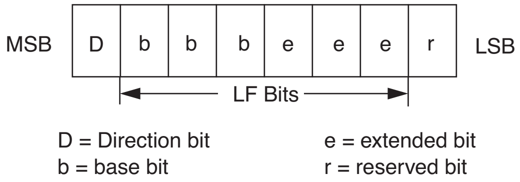


Figure C-5—Base and extended LF bits on the second octet of the RC field

000: 516 octets (ISO 8473, Connectionless Network Protocol, plus LLC)
001: 1 470 octets (IEEE Std 802.3, CSMA/CD LAN)
010: 2 052 octets (80 by 24 character screen with control)
011: 4 399 octets (IEEE Std 802.5, FDDI, 4 Mb/s token ring, ISO 9314-2)
100: 8 130 octets (IEEE Std 802.4 token bus LAN)
101: 11 407 octets (IEEE Std 802.5 4-bit burst errors unprotected)
110: 17 749 octets (IEEE Std 802.5 16 Mb/s token ring LAN)
111: 41 600 octets (Base for extending to 65 535 octets)

Figure C-6—Largest frame base values and rationale

Table C-1—Largest frame base values

Base	Value	Base	Value
000	516 octets	100	8 130 octets
001	1 470 octets	101	11 407 octets
010	2 052 octets	110	17 749 octets
011	4 399 octets	111	>17 749 octets

LF Value = the greatest integer in (blfv + elf x (nlfv – blfv)/8)

where

blfv = Base LF value represented by the low-order 3 bits
nlfv = Next base LF value (the next value after 111 is 65 535)
elf = Extended LF bit encoding (0 through 7)

NOTE—Bridges are not required to support every value in the previous variable list. Also, source-routing end systems on MACs with a particular maximum frame size should not send frames in which the MAC header and trailer, RI fields, and information exceed the maximum frame size for that MAC.

In general, base values in octets were chosen to allow transmission of optimally sized frames through various LAN media access methods. These sizes were calculated by subtracting the maximum size MAC header and the maximum size Routing Information Field (30 octets) from the maximum physical frame size (see Figure C-7). Values associated with the bit encodings represent the maximum length of the MAC information field (see Figure C-7). Extended bits are used to provide seven intermediate values between each set of base values. The equation for calculating these values and the actual calculated values are included in Table C-2.

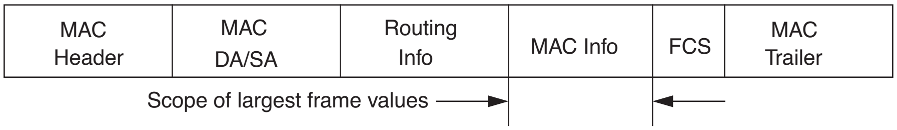


Figure C-7—Scope of the largest frame values

Table C-2—Largest frame extended values

		EXTENSION							
BASE		000	001	010	011	100	101	110	111
	000	516	635	754	873	993	1 112	1 231	1 350
	001	1 470	1 542	1 615	1 688	1 761	1 833	1 906	1 979
	010	2 052	2 345	2 638	2 932	3 225	3 518	3 812	4 105
	011	4 399	4 865	5 331	5 798	6 264	6 730	7 197	7 663
	100	8 130	8 539	8 949	9 358	9 768	10 178	10 587	10 997
	101	11 407	12 199	12 992	13 785	14 578	15 370	16 163	16 956
	110	17 749	20 730	23 711	26 693	29 674	32 655	35 637	38 618
	111	41 600	44 591	47 583	50 575	53 567	56 559	59 551	>59 551

Route Descriptor (RD) field. This field is depicted in Figure C-8.

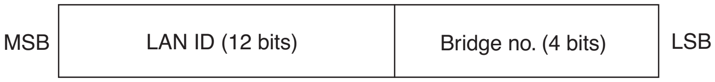


Figure C-8—Route descriptor field

The sequence of RD fields defines a specific route through the network. Each RD contains a network-unique 12-bit LAN ID plus a 4-bit Bridge number that is used to differentiate between two or more Bridges when they connect the same two LANs (parallel Bridges). The last Bridge number in the routing information field is reserved and set to zero.

Reserved (r) Bits. All reserved (r) bits are ignored upon receipt and transmitted as received.

C.3.3.3 Source-routing frame types

Frames are processed by Bridges based on the routing type (RT) of the frame. The RT of the frame to be processed is indicated in the RT subfield of the RI field of the frame. C.3.3.3.1 through C.3.3.3.3 define each of the routing types.

C.3.3.3.1 Specifically routed frame type (RT=0XX)

The RD fields contain a specific route through the network. This type is used for the normal routing of data frames from source to destination.

C.3.3.3.2 All Routes Explorer frame type (RT=10X)

This frame is routed along all non-repeating routes in the network. This type, known as an All Routes Explorer frame, will result in as many frames arriving at the destination station as there are allowable routes to that station. The Bridge adds routing information to the frame as it forwards it.

C.3.3.3.3 Spanning Tree Explorer frame type (RT=11X)

Spanning Tree Explorer frames are forwarded (by the source-routing logic) through only those Bridge Ports in the forwarding state, (i.e., along the spanning tree). The result is that the frame will appear only once at the destination station. Routing descriptors are added to this frame by the Bridges, and filtering on the destination address will not be performed. These frames traverse the adjacent portion of the spanning tree that is connected by SRT Bridges, acquiring the route as they are forwarded.

C.3.3.4 Bridge processing of source-routed frames

Processing by the Bridge is based on a two-stage procedure: frame reception and frame forwarding. Explicit function definitions for these are included in C.3.5 and C.3.7, respectively. The process of frame reception refers to the indication of a frame to the MAC Relay Entity from the MAC entity. The process of frame forwarding refers to the internal checking of frames that have been copied, the possible addition of routing information, and the actions that are performed to forward frames to another LAN.

C.3.4 Port state information

Port states are described in 7.4.

C.3.5 Frame reception

If a frame has no routing information (RII=0), then the Bridge shall process the frame as described in Clause 7.

If a frame has routing information (RII=1), then the frame shall be processed as indicated in C.3.7. The user_priority of such frames shall be regenerated as specified in 7.5.1.

C.3.6 Frame transmission

Frames are transmitted as described in 7.6.

C.3.7 Frame forwarding

The decision to forward a frame is based on the content of the RI field and the internal state of the Bridge. Frames of a given user_priority and frame_type shall be forwarded in the order that they were copied from the LAN. The forwarding process forwards received frames that are to be relayed to other Bridge Ports. Non-source-routed frames (RII=0) shall be forwarded as specified in 7.7. The processing of source-routed frames (RII=1) is included in the subclauses that follow. The following subclauses describe the actions of the SRT Bridge for a single enabled Bridge Port pair (LIN and LOUT). Multiport Bridges include multiple

Bridge Port pairs and shall behave in the manner described in this clause for each Port pair. No frames shall be forwarded through disabled BPPs.

C.3.7.1 Specifically routed data frames (RT=0XX)

Bridges that find an RI field-to-bridge configuration (LIN, BN, LOUT combination) match shall forward such frames without alteration of the RI provided that both MAC entities use the same frame formats (e.g., the Bridge is from Token Ring to Token Ring, IEEE Std 802.5). If multiple occurrences of LOUT are in the routing information field of a frame, the Bridge shall discard the frame and, if implemented, increment the DupLout counter. Note that SRFs are forwarded or discarded based on the state tables, regardless of whether the destination is to an individual or group address. Table C-3 is explained as follows:

(SRF1). If a specifically routed frame is received by an SRT Bridge and a LIN-BN-LOUT match is not found in the routing information field, then the frame shall not be forwarded on the Port indicated by LOUT.

(SRF2). If a specifically routed frame is received by an SRT Bridge and the LIN-BN-LOUT matches, and there are no multiple occurrences of LOUT in the routing information field, and the Length field is even, the frame shall be forwarded and, if implemented, the SRFsForwarded counter shall be incremented.

(SRF3). If a specifically routed frame is received by an SRT Bridge and the LIN-BN-LOUT matches but the Length field is zero or an odd number, the frame shall be discarded and, if implemented, the Invalid RI counter shall be incremented.

(SRF4). If a specifically routed frame is received by an SRT Bridge and the LIN-BN-LOUT matches but there are multiple occurrences of LOUT in the routing information field, the frame shall be discarded and, if implemented, the DupLout counter shall be incremented.

Table C-3—Specifically routed frame forwarding state table

Ref.	Condition	Action
SRF1	LIN-BN-LOUT does not match	Do not forward frame
SRF2	(LIN-BN-LOUT match) & (occurrences of LOUT = 1) & (RI_LTH = even number)	Forward Frame on LOUT Increment (SRFsForwarded)
SRF3	(LIN-BN-LOUT match) & ((RI_LTH = odd number) | (RI_LTH = 0))	Discard Frame Increment (InvalidRi)
SRF4	(LIN-BN-LOUT match) & (occurrences of LOUT > 1)	Discard Frame Increment (DupLout)

In all cases above: RII = 1, RT = 0XX, and BPP State = enabled

Definitions: Discard = Do not forward on any LAN-out
Forward = Transmit the frame on LAN-out

C.3.7.2 All Routes Explorer frames (RT=10X)

In order to limit unnecessary traffic, SRT Bridges are allowed to filter ARE frames that are superfluous or redundant. Multiport SRT Bridges may discard (filter) ARE frames that

a) Have been through the Bridge before. This can be determined by looking for a LIN-BN-LOUT combination in the RI field of the received frame.
b) Have been through any LAN attached to the Bridge before. This condition is more restrictive than the condition described in item a). This can be determined by examining the RI field of the received frame for a LAN ID match with any of the LANs attached to the Bridge other than the LIN.

All SRT Bridges shall forward all other frames on each LOUT except when any of the following conditions hold:

1) LOUT is already present in the RI field.
2) The last LAN ID in the RI field does not match LIN.
3) The number of route descriptors meets or exceeds the ARERdLimit.
4) The RI field length is an odd number.
5) The RI field length is zero.
6) The RI field length is four (optionally—see Table C-4).
7) The Direction bit is not 0.

If any of these conditions hold, the frame shall be discarded. If condition 2) occurs, the LanIdMismatch counter is incremented. If conditions 4), 5), or 7) occur, the Invalid RI counter is incremented.

If none of these conditions holds then,

- If the LTH = 2, the Bridge adds an RD with LIN to the RI field and,
- The Bridge adds its Bridge number and the number of LOUT to the frame.

It also increments the length field by two for every route descriptor it adds. Before forwarding the frame, the Bridge adjusts the Largest-Frame (LF) field to reflect the transfer capacity of the Bridge if the transfer capacity of the Bridge Port pair is less than the size indicated by the received LF. The transfer capacity of the Bridge is the least of

- The maximum frame size of LIN,
- The maximum frame size of LOUT, and
- The maximum frame size that can be handled by the Bridge itself.

Finally, the FCS is recomputed. ARE frames are processed as indicated in Table C-4.

Table C-4 is explained as follows:

(ARE1). If an All Routes Explorer frame is received by an SRT Bridge with the LAN-out ID already in the RI field or if the Bridge has filtered the frame, the frame shall not be forwarded on the LAN-out indicated by the LAN-out ID.

(ARE2). If an All Routes Explorer frame is received by an SRT Bridge with the last LAN ID not matching LIN, the frame shall be discarded and, if implemented, the LanIdMismatch counter shall be incremented.

(ARE3). If an All Routes Explorer frame is received by an SRT Bridge with the number of RDs meeting or exceeding AreRdLimit of the Port indicated by LOUT, the frame shall not be forwarded and, if implemented, the AreRdLmtExceeded counter of the Port indicated by LOUT shall be incremented.

(ARE4). If an All Routes Explorer frame is received by an SRT Bridge with a Length field of zero or an odd number, or the Direction bit is not 0, the frame shall be discarded and, if implemented, the InvalidRi counter shall be incremented.

(ARE5). If an SRT Bridge receives a valid All Routes Explorer with a Length field equal to 2, and the AreRdLimit is greater than 1, then the Bridge shall modify the frame being forwarded by adding its LAN-in ID (LIN), its Bridge number (BN), its LAN-out ID (LOUT), and 4 bits as 0, setting the LTH field to 6, setting the LF field of the RI to the minimum of the received LF and the transfer capacity of the Bridge, and recalculating the FCS, and then queueing the frame for transmission on its LAN-out and, if implemented, the AREFramesForwarded counter shall be incremented.

Table C-4—All Routes Explorer frame forwarding state table

Ref.	Condition	Action
ARE1	(LOUT already in RI) | (bridge has filtered)	Do not forward on LOUT
ARE2	Last LAN ID in RI ¬ = LIN	Discard Frame Increment(LanIdMismatch)
ARE3	# RDs >= AreRdLimit	Do not forward Increment(ARERdLmtExceeded)
ARE4	(RI_LTH = odd number) | (RI_LTH = 0) | (D ¬ = 0)	Discard Frame Increment(InvalidRi)

ARE5	(RI_LTH = 2) & (bridge has not filtered) &
	(AreRdLimit > 1)
		Add LIN,BN,LOUT to RI field
		SET RI_LTH = 6
		SET LF=MIN(LF, LIN, BR, LOUT)
		Recalculate FCS
		Forward frame
		Increment(AREFramesForwarded)

ARE6	(RI_LTH = 4) & (bridge has not filtered) & (Last LAN ID in RI = LIN) & (AreRdLimit > 1)	Discard frame Increment(InvalidRi) OR (Optionally) Add BN,LOUT to RI field RI_LTH = RI_LTH + 2 SET LF = MIN(LF, LIN, BR, LOUT) Recalculate FCS Forward frame Increment(AREFramesFowarded)

ARE7	(RI_LTH = even number between 6 and 28 inclusive) &
	(LOUT not already in RI) &
	(bridge has not filtered) &
	(Last LAN ID in RI = LIN) &
	(# RDs < AreRdLimit)
		Add BN,LOUT to RI field
		RI_LTH = RI_LTH + 2
		SET LF = MIN(LF, LIN, BR, LOUT
		Recalculate FCS
		Forward frame
		Increment(AREFramesFowarded)

In all cases above: RII = 1, RT = 10X, and BPP State = enabled

Definitions: Discard = Do not forward on any LAN-out
Forward = Transmit the frame on LAN-out

(ARE6). If an SRT Bridge receives a valid All Routes Explorer frame with RI length = 4, and the AreRdLimit is greater than 1, then it shall either

Discard the frame and, if implemented, increment the InvalidRi counter; or

Modify the frame being forwarded by setting the last 4 bits of the previous RD to its Bridge number (BN), adding the next two-octet RD consisting of the LAN-out (LOUT) and remaining 4 bits as 0, incrementing the Length by 2, setting the largest frame field to the minimum of the received LF and the transfer capacity of the Bridge, recalculating the FCS, queueing the frame for transmission on its LAN-out, and, if implemented, increment the AREFramesForwarded counter.

(ARE7). If an SRT Bridge receives a valid All Route Explorer with a length field of 6 to 28 (inclusive, even numbers only), and the number of RDs less than the AreRdLimit, then the Bridge shall modify the frame being forwarded by setting the last 4 bits of the previous RD to its Bridge number (BN), adding the next two-octet RD consisting of the LAN-out (LOUT) and remaining 4 bits as 0, incrementing the length by 2, setting the largest frame field to the minimum of the received LF and the transfer capacity of the Bridge, recalculating the FCS, queueing the frame for transmission on its LAN-out and, if implemented, increment the AREFramesForwarded counter.

C.3.7.3 Spanning Tree Explorer (STE) frames

The conditions for forwarding STE frames are identical to those for ARE frames with the following exceptions:

a) No explorer filtering is necessary because of the spanning tree.
b) Forwarding is also conditional on the state of the LAN-out port and the state of the spanning tree.

Filtering on the destination address of STE frames shall not be performed. These frames shall traverse the entire spanning tree. STE frames are processed as indicated in Table C-5.

Table C-5—Spanning tree explorer frame forwarding state table

Ref.	PortState		Condition	Action
	LIN	LOUT		
STE1	F	F	(LOUT already in RI)	Discard Frame, Increment(DupLanIdOrTreeError)
STE2	F	—	Last LAN ID in RI ¬ = LIN	Discard Frame Increment(LanIdMismatch)
STE3	F	F	# RDs >= SteRdLimit	Discard Frame Increment Increment(SteRdLmtExceeded)
STE4	F	—	(D ¬ = 0) | (RI_LTH = 0) | (RI_LTH=odd number)	Do not forward Increment(InvalidRi)

STE5	F	F	(RI_LTH = 2) &
			(SteRdLimit > 1)
				Add LIN,BN,LOUT to RI field
				SET RI_LTH = 6
				SET LF = MIN_(LF, LIN, BR, LOUT)
				Recalculate FCS
				Forward frame
				Increment(STEFramesForwarded)

STE6	F	F	(RI_LTH = 4) & (Last LAN ID in RI = LIN) & (SteRdLimit > 1)	Discard frame Increment (InvalidRI) OR (Optionally) Add BN,LOUT to RI field RI_LTH = RI_LTH + 2 SET LF = MIN(LF, LIN, BR, LOUT) Recalculate FCS Forward frame Increment(STEFramesFowarded)

STE7	F	F	(RI_LTH=even number between 6 and 28 inclusive) &
			(LOUT not already in RI) & (Last LAN ID in RI = LIN) &
			(# RDs < SteRdLimit)
				Add BN,LOUT to RI field
				RI_LTH = RI_LTH + 2
				SET LF = MIN(LF, LIN, BR, LOUT)
				Recalculate FCS
				Forward frame
				Increment(STEFramesFowarded)

STE8	¬F	—		Do not forward on LOUT
STE9	F	¬F		Do not forward on LOUT

In all cases above: RII = 1, RT = 11X, and BPP State = enabled
Definitions:
Discard= Do not forward on any LAN-out
Forward= Transmit the frame on LAN-out
F= Forwarding Port State
¬F= Not Forwarding Port State
—= Any Port State

Table C-5 is explained as follows:

(STE1). If a Spanning Tree Explorer frame is received by an SRT Bridge and the LAN-in Port and LAN-out Port are in forwarding state and LOUT is already in the RI field, then the frame shall be discarded and, if implemented, the Bridge shall increment the DuplicateLanIdOrTreeError counter.

(STE2). If a Spanning Tree Explorer frame is received by an SRT Bridge, the LAN-in Port is in forwarding state, and the last LAN ID does not match LIN, the frame shall be discarded and, if implemented, the LanIdMismatch counter shall be incremented.

(STE3). If a Spanning Tree Explorer frame is received by an SRT Bridge, the LAN-in Port and LAN-out Port are in forwarding state and the number of RDs meets or exceeds the SteRdLimit of the Port indicated by LOUT, the frame shall not be forwarded on LOUT and, if implemented, the SteRdLimit Exceeded counter of the Port indicated by LOUT, shall be incremented.

(STE4). If a Spanning Tree Explorer frame is received by an SRT Bridge, the LAN-in Port is in forwarding state, and the length field is zero or an odd number or the Direction bit is not 0, the frame shall be discarded and, if implemented, the InvalidRi counter shall be incremented.

(STE5). If an SRT Bridge with LAN-in and LAN-out Ports in forwarding state receives a valid Spanning Tree Explorer with a length field equal to 2, and the SteRdLimit is greater than 1, then the SRT Bridge shall modify the frame being forwarded by adding its LAN-in ID (LIN), its Bridge number (BN), its LAN-out ID (LOUT), and 4 bits as 0. The SRT Bridge shall also set the length field to 6, set the largest-frame field of the RI to the minimum of the received LF and the transfer capacity of the Bridge, and recalculate the FCS. The SRT Bridge shall then queue the frame for transmission on its LAN-out and, if implemented, increment the STEFramesForwarded counter.

(STE6). If an SRT Bridge with LAN-in and LAN-out ports in forwarding state receives a valid Spanning Tree Explorer with a length field of 4, and the SteRdLimit is greater than 1, then the Bridge shall either

a) Discard the frame and, if implemented, increment the InvalidRi counter; or
b) Modify the frame being forwarded by setting the last 4 bits of the previous RD to its Bridge number (BN), adding the next two-octet RD consisting of the LAN-out (LOUT) and remaining 4 bits as 0. The SRT Bridge shall also increment the length by 2, set the largest frame field to the minimum of the received LF and the transfer capacity of the Bridge, and recalculate the FCS. The SRT Bridge shall then queue the frame for transmission on its LAN-out and, if implemented, increment the STEFramesForwarded counter.

(STE7). If an SRT Bridge with LAN-in and LAN-out Ports in forwarding state receives a valid Spanning Tree Explorer with a length field of 6 to 28 (inclusive, even numbers only), and the number of RDs is less than the SteRdLimit, then the SRT Bridge shall modify the frame being forwarded by setting the last 4 bits of the previous RD to its Bridge number (BN), adding the next two-octet RD consisting of the LAN-out (LOUT) and remaining 4 bits as 0. The SRT Bridge shall also increment the length by 2, set the largest frame field to the minimum of the received LF and the transfer capacity of the Bridge, and recalculate the FCS. The SRT Bridge shall then queue the frame for transmission on its LAN-out and, if implemented, increment the STEFramesForwarded counter.

(STE8). SRT Bridges shall only forward Spanning Tree Explorer frames received on a Port if the Port is in forwarding state.

(STE9). SRT Bridges shall only forward Spanning Tree Explorer frames on a Port if the Port is in forwarding state.

C.3.7.4 Duplicate Bridge number test

Bridge management may check for parallel Bridges assigned the same number as its Bridge number by using the method below.

For every two LANs the Bridge interconnects, Bridge management initiates transmission of an LLC TEST PDU with the destination address set to the MAC Address of the Port connected to one LAN and the source address set to the MAC Address of the Port connected to the other LAN. The TEST PDU should contain a route consisting of the LAN-in relative to the source address, the Bridge number, and the LAN-out relative to the source address. If Bridge management receives more than one response back from this TEST, a parallel Bridge exists that is assigned the same Bridge number and network management should be notified. See Figure C-9.

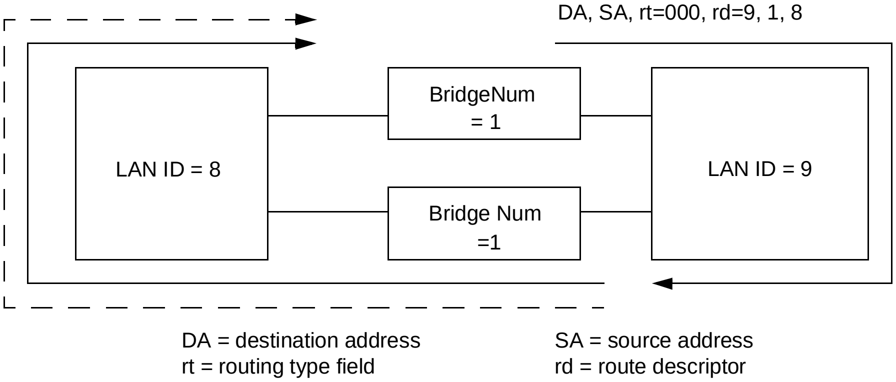


Figure C-9—Duplicate bridge number test illustration

C.3.7.5 Queued frames

The forwarding process provides storage for queued frames, awaiting an opportunity to submit these for transmission to the individual MAC entities associated with each Bridge Port. The order of frames received on the same Bridge Port shall be preserved for the following:

a) Unicast frames with a given user_priority for a given combination of destination_address and source_address.
b) Multicast frames with a given user_priority for a given destination_address.

The Forwarding Process may provide more than one transmission queue for a given Bridge Port. Frames are assigned to storage queue(s) on the basis of their user_priority using a traffic class table that is part of the state information associated with each Port. The table indicates, for each possible value of user_priority, the corresponding value of traffic class that shall be assigned. Priority value 0 represents the lowest value of user_priority, and 7 the highest value. Queues correspond one-to-one with traffic classes.

For management purposes, up to eight traffic classes are supported by the traffic class tables in order to allow for separate queues for each level of user_priority. Traffic classes are numbered 0 through N-1, where N is the number of traffic classes associated with a given outbound Port. Management of traffic class information is optional. Traffic class 0 corresponds to non-expedited traffic; non-zero traffic classes are expedited classes of traffic.

NOTE—In a given Bridge, it is permissible to implement different numbers of traffic classes for each Port. Ports associated with media access methods that support a single transmission priority, such as CSMA/CD, can support more than one traffic class.

Where the Forwarding Process does not support expedited classes of traffic for a given Port, in other words, where there is a single traffic class associated with the Port, all values of user_priority map to traffic class 0. In Bridges that support expedited traffic, the default mapping of user_priority to traffic class, for the number of traffic classes implemented, is as shown in Table 7-2.

A frame queued by the forwarding process for transmission on a Port shall be removed from that queue on submission to the individual MAC entity for that Port; no further attempt shall be made to transmit the frame on that Port even if the transmission is known to have failed.

A frame queued by the forwarding process for transmission on a Port can be removed from that queue, and not subsequently transmitted, if the time for which buffering is guaranteed has been exceeded for that frame.

A frame queued for transmission on a Port shall be removed from that queue, and not subsequently submitted to the individual MAC entity for that Port, if that is necessary to ensure that the maximum Bridge transit delay (see 6.3.6) will not be exceeded at the time at which the frame would be subsequently transmitted.

An STE frame queued for transmission on a Port shall be removed from that queue if the associated Port leaves the forwarding state.

Removal of a frame from a queue for transmission on any particular Port does not of itself imply that it shall be removed from a queue for transmission on any other Port.

C.3.7.6 Selecting frames for transmission

Frames shall be selected for transmission in accordance with the provisions of 7.7.4.

C.3.7.7 Priority mapping

The source-routing forwarding process determines the value of the user_priority and access_priority parameters used to relay frames according to 7.7.5.

C.3.8 Addressing

In a source-routing Bridged Local Area Network, each LAN and each Bridge shall be assigned a number as indicated in C.3.8.1 through C.3.8.3.

C.3.8.1 LAN ID

Each individual LAN in a multiple-LAN network shall be assigned a unique 12-bit non-zero LAN ID that is consistently known by all Bridges connecting to the given LAN.

C.3.8.2 Bridge number

Each Bridge shall have a Bridge number (BN). The route descriptor sequence (LIN-BN-LOUT) shall be unique for each path between two LANs connected by a Bridge.

C.3.8.3 Route descriptor

The route descriptor is a two-octet field (16 bits). The first 12 bits represent the LAN ID and the 4 bits following represent the Bridge number (see C.3.3.2 for frame format details). A sequence of route descriptors defines a unique route through the Bridged Local Area Network. Because the end of a route is a LAN and not a Bridge, the individual Bridge portion of the last route descriptor in the routing information field is reserved and shall be set to zero. C.3.7 describes the method for creating, building, and forwarding frames with source routes.

C.4 Bridge management

The enhancements in the following subclauses are made to the management facilities defined in Clause 14. These include the ability to control the source-routing paths through the Bridged Local Area Network.

C.4.1 Bridge management entity

The following enhancements are made to the Bridge Management Entities defined in 14.4.

C.4.1.1 Bridge configuration

C.4.1.1.1 Discover bridge

To enable the Discover Bridge management operations defined in 14.4.1.1 to report SRT capability in the Bridge, the parameter that follows is added to the outputs of the operation.

C.4.1.1.1.1 Purpose

To indicate the SRT transfer capability of the Bridge.

C.4.1.1.1.2 Additional inputs

None.

C.4.1.1.1.3 Additional outputs (14.4.1.1.3)

— SRT Transfer Capacity—a value field that indicates the largest MSDU field of a frame that can be handled by the SRT Bridge MAC relay entity. If SRT is not supported in the Bridge, this value shall be reported as zero.

C.4.1.1.2 Read bridge

To enable the Read Bridge management operations defined in 14.4.1.2 to report SRT capability in the Bridge, the parameter that follows is added to the outputs of the operation.

C.4.1.1.2.1 Purpose

To indicate the SRT transfer capability of the Bridge.

C.4.1.1.2.2 Additional inputs

None.

C.4.1.1.2.3 Additional outputs (14.4.1.2.3)

— SRT Transfer Capacity—a value field that indicates the largest MSDU field of a frame that can be handled by the SRT Bridge MAC relay entity. If SRT is not supported in the Bridge, this value shall be reported as zero.

C.4.2 Forwarding process

The following enhancements are made to the Forwarding Process as defined in 14.6.

C.4.2.1 The port counters

See 14.6.1.

C.4.2.1.1 Read forwarding port counters

C.4.2.1.1.1 Purpose

The Port Counter objects for the SRT Bridge consist of the counters defined in 14.6.1.1, along with the SRT-specific counters in the following subclauses.

C.4.2.1.1.2 Additional inputs

None.

C.4.2.1.1.3 Additional outputs

The following port counters are added to the list of outputs defined by 14.6.1.1.3. If the counter is not supported, the value shall be reported as zero.

a) InvalidRI—count of frames that were discarded due to a formatting error (i.e., an odd RI length, or a 0 RI length).
b) DupLout—count of frames that were discarded due to a duplicate LOUT on SRFs.
c) LanIdMismatch—count of ARE and STE frames that were discarded because the last LAN ID in the routing information field did not equal the LAN-in ID.
d) DupLanIdOrTreeError—count of STE frames that were discarded because the LAN-out ID already existed in the routing information field.
e) STERDlimitExceeded—count of STE frames discarded due to STERD Limit exceeded.
f) ARERDlimitExceeded—count of ARE frames discarded due to ARERD Limit exceeded.
g) SRFs Forwarded—inbound count of SRFs forwarded from this Port to another Port on the Bridge.
h) STEFramesForwarded—inbound count of STE frames forwarded from this Port to another Port on the Bridge.
i) AREFramesForwarded—inbound count of ARE frames forwarded from this Port to another Port on the Bridge.

C.4.3 SRT Bridge management entity

C.4.3.1 SRT Bridge configuration

The SRT Bridge Configuration operations for the SRT Bridge Management Entity allow management to Read the Bridge RD Limits, Set the Bridge RD Limits, Read the Bridge LF Mode, and Set the Bridge LF Mode.

C.4.3.1.1 Read LF Mode

C.4.3.1.1.1 Purpose

To report the mode that the Bridge uses to set LF bits in ARE and STE frames.

C.4.3.1.1.2 Inputs

None.

C.4.3.1.1.3 Outputs

a) LF Mode—a value field that indicates whether the Bridge uses base mode or extended mode to set LF bits.

C.4.3.1.2 Set LF Mode

C.4.3.1.2.1 Purpose

To set the mode that the Bridge will use to set the LF bits in ARE and STE frames.

C.4.3.1.2.2 Inputs

a) LF Mode—a value field that indicates whether the Bridge will use the base or extended mode for LF encodings.

C.4.3.1.2.3 Outputs

None.

C.4.3.2 SRT Port configuration

The SRT Port Configuration operations for the SRT Bridge Management Entity allow management to Read and Set the SRT attributes for each Port (LAN ID and Largest MSDU Size). Comparable to the operations defined in 14.8.2, the operations in the following subclauses can be performed on an SRT Port.

C.4.3.2.1 Read port parameters

C.4.3.2.1.1 Purpose

To obtain information regarding a specific Port within the SRT Bridge Protocol Entity.

C.4.3.2.1.2 Inputs

a) Port number

C.4.3.2.1.3 Outputs

a) SRT Port type—a value field that indicates the port type available. Types include TB and SRT.
b) LAN ID—a value field that indicates the LAN ID corresponding to the LAN to which the Port is attached.
c) Largest MSDU Size—a value field that indicates the largest MSDU of a frame that can be handled by this Port.
d) ARE RD Limit—a value field that indicates the maximum number of route descriptors (RDs) that an ARE frame forwarded by the Bridge is allowed to contain.
e) STE RD Limit—a value field that indicates the maximum number of route descriptors (RDs) that an STE frame forwarded by the Bridge is allowed to contain.

C.4.3.2.2 Set LAN ID

C.4.3.2.2.1 Purpose

To associate a LAN ID with a specific Bridge Port.

C.4.3.2.2.2 Inputs

a) Port Number—the number of the Bridge Port.
b) Lan ID—the Lan ID corresponding to the LAN into which the Port is inserted made up of 12 bits.

C.4.3.2.2.3 Outputs

None.

C.4.3.2.3 Set Largest MSDU Size

C.4.3.2.3.1 Purpose

To associate a largest MSDU size with a Bridge Port.

C.4.3.2.3.2 Inputs

a) Port Number—the number of the Bridge Port.
b) Largest MSDU Size—a value field that specifies the maximum size MSDU that can be handled by this Port.

C.4.3.2.3.3 Outputs

None.

C.4.3.2.4 Set RD limit

C.4.3.2.4.1 Purpose

To associate a route descriptor limit, readable by the Read RD Limits operation, with ARE and STE frames forwarded by the Bridge.

C.4.3.2.4.2 Inputs

a) Port number—the number of the Bridge Port.
b) RD Limit Type—a value field that indicates whether the limit input is to be associated with ARE or STE frames.
c) RD Limit Value—a value field that specifies the maximum number of RDs that may be contained in the RI field of the frame type specified by the RD Limit Type. This field takes values from 0 to 14.

C.4.3.2.4.3 Outputs

None.

C.4.4 SRT Bridge port pair database

The operations that follow can be performed on an SRT Bridge.

C.4.4.1 SRT bridge port pair configuration

C.4.4.1.1 Read SRT bridge port pair database size

C.4.4.1.1.1 Purpose

To read the size of the Bridge Port Pair Database.

C.4.4.1.1.2 Inputs

None.

C.4.4.1.1.3 Outputs

a) Database Size—a value field that indicates the number of entries in the Bridge Port Pair Database.

C.4.4.1.2 Read SRT bridge port pair database entry

C.4.4.1.2.1 Purpose

To read the attributes of an SRT Bridge Port pair database entity.

C.4.4.1.2.2 Inputs

a) Low Port number
b) High Port number

C.4.4.1.2.3 Outputs

a) Source-routing Bridge number
b) Bridge state (enabled or disabled)

C.4.4.1.3 Set SRT Bridge Port Pair Database Entry

C.4.4.1.3.1 Purpose

To set the attributes of an SRT Port pair database entry.

C.4.4.1.3.2 Inputs

a) Low Port number
b) High Port number
c) Source-routing Bridge number
d) Bridge state (enabled or disabled)

C.4.4.1.3.3 Outputs

None.

C.5 Management protocol

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

Annex D

(normative)

PICS Proforma for source-routing transparent bridge operation[11]

D.1 Introduction

The supplier of a protocol implementation which is claimed to conform to Annex C of this standard shall complete the following Protocol Implementation Conformance Statement (PICS) proforma, in addition to the PICS Proforma defined in Annex A (normative). The provisions of A.1 through A.3 apply to the PICS Proforma defined in this annex.

D.2 Relay and filtering of frames

Item	Feature	Status	References	Support
(2a)	Is the order of relayed frames of given user priority preserved?	M	C.2.3.1	Yes [ ]
(2b)	Are STE, ARE, and SRF Frames submitted to a MAC entity for transmission only once?	M	C.2.3.2	Yes [ ]
(2c)	Is the Largest Frame field set by the Bridge as described in C.3.3.2?	M	C.3.3.2	Yes [ ]
(2d)	Are frames received with no routing information (NSR) processed as described in IEEE Std 802.1D?	M	C.3.5	Yes [ ]
(2e)	Are received Specifically Routed Frames (SRF) processed as described in C.3.7.1?	M	C.3.7.1	Yes [ ]
(2f)	Are received All Routes Explorer (ARE) Frames processed as described in C.3.7.2?	M	C.3.7.2	Yes [ ]
(2g)	Does the implementation support option 1 that allows filtering of frames that have been through the Bridge before?	O	C.3.7.2	Yes [ ] No [ ]
(2h)	Does the implementation support option 2 that allows filtering of frames that have been through the Bridge before?	O	C.3.7.2	Yes [ ] No [ ]
(2i)	Are received Spanning Tree Explorer (STE) frames processed as described in C.3.7.3?	M	C.3.7.3	Yes [ ]
(2j)	Does the implementation forward explorer frames (STE, ARE) with LTH = 4 as described in C.3.7.2 and C.3.7.3?	O	C.3.7.2, C.3.7.3	Yes [ ] No [ ]

[11] *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.

D.3 Bridge numbers and LAN IDs

Item	Feature	Status	References	Support
(3a)	Can the LAN IDs of the attached LANs be configured in the Bridge?	M	C.3.8.1	Yes [ ]
(3b)	Can a Bridge Number be assigned to the Bridge?	M	C.3.8.2	Yes [ ]
(3c)	Can Bridge management use the duplicate Bridge number test to check for parallel Bridges that are assigned the same number?	O	C.3.7.4	Yes [ ] No [ ]

D.4 Bridge management

This matter is subject to further ongoing study and resolution.

Item	Feature	Status	References	Support
(4a)	Bridge Management Operations	O	C.4	Yes [ ] No [ ]
(4b)	Read LF Mode.	M	C.4.3.1.1	Yes [ ]
(4c)	Set LF Mode.	M	C.4.3.1.2	Yes [ ]
(4d)	Read Port Parameters.	M	C.4.3.2.1	Yes [ ]
(4e)	Set ARE RD Limit.	M	C.4.3.2.4	Yes [ ]
(4f)	Specify the max value of the ARE RD Limit in this implementation.	M	C.4.3.2.4	A___
(4g)	Set STE RD Limit.	M	C.4.3.2.4	Yes [ ]
(4h)	Specify the max value of the STE RD Limit in this implementation.	M	C.4.3.2.4	A___
(4i)	Set LAN Id.	M	C.4.3.2.2	Yes [ ]
(4j)	Set Largest MSDU Size.	M	C.4.3.2.3	Yes [ ]
(4k)	Read SRT Bridge Port Pair Database Size.	M	C.4.4.1.1	Yes [ ]
(4l)	Read SRT Bridge Port Pair Database Entity.	M	C.4.4.1.2	Yes [ ]
(4m)	Set SRT Bridge Port Pair Database Entity.	M	C.4.4.1.3	Yes [ ]

Annex E

(informative)

Allocation of Object Identifier values

In IEEE Std 802.1D, 1998 Edition this annex contained a summary of all object identifier values that have been allocated by this standard, both in this revision and in previous revisions. Since these management protocol that made use of these definitions (Clause 15 of the 1998 Edition) is no longer specified by this standard and its use deprecated, the body of this annex has been removed.

Annex F

(informative)

Preserving the integrity of FCS fields in MAC Bridges

F.1 Background

When relaying frames between Ports of a MAC Bridge, one of the functions of the Bridge is to regenerate the FCS field in accordance with the MAC procedures that apply to the medium access method through which the frame will be relayed. One of the requirements of the MAC Bridge standard is that any such regeneration of the FCS shall not increase the level of undetected FCS errors that are experienced on the transmitting MAC over that which would be experienced by preserving the FCS (see 6.3.7 and 7.7.6). The point at issue here is only the *undetected* error rate; a conformant Bridge will discard frames that are received with an FCS error; however, there is a finite possibility of undetectable corruption taking place, i.e., the frame is corrupted, but the FCS algorithm does not reveal the error.

This annex looks at the various cases that apply to the operation of the MAC Bridge with respect to FCS regeneration and the alternative approaches that can be taken in order to address the cases. The following cases need to be considered:

a) The source and destination media access methods are identical from the point of view of the operation of the FCS algorithm.

NOTE—This is the case, for example, where the source and destination media access methods are identical. It could also happen with dissimilar media access methods if the FCS coverage, the FCS algorithm, the definitions of the fields covered by the FCS, and the bit/octet ordering are the same in both media access methods.

b) The source and destination media access methods differ from the point of view of the operation of the FCS algorithm.
c) The data covered by the FCS is not modified by the operation of the relay function of the Bridge.
d) The data covered by the FCS is modified by the operation of the relay function of the Bridge.

In each case, it is important to ensure that there is a low probability of additional undetected errors being generated by corruption of the data portion of the frame between removal of the old FCS and computation of the new one, which would result in the transmitted frame carrying invalid data and a valid FCS. Such corruption might occur, for example, as a result of the effect of environmental noise on the operation of the Bridge hardware.

If both item a) and item c) are true, then the FCS carried in the received frame is still valid, and the ideal approach would be to reuse this value as the FCS for the transmitted frame.

If either item b) or item d) are true, then it will be necessary for the Bridge to recalculate the FCS on transmission, and may therefore need to take additional precautions against in-memory corruption causing increased undetected FCS error levels.

F.2 Basic mathematical ideas behind CRC and FCS

The standard Cyclic Redundancy Check (CRC) algorithm is based on the following ideas:

a) An n-bit message is regarded as a polynomial, M(x), of degree n-1.

b) In order to generate a CRC value of length r bits, a generator polynomial, G(x), is used, of degree r

c) The value of the last r bits of M(x) are chosen such that M(x) ÷ G(x) has a remainder of 0 [i.e., M(x) = 0 mod G(x)].

Messages can be added, in which each coefficient is 0 or 1:

d) M3(x) = M1(x) + M2(x). Messages are added together by bit-wise addition (XOR) of coefficients with the same bit position (i.e., coefficients with the same exponent).

Subtraction of messages:

e) Addition and subtraction are equivalent operations (as A = A XOR B XOR B); hence, using the messages in example d) above, M1(x) can be regenerated from M3(x) by adding M2(x).

Linearity:

f) If M(x) = 0 mod G(X), then (M(x) + E(x)) = 0 mod G(x) IFF E(x) = 0 mod G(x). In other words, if an error pattern, E(x), is added to a message, M(x), then the resultant message will give no remainder when divided by the generator polynomial if both M(x) and E(x) gave no remainder when divided by the generator polynomial.

g) x^mM(x) = 0 mod G(x) IFF M(x) = 0 mod G(x). A message can be shifted by adding zero padding without affecting the integrity of the CRC.

h) Thus, the CRC algorithm has the property that it will detect any burst error, of length r bits or less, applied to the message, as such an error must result in a nonzero remainder when the message is divided by the generator polynomial. Alternatively, adding two messages together, both of which have valid CRCs, results in a message with a valid CRC.

In Ethernet, the algorithm used differs from the standard CRC, and is known as a Frame Check Sequence (FCS). The FCS is based on the CRC but with the following differences:

- The first 32 bits of M(x) are complemented before the FCS value is calculated.
- A CRC value is calculated, as described above, by dividing the first n-32 bits of M(x) by G(x) and taking the remainder as the CRC value.
- The CRC value is complemented and inserted as the last 32 bits of M(x).

By application of the addition and linearity rules, item d) and item f) above, the Ethernet frame with its FCS can be viewed as consisting of the following component messages, added together:

- An n-bit message, M(x), carrying a standard CRC as the last 32 bits.
- An n-bit “FCS adjustment factor,” An(x), which adjusts the CRC to form an FCS.

Figure F-1 illustrates the conversion of a message with a CRC to the equivalent message with an FCS, by the addition of the adjustment factor. The adjustment factor consists of the following two components:

- The first component, when added to M(x), contributes an adjustment to the CRC, which is equivalent to the effect of complementing the first 32 bits of M(x) before calculating the CRC bits, and complements the first 32 bits of M(x).
- The second component, when added to the partially adjusted M(x), restores the first 32 bits of M(x) to their original value, and complements the (adjusted) CRC to form the final FCS.

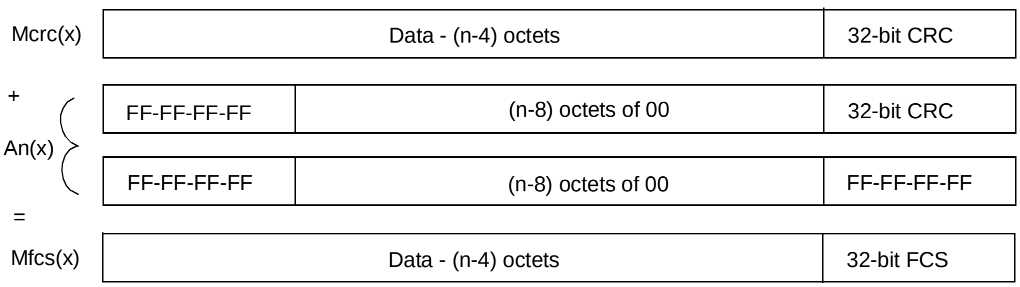


Figure F-1—Converting a CRC to an FCS

F.3 Detection Lossless Circuit approach

This approach is illustrated in Figure F-2. The basis of this approach is that the FCS used in the modified message is always recomputed from scratch using the normal FCS generation algorithm (Check Generator B); however, the original message data and FCS are also used as an input to an FCS checker (Checker A) in order to check that the inputs to the FCS generator were correct. Hence, if errors are introduced into the message while it is in memory in the Bridge in the time period since the message was received and FCS-checked, then Checker A will detect the error at the same time as the new FCS for the modified message is being generated.

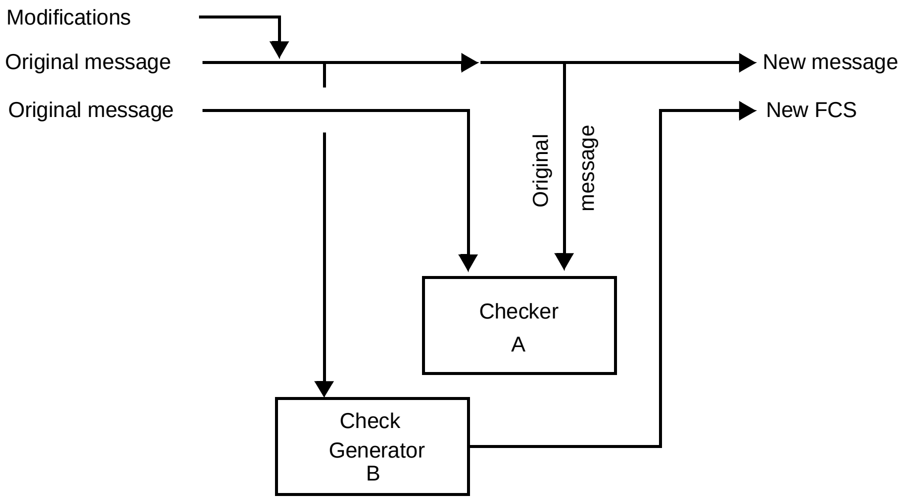


Figure F-2—Detection Lossless Circuit

Checker A is fed both with the original message in its unmodified form, and with the original message as used in the construction of the new message; this occurs after (or simultaneously with) the generation of the new FCS by Check Generator B. Any discrepancies detected by Checker A indicate that the information used to generate the new message and its FCS are suspect, and that the message should therefore be discarded.

F.4 Algorithmic modification of an FCS

In cases where the need to recalculate the FCS comes about as a result of changes to the data rather than by changes in the operation of the FCS algorithm, one option, rather than recalculating an FCS from scratch for a given message, is to examine the changes that have been made to the message and to calculate an FCS adjustment factor that reflects those changes. There are two cases, as follows:

a) The overall length of the message is unchanged, but the contents of one or more octets of data covered by the FCS have changed.
b) The overall length of the message has increased or decreased, as a result of the addition or removal of octets covered by the FCS, but the contents of the original octets is unchanged (although some may have been shifted in position as a result of the insertion/deletion).

F.4.1 Data changed, length unchanged

Adjustment of a message, $MF_1(x)$, in order to change the value of a field F from current value F_1 to new value F_2 consists of the following steps:

— Remove the FCS adjustment factor for a message of length n, An(x).
— Remove the contribution to the message and its FCS provided by the current value of the field, $F_1(x)$.
— Add the contribution to the message and its FCS provided by the new value of the field, $F_2(x)$.
— Add the FCS adjustment factor for a message of length n, An(x).

In other words,

$$MF_2(x) = MF_1(x) – An(x) – F_1(x) + F_2(x) +An(x)$$

The addition and subtraction of An(x) cancel each other out, so,

$$MF_2(x) = MF_1(x) – F_1(x) + F_2(x)$$

Also, the correction factors $F_1(x)$ and $F_2(x)$ can be replaced by a single factor, $F_3(x)$, which is the contribution to M(x) and its FCS that would be provided by the value of Field F that would be produced by F_1 XOR F_2. So,

$$MF_2(x) = MF_1(x) + F_3(x)$$

This is shown in Figure F-3.

Alternatively,

$$\text{FCS of } MF_2(x) = (\text{FCS of } MF_1(x)) + \text{adjustment}$$

where

$$\text{Adjustment} = (\text{FCS of } F_1(x)) + (\text{FCS of } F_2(x))$$

or equivalently,

$$\text{FCS of } MF_2(x) = (\text{FCS of } MF_1(x)) + (\text{FCS of } F_3(x))$$

The adjustment factor (FCS of $F_3(x)$) is incidentally the same factor that would be used if the value of F were to be changed back from F_2 to F_1.

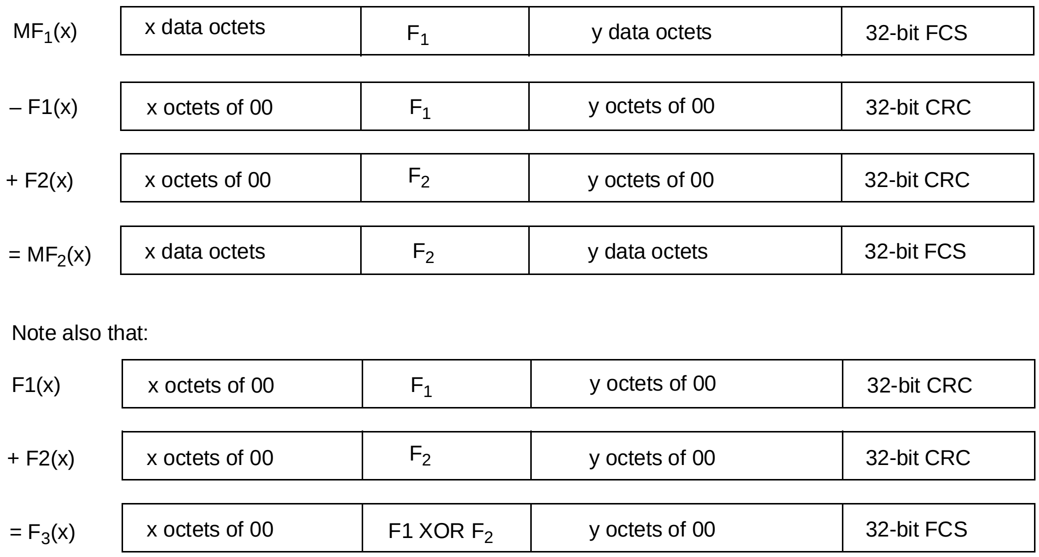


Figure F-3—Field change adjustment

The relative simplicity of this case (relative to the case where the length changes) is a consequence of linearity and the fact that the FCS adjustment factor, An(x), is constant for any given message length n, not a function of the data carried in the message.

F.4.2 Length changed, original data unchanged

Adjustment of a message M(x) of length n to cater for an inserted field, I, of length i involves the following steps:

- Remove the FCS adjustment factor for a message of length n, An(x);
- Remove the contribution to the message and FCS that is provided by the header portion of the message (the portion before the inserted field), H(x);
- Add the contribution to the message and FCS that is provided by the header plus the inserted field I, HI(x);
- Add the FCS adjustment factor for a message of length n+i, An+i(x).

So,

$$MI(x) = M(x) – An(x) – H(x) + HI(x) + An+i(x)$$

This is shown in Figure F-4.

Alternatively,

$$\text{FCS of } MI(x) = (\text{FCS of } M(x)) + \text{adjustment}$$

where

$$\text{adjustment} = (\text{CRC of } An(x)) + (\text{CRC of } H(x)) + (\text{CRC of } HI(x)) + (\text{CRC of } (An+i(x))$$

As with the field change example, the value of *adjustment* is the same adjustment factor that would be used to adjust the FCS of MI(x) if field I were to be removed from the message.

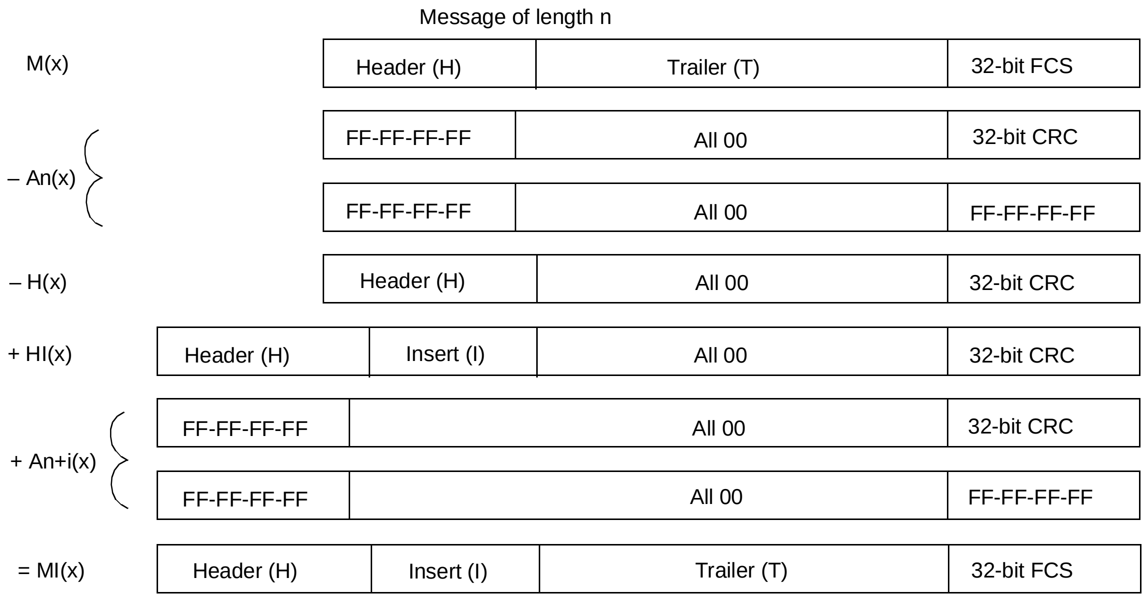


Figure F-4—Field insertion adjustment

F.4.3 Preservation of detectability

With either of the algorithmic adjustment methods described here, the important question is whether the ability of the FCS to detect errors in the message is preserved across these transformations; in other words, if a detectable error pattern, E(x), is added to the message M(x) before the modification, addition, or removal of a field, is that error still detectable after the message has been adjusted.

NOTE—The following analysis assumes that E(x) is a detectable error pattern, of length 32 bits or less. Analysis of the performance of this method with longer error patterns has not been attempted here.

In the cases where the error component is in those parts of the message that are not affected (changed or shifted) by the transformation, it is reasonably apparent that, as the transformation takes no account of the portion of the message that is in error, the detectability of that error is unchanged.

In the case of a field change, where field F_1 is to be replaced with F_2, and E(x) had been applied in some part of the message before changing the field, the new FCS value would be calculated as follows:

$$\text{FCS of } MF_2(x) = (\text{FCS of } MF_1(x)) + (\text{FCS of } F_1(x)) + (\text{FCS of } F_2(x))$$

However, the correction factor that is applied to the FCS contains components based only on F_1 and F_2, and no contribution related to the error pattern E(x). Therefore, after the value of the field has been changed to F_2, the new FCS value as calculated above will still be capable of detecting error pattern E(x), regardless of where the error pattern appears in the message.

In the case of a field insertion (or removal, as insertion and removal have already been shown to be equivalent), where E(x) has been applied to H(x), the correction factor that is applied to the FCS will contain two components related to E(x); the first based on E(x) in its original position, and the second based on E(x) shifted by i, the length of the inserted field, as follows:

$$\text{FCS of } MI(x) = (\text{FCS of } M(x)) + \text{adjustment}$$

where

adjustment = (CRC of An(x)) + (CRC of H(x)) + (CRC of HI(x)) + (CRC of (An+i(x))
+ (CRC of E(x)) + (CRC of (E(x) shifted by i))

The presence of the (CRC of E(x)) component ensures that the final message will still indicate an FCS error. (Note that, even if (CRC of E(x)) and (CRC of (E(x) shifted by i)) turn out to be the same value, the final message will still indicate an FCS error, as its FCS is now correct for the value of MI(x) without the error component.)

The above does not exhaustively analyze all the cases of field insertion and removal, and in particular, does not analyze the cases where the error pattern crosses the insertion/removal boundaries; however, it can be shown that the error detection characteristics of the FCS are preserved in these cases as well.

The conclusion from the above examples is that, as long as the field values used in the calculation of these FCS correction factors are the same as the field values present in the original and final messages (e.g., that the same value of H+E is used both in its original and shifted forms), then the properties of the FCS are preserved. However, if different values are used for calculation of the correction factor from those present in the messages (as could happen, for example, as a result of in-memory corruption), then the properties of the FCS are no longer preserved. Hence, if these algorithmic approaches are used, it is necessary to design the implementation in a manner that ensures that these conditions are met.

F.5 Conclusions

Where it is necessary to recalculate the FCS before transmission, then either the algorithmic FCS modification approaches or the Detection Lossless Circuit approach offer a basis for this to be achieved while still preserving the error detection coverage that was provided by the original FCS.

The Detection Lossless Circuit approach requires access to the entire contents of the original message, to compute the necessary FCS adjustments or to check the integrity of the data that is being used to create the new message and its FCS. In contrast, the algorithmic approach requires access only to the header, insert, and FCS of the original message.

With the algorithmic approach, care needs to be taken in the implementation in order to ensure that the inputs to the correction algorithms are consistent with the contents of the original and modified messages, whereas in the Detection Lossless Circuit, such consistency checking is inherent in the way that the original FCS is used to check the generation of the new FCS.

Annex G

(informative)

User priorities and traffic classes

This annex documents some of the reasoning behind the choice of the default mappings between user_priority values and traffic classes in Table 7-2. There are many possible ways to assign user priority semantics, and to use a limited number of supporting queues. This annex does not attempt to survey the entire range of possibilities, but aims to set out a reasonable set of defaults for use in typical Bridged Local Area Networks providing integrated services.

This standard allows priorities, queue mappings, and queue service disciplines to be managed to best support the user's goals. However, it is widely appreciated that a set of well known and easily understood defaults will greatly facilitate plug and play interworking and the deployment of integrated services. The defaults advocated here were chosen both to support the integrated services mapping work in the IETF (ISSLL, and IS802 in particular), and to provide useful service without any management of the Bridges.

G.1 Traffic types

A full description of the QoS needs of an application and of the network traffic generated, together with characterization of the traffic itself, can be complex—surely too complex to be represented by a simple number 0 through 7. The pragmatic aim of traffic classification is to simplify requirements to preserve the high-speed, low-cost handling characteristic of Bridges. At the margin, potential bandwidth efficiency is traded for simplicity—historically a good decision in the LAN.

The following list of traffic types, each of which can benefit from simple segregation from the others, seems to command widespread support:

a) Network Control—characterized by a "must get there" requirement to maintain and support the network infrastructure.
b) "Voice"—characterized by less than 10 ms delay, and hence maximum jitter (one way transmission through the LAN infrastructure of a single campus).
c) "Video"—characterized by less than 100 ms delay.
d) Controlled Load—important business applications subject to some form of "admission control," from pre-planning of the network requirement at one extreme to bandwidth reservation per flow at the time the flow is started at the other.
e) Excellent Effort—or "CEO's best effort," the best-effort type services that an information services organization would deliver to its most important customers.
f) Best Effort—LAN traffic as we know it today.
g) Background—bulk transfers and other activities that are permitted on the network but that should not impact the use of the network by other users and applications.

G.2 What are we managing?

User priorities and traffic classes facilitate management of

a) Latency, to support new applications.
b) Throughput, to meet service level agreements centered around bandwidth for types of traffic.

With distinct traffic classification and Bridge queuing, latency and bandwidth guarantees can be supported at higher levels of network loading. With few classes the focus is on meeting latency requirements—the bandwidth surplus required in a bursty data environment to guarantee sub-10 ms delays without distinct traffic classification is uneconomically large. As the number of traffic classes that can be used increases, the focus can shift to managing throughput.

The simple default queue servicing policy defined in this standard, strict priority, supports latency management. Active management of bandwidth sharing necessarily requires some Bridge management, to determine shares, etc.

G.3 Traffic type to traffic class mapping

Table G-1 describes a grouping of the traffic types introduced above as the number of Bridge queues increases. Each grouping of types is shown as {*Distinguishing type*, Type, Type, . . .}. The "distinguishing type" is not treated in any way differently in a Bridge, but is italicized here to illustrate, for any given number of queues, which traffic types have driven the allocation of types to classes.

Table G-1—Traffic type to traffic class mapping

Number of queues	Traffic types
1	{*Best Effort*, Excellent effort, Background, Voice, Controlled Load, Video, Network Control}

2	{*Best Effort*, Excellent effort, Background}
	{*Voice*, Controlled Load, Video, Network Control}

3	{*Best Effort*, Excellent effort, Background}
	{*Controlled Load*, Video}
	{*Voice*, Network Control}

4	{*Background*}
	{*Best Effort*, Excellent effort}
	{*Controlled Load*, Video}
	{*Voice*, Network Control}

5	{*Background*}
	{*Best Effort*, Excellent effort}
	{*Controlled Load*}
	{*Video*}
	{*Voice*, Network Control}

6	{*Background*}
	{*Best Effort*
	{*Excellent effort*}
	{*Controlled Load*}
	{*Video*}
	{*Voice*, Network Control}

7	{*Background*}
	{*Best Effort*}
	{*Excellent effort*}
	{*Controlled Load*}
	{*Video*}
	{*Voice*}
	{*Network Control*}

This grouping is proposed as the default user priority to traffic class mapping, with the correspondence of traffic types to user priority numbers as shown further below. It is designed to work with default queue handling. The remaining user priority value (only seven instead of eight traffic types have been discussed) is also assigned within the priority hierarchy.

The logic behind the step by step breaking out of traffic types as more classes become available runs is:

a) With a single queue, there are no choices, everything functions as with Bridges conformant to IEEE Std 802.1D, 1993 Edition. If a distinguishing traffic type were to be identified, it would be Best Effort.

b) In order to support integrated services (voice, video, and data) in the presence of bursty best effort data, it is necessary to segregate all the time-critical traffic. In addition, further traffic that is to receive superior service and that is operating under admission control also needs to be separated from the uncontrolled traffic. The amount of priority traffic will be restricted by the need to support low latency (better than 10 ms one way for the local LAN infrastructure) for Voice, which becomes the defining type for the additional queue.

c) A further queue is best used to separate Controlled Load, i.e., well-behaved traffic with admission control or policy constraints, from the very latency sensitive Voice traffic. This allows rather more controlled load (and video) traffic than with two queues, while still allowing voice latencies to be met. However, all the traffic in the new queue will still have to meet interactive video latencies (if any such traffic is present), so there may still be artificially low limits on the throughput of well-behaved traffic.

d) Up to this point, additional queues have been used first to address latency issues, and next to recover throughput for well-behaved traffic with less onerous latency constraints. This has now been largely achieved for the major bandwidth users, and the focus can shift to distinguishing between the different types of uncontrolled bursty traffic, distinguishing between them on the basis of business importance. The first step is to deal with high throughput background traffic, which may cause TCP windows and timers for normal best effort (and excellent effort) traffic to slow, by segregating Background traffic.

e) Allocating the next queue to (interactive) Video, traffic with latency and jitter less than 100 ms, frees the controlled load traffic from the latency constraint, raising the achievable throughput.

f) The next queue is devoted to a little extra job security by separating business critical applications that are not sufficiently well behaved to have been classified as controlled load into their own Excellent Effort category.

g) Finally, Network Control can be segregated from Voice, though the benefits of this are unlikely to be felt with default queue handling, since network control is likely to be less delay sensitive than Voice (but probably more than background), but requires a higher delivery guarantee.

h) Eight queues are not shown, since only seven distinct types of traffic are shown. From the point of view of defining defaults at the present time it seems reasonable to allocate an additional priority around best effort to support bandwidth sharing management for bursty data applications.

Table G-2 shows the correspondence between traffic types and user_priority values used to select the defaults in Table 7-2, and defines a set of acronyms for the traffic types.

Table G-2—Traffic type acronyms

user_priority	Acronym	Traffic type
1	BK	Background
2	—	Spare
0 (Default)	BE	Best Effort
3	EE	Excellent Effort
4	CL	Controlled Load
5	VI	“Video,” < 100 ms latency and jitter
6	VO	“Voice,” < 10 ms latency and jitter
7	NC	Network Control

Table G-3 compresses the lists of traffic types for queues shown in Table G-1, using the acronyms defined in Table G-2, and shows just the defining traffic types for the given number of queues.

Table G-3—Defining traffic types

<table>
<tr><th>Number of queues</th><th colspan="8">Defining traffic type</th></tr>
<tr><td>1</td><td colspan="8">BE</td></tr>
<tr><td>2</td><td colspan="4">BE</td><td colspan="4">VO</td></tr>
<tr><td>3</td><td colspan="4">BE</td><td colspan="2">CL</td><td colspan="2">VO</td></tr>
<tr><td>4</td><td colspan="2">BK</td><td colspan="2">BE</td><td colspan="2">CL</td><td colspan="2">VO</td></tr>
<tr><td>5</td><td colspan="2">BK</td><td colspan="2">BE</td><td>CL</td><td>VI</td><td colspan="2">VO</td></tr>
<tr><td>6</td><td colspan="2">BK</td><td>BE</td><td>EE</td><td>CL</td><td>VI</td><td colspan="2">VO</td></tr>
<tr><td>7</td><td colspan="2">BK</td><td>BE</td><td>EE</td><td>CL</td><td>VI</td><td>VO</td><td>NC</td></tr>
<tr><td>8</td><td>BK</td><td>—</td><td>BE</td><td>EE</td><td>CL</td><td>VI</td><td>VO</td><td>NC</td></tr>
</table>

G.3.1 Traffic types and user priority values

The default user priority used for transmission by token ring stations is 0, as is the default assumed by IEEE Std 802.1D, 1993 Edition-conformant Bridges for frames currently received from an Ethernet. Changing this default would result in some confusion, and likely some lack of interoperability. At the same time the default traffic type today is definitely Best Effort. The proposed solution to this dilemma is to continue to use zero both for default user priority and for best effort data transmission, and associate Background, the spare value, and Excellent Effort with user priority values of 1, 2, and 3, respectively. This means that the values 1 and 2 effectively communicate a lower priority than 0.

Annex H

(informative)

Generic Attribute Registration Protocol Design

This annex documents some of the more significant design decisions taken during the development of GARP. The main purpose of this annex is to ensure that the history of such decisions is clear, thus avoiding repeated reexamination of issues that have already been resolved, and reinvention of solutions to them.

H.1 Use of an unconfirmed protocol

GARP relies on the assumption that as long as the state machines are designed to ensure that more than one BPDU conveying the critical messages (JoinIn, JoinEmpty and Empty) is sent for each set of related messages, there is a high probability that all intended recipients will receive them. Early attempts at designing a GARP that involved explicit confirmations revealed a number of problems, particularly:

a) Where multiple GARP Participants attach to a LAN, it is undesirable for all to respond to a request from one Participant. Limiting the responses to a single Participant is feasible; however, the value of a response indicating “I got that” from a single Participant is dubious, given that there is no guarantee that the other Participants involved have also seen the same message.
b) Developing a distribution mechanism that gives a high level of guarantee of correct operation is clearly possible (e.g., X.500’s distributed directory protocol); however, the complexity of such an approach would conflict with the fundamental design goals of MAC Bridges, i.e., that they are based on simple mechanisms and can therefore be engineered for high performance/cost ratios.
c) If a basic design goal of handling single packet loss is assumed, strategies based on multiple transmission of messages can provide the appropriate level of reliability while maintaining simplicity.
d) The users of the MAC Service have other means at their disposal to determine whether a service is being provided. For example, if the user requests receipt of a video channel, it will soon become apparent as to whether it is being received. The user/application concerned can therefore adopt “higher layer” recovery strategies to deal with the situation, without impacting the complexity of the Bridge. This is in any case probably appropriate, as the Bridge would not be in a position to determine appropriate courses of action in some combinations of user requirement and error condition.

H.2 Design of the Applicant state machine

The Applicant state machine has been designed to implement the requirements of GARP Participants in both end stations and Bridges. Strictly speaking, the requirements of the Applicant are slightly simpler in the end station than those of the Bridge; however, the two have been combined into a single description for reasons of overall clarity and simplicity. The main differences are as follows.

An end station that does not need to know about the registration status of other Participants (e.g., a recipient of traffic for a Attribute) is mainly concerned with ensuring that all other Participants attached to its LAN see its Join messages. If one of its Leave messages goes astray, the Leave All garbage collection mechanism will eventually fix the problem, so reliable reception of Leaves by other Participants is not a major issue. The state machine for the Applicant in such an end station could be further simplified, to one in which it simply sends two JoinIns to enter the Active Member state, and sends a single Leave to enter the Observer state.

For end stations or Bridge Ports that are concerned not only with their own registrations but also with the registration state of other Participants, the Registrar's situation is more involved. The major difference this creates is that if there are other Members or would-be Members on the same LANs it is important that at least two of the members are heard by all Participants. This then ensures that each Participant, including those that actually generate the messages, will hear at least one Join on the LAN. Hearing two different Participants is not the same thing as seeing two Join messages; both could potentially have been generated by the same Participant. Hence, the Join messages sent in GARP carry an indication of the Participant's registration state. JoinIn indicates "I wish to Join and I have seen one or more other Joins for this Attribute;" JoinEmpty indicates "I wish to Join and I have not seen any other Joins for this Attribute." In effect, the flavor of Join message sent reflects the registration state held by the Registrar for that Attribute. With this refinement, a Participant that considers itself to be in a Member state can respond to a JoinEmpty (indicating that the originator of the JoinEmpty has not registered its membership) by sending a JoinIn.

There are the following possibilities that apply when a Participant attempts to become a Member:

a) There are no other Participants in membership for that Attribute.
b) There is one other Participant in membership for that Attribute.
c) There are two or more other Participants in membership for that Attribute.

In case a), the new Member will send two JoinEmpty messages, and enter the Quiet/Active Member state. All other Participants on the LAN will see one or both of the JoinEmpty messages, and their Registrar state will therefore be IN for that Attribute.

In case b), the new Member sends two JoinIn messages, as the Registrar state is IN (one other member exists); this will ensure that the other Member registers the Attribute.

In case c), the new Member has seen two other Participants Join, so it can therefore enter the Passive Member state without sending a Join message at all, and by a similar argument, can become an Observer again without issuing a Leave.

H.3 Design of the Registrar state machine

The Registrar state machine is entirely passive, i.e., it does not transmit GARP PDUs; its job is simply to record the current registration state of an Attribute. The IN and LV states indicate a current registration; the MT state indicates de-registration. The LV to MT transition occurs if LeaveTime elapses since the last Leave or LeaveAll was seen and no Join has been received in response.

H.4 Analysis of GARP State Machine Operation

This subclause shows some example GARP protocol sequences, in order to

a) Illustrate normal operation of GARP.
b) Illustrate GARP operation under some failure conditions.

Each illustration consists of the following elements:

c) Text and a diagram, describing the scenario, the topology concerned, and the component stations/ bridges.
d) A sequence showing the state transitions that occur in the state machines in each component involved in the scenario. These sequences are documented using *state sequence tables;* each row of the table shows the combined state of all components at an instant in time, plus any "system state,"

that is to say, state that is in the system as a whole, but that is not visible simply by looking at the state of the component end stations/Bridges. An example of “system state” would be a Join PDU that has been sent, but that has not yet been received at its destination. Rows earlier in the table represent combined states that occurred earlier in the time sequence. The table contains columns for each state machine type, plus a final column for descriptive commentary. Where appropriate, the abbreviations introduced in 12.7 are used.

H.4.1 Initial Join Scenarios

Topology: a leaf LAN, with a single station, A, attached; see Figure H-1.

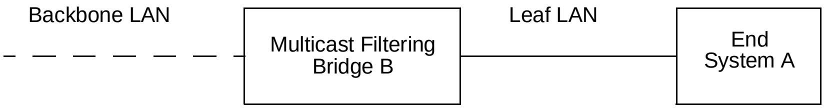


Figure H-1—Topology: Leaf LAN Scenarios

First Scenario: A requests membership for Attribute AT, which hitherto has had no members; Bridge B responds normally. No packet loss occurs during the exchange. This scenario is analyzed in Table H-1. As can be seen, B considers the Attribute to be registered after the first JoinEmpty has been received from A, at step 4.

Table H-1—Initial Join: No Packet Loss

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	VO — —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No “system” state; A has no Registrar state machine.
2	Applicant Registrar “System”	VP — —	VO MT —	A’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar “System”	AA — JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar “System”	AA — —	VO IN —	B receives the JoinEmpty; the Registrar registers the Attribute (IN) and the Applicant state is unchanged.
5	Applicant Registrar “System”	QA — JE(AT)	VO IN —	A sends a second JoinEmpty, and enters QA.
6	Applicant Registrar “System”	QA — —	VO IN —	The second JoinEmpty has no effect on the state of B.

Second Scenario: A requests membership for Attribute AT, which hitherto has had no members; B fails to see A’s first JoinEmpty PDU. This scenario is analyzed in Table H-2. B does not register the Attribute until it receives A’s second JoinEmpty PDU, at step 6.

Table H-2—Initial Join: Packet Loss on First Join PDU

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	VO — —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No “system” state; A has no Registrar state machine.
2	Applicant Registrar “System”	VP — —	VO MT —	A’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar “System”	AA — JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar “System”	AA — —	VO MT —	B fails to see the JoinEmpty; the Registrar and Applicant state is unchanged.
5	Applicant Registrar “System”	QA — JE(AT)	VO MT —	A sends a second JoinEmpty, and enters QA.
6	Applicant Registrar “System”	QA — —	VO IN —	The second JoinEmpty causes B’s Registrar to enter the IN state.

Third Scenario: A requests membership of Attribute AT, which hitherto had no members; B fails to see A’s second Join PDU. Analyzed in Table H-3, for practical purposes indistinguishable from the First Scenario.

Table H-3—Initial Join: Packet Loss on Second Join PDU

Step	State Of:	End Station A	Bridge B	Commentary
1	Applicant Registrar “System”	VO — —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No “system” state; A has no Registrar state machine.
2	Applicant Registrar “System”	VP — —	VO MT —	A’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar “System”	AA — JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar “System”	AA — —	VO IN —	B receives the JoinEmpty; the Registrar registers the Attribute (IN) and the Applicant state is unchanged.
5	Applicant Registrar “System”	QA — JE(AT)	VO IN —	A sends a second JoinEmpty, and enters QA.
6	Applicant Registrar “System”	QA — —	VO IN —	The second JoinEmpty is lost, and has no effect on the state of B.

H.4.2 Last to Leave Scenarios

Topology: as described for H.4.1, Initial Join Scenario.

First Scenario: A requests to de-register membership for Attribute AT, which hitherto had only A as a member; Bridge B responds normally. No packet loss occurs during the exchange. This scenario is analyzed in Table H-4. As can be seen, B considers the Attribute still to be registered until the Leave Timer has expired. Other members get two opportunities to declare the Attribute, as both a Leave Empty and an Empty message are transmitted before the Attribute is finally de-registered by B.

Table H-4—Last To Leave: No Packet Loss

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	QA — —	VO IN —	Starting conditions: Attribute AT active in A’s Applicant and B’s Registrar state machines. A has no Registrar state machine.
2	Applicant Registrar “System”	LA — —	VO IN —	A’s Applicant receives a ReqLeave primitive from the GARP Application. A prepares to issue a Leave by entering the LA state.
3	Applicant Registrar “System”	VO — LE(AT)	VO IN —	A transmission opportunity occurs; A transmits Leave Empty and enters the VO state.
4	Applicant Registrar “System”	VO — —	LO LV —	B sees the Leave Empty. The Registrar starts Leave Timer; the Applicant enters LO and waits for a transmission opportunity.
5	Applicant Registrar “System”	VO — —	VO LV E(AT)	B transmits an Empty message for AT, to prompt any remaining members to rejoin.
6	Applicant Registrar “System”	VO — —	VO LV —	The Empty message has no effect on A.
7	Applicant Registrar “System”	VO — —	VO MT —	The Leave Timer expires; B’s Registrar de-registers the Attribute.

Second Scenario: A requests to de-register membership for Attribute AT, which hitherto has had only A as a member; Bridge B fails to see A's Leave Empty PDU. This scenario is analyzed in Table H-5. On average, this scenario takes B half a Leave All Period plus a Leave Period before it finally de-registers the Attribute.

Table H-5—Last To Leave: Packet Loss on First Leave PDU

Step	State Of:	A	B	Commentary
1	Applicant Registrar "System"	QA — —	VO IN —	Starting conditions: Attribute AT active in A's Applicant and B's Registrar state machines. A has no Registrar state machine.
2	Applicant Registrar "System"	LA — —	VO IN —	A's Applicant receives a ReqLeave primitive from the GARP Application. A prepares to issue a Leave by entering the LA state.
3	Applicant Registrar "System"	VO — LE(AT)	VO IN —	A transmission opportunity occurs; A transmits Leave Empty and enters the VO state.
4	Applicant Registrar "System"	VO — —	VO IN —	B fails to see the Leave Empty.
5	Applicant Registrar "System"	VO — —	LO LV LeaveAll	B transmits a LeaveAll message. B's Registrar enters LV and starts the Leave Timer. The Applicant enters LO.
6	Applicant Registrar "System"	VO — —	LO LV —	The Leave All message has no effect on A.
7	Applicant Registrar "System"	VO — —	VO LV E(AT)	A transmission opportunity occurs; B transmits an Empty message for AT, and the Applicant returns to VO.
8	Applicant Registrar "System"	VO — —	VO LV —	The Empty message has no effect on A.
9	Applicant Registrar "System"	VO — —	VO MT —	The Leave Timer expires. B's Registrar de-registers the attribute.

Third Scenario: Strictly speaking, there are further single packet loss scenarios to be considered, involving loss of B's Leave All and Empty messages, but the sequential analysis is identical to the second scenario, as the Empty or Leave All messages would have had no effect on A's state in either case.

H.4.3 Leave/Rejoin Scenarios—Single Member

Topology: as described for H.4.1, Initial Join Scenario.

First Scenario: A is a member for Attribute AT, which has no other members. Bridge B issues a Leave All; A rejoins. No packet loss occurs during the exchange. This scenario is analyzed in Table H-6. As can be seen, A remains registered at all times. A slight variant of this scenario occurs if B manages to transmit Empty before A transmits the first Join Empty (between steps 3 and 4); however, this has no effect on steps 4 through 7.

Table H-6—Single Member LeaveAll/Rejoin: No Packet Loss

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	QA — —	VO IN —	Starting conditions: Attribute AT active in A’s Applicant and B’s Registrar state machines. A has no Registrar state machine.
2	Applicant Registrar “System”	QA — —	LO LV LeaveAll	B sends a LeaveAll. This causes B to enter LO and LV states for Applicant and Registrar, and to start the Leave Timer.
3	Applicant Registrar “System”	VP — —	LO LV —	A becomes very anxious, and awaits an opportunity to send a JoinEmpty.
4	Applicant Registrar “System”	AA — JE(AT)	LO LV —	A transmits JoinEmpty, and becomes less anxious.
5	Applicant Registrar “System”	AA — —	VO IN —	B sees the Join, registers the membership and re-enters VO.
6	Applicant Registrar “System”	QA — JE(AT)	VO IN —	A transmits a second JoinEmpty, and becomes Quiet.
7	Applicant Registrar “System”	QA — —	VO IN —	The second Join Empty has no effect on B.

Second Scenario: A is a member of Attribute AT, which has no other members. Bridge B issues a Leave All; A fails to see it, and does not rejoin until B issues the Empty message. This scenario is analyzed in Table H-7. As can be seen, A remains registered at all times, despite the loss of the LeaveAll.

Table H-7—Single Member Leave/Rejoin: Packet Loss on LeaveAll PDU

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	QA — —	VO IN —	Starting conditions: Attribute AT active in A’s Applicant and B’s Registrar state machines. A has no Registrar state machine.
2	Applicant Registrar “System”	QA — —	LO LV LeaveAll	B sends a LeaveAll. This causes B to enter LO and LV states for Applicant and Registrar, and to start the Leave Timer.
3	Applicant Registrar “System”	QA — —	LO LV —	A does not see the LeaveAll.
4	Applicant Registrar “System”	QA — —	VO LV E(AT)	A transmission opportunity occurs for B; it sends an Empty.
5	Applicant Registrar “System”	VA — —	VO LV —	A sees the Empty and becomes Very Anxious.
6	Applicant Registrar “System”	AA — JE(AT)	VO LV —	A transmission opportunity occurs for A. It transmits a JoinEmpty, and becomes Anxious.
7	Applicant Registrar “System”	AA — —	VO IN —	B sees the JoinEmpty, and re-registers the Attribute.
8	Applicant Registrar “System”	QA — JE(AT)	VO IN —	A transmits a second JoinEmpty, and becomes Quiet.
9	Applicant Registrar “System”	AA — —	VO IN —	The second JoinEmpty has no effect.

Third Scenario: A is a member of Attribute AT, which has no other members. Bridge B issues a Leave All; A responds with a JoinEmpty which B does not see, so B issues an Empty message. This scenario is analyzed in Table H-8. As can be seen, A remains registered at all times, despite the loss of the first JoinEmpty. The effect of the Empty is to reset A’s count of Joins, so three Join Empty messages are sent before A becomes Quiet.

Table H-8—Single Member Leave/Rejoin: Packet Loss on First Join PDU

Step	State Of:	A	B	Commentary
1	Applicant Registrar “System”	QA — —	VO IN —	Starting conditions: Attribute AT active in A’s Applicant and B’s Registrar state machines. A has no Registrar state machine.
2	Applicant Registrar “System”	QA — —	LO LV LeaveAll	B sends a LeaveAll. This causes B to enter LO and LV states for Applicant and Registrar, and to start the Leave Timer.
3	Applicant Registrar “System”	VP — —	LO LV —	A becomes very anxious, and awaits an opportunity to send a JoinEmpty.
4	Applicant Registrar “System”	AA — JE(AT)	LO LV —	A transmits JoinEmpty, and becomes less anxious.
5	Applicant Registrar “System”	AA — —	LO LV —	B fails to see the Join Empty.
6	Applicant Registrar “System”	AA — —	VO LV E(AT)	A transmission opportunity occurs for B; it sends an Empty.
7	Applicant Registrar “System”	VA — —	VO LV —	A sees the Empty and becomes Very Anxious.
8	Applicant Registrar “System”	AA — JE(AT)	VO LV	A transmits a second JoinEmpty, and becomes Anxious.
9	Applicant Registrar “System”	AA — —	VO IN —	B sees the JoinEmpty, and re-registers the Attribute.
10	Applicant Registrar “System”	QA — JE(AT)	VO IN	A transmits a third JoinEmpty, and becomes Quiet.
11	Applicant Registrar “System”	AA — —	VO IN —	The third JoinEmpty has no effect.

H.4.4 Backbone LAN Initial Join Scenarios

Topology: a backbone LAN, with two Bridges, A and B, attached; see Figure H-2.

The scenarios look only at the behavior of the Bridge Ports that are connected to the backbone LAN.

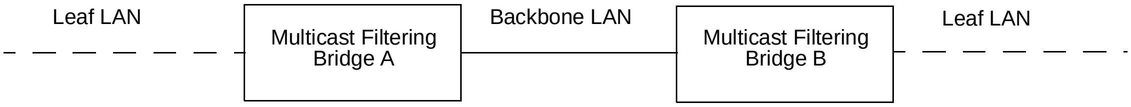


Figure H-2—Topology: Backbone LAN Scenarios

First Scenario: A requests membership for Attribute AT on the backbone, as a result of initial registrations by Applicants on its leaf LAN. B registers the same Attribute some time later. Both Bridges respond normally. No packet loss occurs during the exchange. This scenario is analyzed in Table H-9. As can be seen, B considers the Attribute to be registered after the first Join has been received from A; similarly, A considers that the Attribute is registered after B's first JoinIn is received.

Table H-9—Backbone Initial Join: No Packet Loss

Step	State Of:	A	B	Commentary
1	Applicant Registrar "System"	VO MT —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No "system" state.
2	Applicant Registrar "System"	VP MT —	VO MT —	A's Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar "System"	AA MT JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar "System"	AA MT —	VO IN —	B receives the JoinEmpty; the Registrar registers the Attribute (IN) and the Applicant state is unchanged.
5	Applicant Registrar "System"	AA MT —	VP IN —	B's Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
6	Applicant Registrar "System"	AA MT —	AA IN JI(AT)	B issues a JoinIn.
7	Applicant Registrar "System"	QA IN —	AA IN —	B's JoinIn causes A to register the attribute, and also suppresses A's second Join.
8	Applicant Registrar "System"	QA IN —	QA IN JI(AT)	B issues a second JoinIn and becomes Quiet.
9	Applicant Registrar "System"	QA IN —	QA IN —	B's second JoinIn has no effect.

Second Scenario: A requests membership of Attribute AT on the backbone, as a result of initial registrations by Applicants on its leaf LAN. Bridge B fails to see A's first Join PDU. A issues a second Join after Join Time. B subsequently joins the same Attribute. This scenario is analyzed in Table H-10. B does not register the Attribute for A until it sees A's second Join, one JoinTime after A's initial Join attempt.

Table H-10—Backbone Initial Join: Packet Loss on First Join PDU

Step	State Of:	A	B	Commentary
1	Applicant Registrar "System"	VO MT —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No "system" state.
2	Applicant Registrar "System"	VP MT —	VO MT —	A's Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar "System"	AA MT JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar "System"	AA MT —	VO MT —	B fails to see the JoinEmpty.
5	Applicant Registrar "System"	QA MT JE(AT)	VO MT —	A sends a second JoinEmpty, and enters QA.
6	Applicant Registrar "System"	QA MT —	VO IN —	B receives the JoinEmpty; the Registrar registers the Attribute (IN) and the Applicant state is unchanged.
7	Applicant Registrar "System"	QA MT —	VP IN —	B's Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
8	Applicant Registrar "System"	QA MT —	AA IN JI(AT)	B issues a JoinIn.
9	Applicant Registrar "System"	QA IN —	AA IN —	B's JoinIn causes A to register the attribute.
10	Applicant Registrar "System"	QA IN —	QA IN JE(AT)	B issues a second JoinIn and becomes Quiet.
11	Applicant Registrar "System"	QA IN —	QA IN —	B's second JoinIn has no effect.

Third Scenario: A requests membership of Attribute AT on the backbone, as a result of initial registrations by Applicants on its leaf LAN. Bridge B fails to see A's first Join PDU, but issues a Join itself for the same Attribute. This scenario is analyzed in Table H-11. B does not register the Attribute for A until after A's second Join attempt.

Table H-11—Backbone Initial Join: simultaneous Join from two Bridges with packet loss on first Join

Step	State Of:	Bridge A	Bridge B	Commentary
1	Applicant Registrar “System”	VO MT —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No members. No “system” state.
2	Applicant Registrar “System”	VP MT —	VO MT —	A’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar “System”	AA MT JE(AT)	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar “System”	AA MT —	VO MT —	B fails to see the JoinEmpty.
5	Applicant Registrar “System”	AA MT —	VP MT —	B’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
6	Applicant Registrar “System”	AA MT —	AA MT JE(AT)	B sends a JoinEmpty; and enters AA.
7	Applicant Registrar “System”	VA IN —	AA MT —	A receives the JoinEmpty. This causes it to enter VA, and to register the Attribute.
8	Applicant Registrar “System”	AA IN JI(AT)	AA MT —	A sends a JoinIn and enters AA.
9	Applicant Registrar “System”	AA IN —	QA IN —	B sees the JoinIn; it registers the attribute and further Joins from B are suppressed.
10	Applicant Registrar “System”	QA IN JE(AT)	QA IN —	A issues a second JoinIn and becomes Quiet.
11	Applicant Registrar “System”	QA IN —	QA IN —	A’s second JoinIn has no effect on B.

H.4.5 Shared media LAN scenarios

Topology: a backbone LAN, with two Bridges, A and B, attached; see Figure H-3.

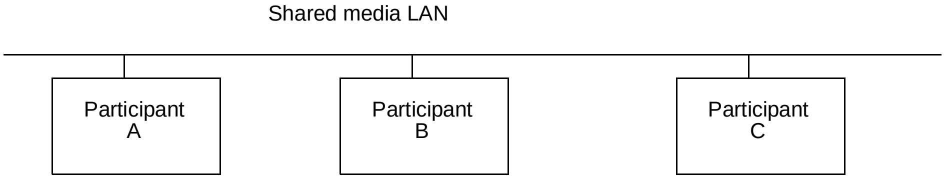


Figure H-3—Topology: Shared media LAN scenarios

First scenario: Illustrates the point, mentioned in H.2, that if there are already two or more Members on a given LAN, then further Members can join and leave without issuing any GARP messages. Three Participants are attached to the LAN. A and B Join for Attribute AT (an identical sequence to Table H-9). Subsequently, C can Join and Leave without issuing any GARP messages. This scenario is analyzed in Table H-12.

Table H-12—Shared media: Third party can Join/Leave without sending GARP messages

Step	State Of:	A	B	C	Commentary
1	Applicant Registrar “System”	VO MT —	VO MT —	VO MT —	Starting conditions: Attribute AT unregistered in all state machines. No “system” state.
2	Applicant Registrar “System”	VP MT —	VO MT —	VO MT —	A’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
3	Applicant Registrar “System”	AA MT JE(AT)	VO MT —	VO MT —	A sends a JoinEmpty, and enters AA.
4	Applicant Registrar “System”	AA MT —	VO IN —	VO IN —	B and C receive the JoinEmpty; the Registrar registers the Attribute (IN) and the Applicant state is unchanged.
5	Applicant Registrar “System”	AA MT —	VP IN —	VO IN —	B’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. This causes it to enter VP and wait for a transmission opportunity.
6	Applicant Registrar “System”	AA MT —	AA IN JI(AT)	VO IN —	B issues a JoinIn.
7	Applicant Registrar “System”	QA IN —	AA IN —	AO IN —	B’s JoinIn causes A to register the attribute, and also suppresses A’s second Join. It also causes C’s Applicant to enter the AO (Anxious Observer) state.
8	Applicant Registrar “System”	QA IN —	QA IN JI(AT)	AO IN —	B issues a second JoinIn and becomes Quiet.
9	Applicant Registrar “System”	QA IN —	QA IN —	QO IN —	B’s second JoinIn has no effect on A but causes C to enter the QO (Quiet Observer) state.
10	Applicant Registrar “System”	QA IN —	QA IN —	QP IN —	C’s Applicant receives a ReqJoin primitive from the GARP Application for Attribute AT. Because C has seen two JoinIn’s for AT, it is able to directly enter the QP (Quiet/Passive member) state, without issuing any GARP messages.
11	Applicant Registrar “System”	QA IN —	QA IN —	QO IN —	C’s Applicant receives a ReqLeave primitive from the GARP Application for Attribute AT. Because it Joined AT without issuing any messages, it is able to Leave by directly entering the QO (Quiet/ Observer) state, without issuing any GARP messages.

Second scenario: Shows the “last to leave” sequence for circumstances where more than two participants exist on a LAN. A has declared Attribute AT; B and C have registered AT in their Registrar state machines. A Leaves. The protocol exchanges occur normally. This scenario is analyzed in Table H-13.

NOTE—This second scenario is included primarily to demonstrate that the length of time taken for the last member to Leave is not a function of the number of Participants involved. This was a problem that existed with earlier versions of GARP, where the receipt of an Empty message by a registrar in the LV state (or its equivalent) caused the Leave Timer to be restarted. In the protocol as described, the time taken for the de-registration to occur after the last member leaves is equal to one Leave Time, regardless of the number of Participants.

Table H-13—Shared media: Leave sequence for three Participants

Step	State Of:	A	B	C	Commentary
1	Applicant Registrar “System”	QA MT —	VO IN —	VO IN —	Starting conditions: Attribute AT active in A’s Applicant and in B and C’s Registrar state machines.
2	Applicant Registrar “System”	LA MT —	VO IN —	VO IN —	A’s Applicant receives a ReqLeave primitive from the GARP Application. A prepares to issue a Leave by entering the LA state.
3	Applicant Registrar “System”	VO MT LE(AT)	VO IN —	VO IN —	A transmission opportunity occurs; A transmits Leave Empty and enters the VO state.
4	Applicant Registrar “System”	VO MT —	LO LV —	LO LV —	B and C see the Leave Empty. Their Registrars start Leave Timer; their Applicants enter LO and wait for a transmission opportunity.
5	Applicant Registrar “System”	VO MT —	VO LV E(AT)	LO LV —	A transmission opportunity for B occurs; it transmits an Empty message for AT, to prompt any remaining members to rejoin.
6	Applicant Registrar “System”	VO MT —	VO LV —	VO LV —	The Empty message has no effect on A, but suppresses C’s intent to transmit an Empty; C enters VO.
7	Applicant Registrar “System”	VO MT —	VO MT —	VO MT —	The Leave Timers expire in B and C; their Registrars both de-register the Attribute.

Annex I

(informative)

Introduction of GARP, GMRP, and Extended Filtering Services

I.1 Migration considerations

This subclause examines some of the issues, and their possible solution, that will be encountered in mixed mode LANs; i.e., LANs that contain

a) A mixture of Legacy Bridges (Bridges that support only Basic Filtering Services) and Enhanced Bridges (Bridges that support Extended Filtering Services).

b) A mixture of legacy end stations, that are GMRP-unaware, and GMRP-aware end stations.

Given the range of uses, configurations, traffic patterns, and performance requirements that are to be found in LAN environments is impossibly large, this text is not intended to be definitive; it is essential that the migration strategy in a given network be defined in terms of its own particular environment and requirements.

I.1.1 Heterogeneous Bridge environments

The operation of GMRP is entirely transparent to Legacy Bridges; i.e., all GMRP PDUs are forwarded transparently by Legacy Bridges. It is therefore possible to intermix Legacy Bridges and Enhanced Bridges in an arbitrary manner without affecting the integrity of the GMRP-generated directed graphs.

NOTE—This is true only if the Legacy Bridges do not contain any Static Filtering Entries that might interfere with the transmission of PDUs destined for any of the GARP Application addresses.

In a heterogeneous Bridged Local Area Network, placing any Enhanced Bridges at the periphery of the network, rather than within the core of the network, is likely to produce the maximum benefit in terms of traffic segregation. However, such benefit may also depend on the mix and distribution of GMRP-aware and GMRP-unaware end stations (see I.1.2). In general, the nearer to the “center” of the network that a particular Bridge is placed, the more likely it is that any Groups defined in its Filtering Database will have registered members on all Ports; the Bridge will therefore be filtering relatively little Group traffic. Placing a Legacy Bridge at a similar location in the network therefore produces relatively little additional load on adjacent LANs. Conversely, the nearer the periphery of the network, the more likely it is that the Groups defined in the Filtering Database will have different membership patterns on each Port and Extended Filtering Services will therefore be of more value.

I.1.2 Heterogeneous end station environments

I.1.2.1 Use of Basic Filtering Mode and legacy bridges

As legacy Bridges are transparent to GARP protocol exchanges, the placement of end stations on LANs served only by Legacy Bridges is of little interest to this discussion.

I.1.2.2 Use of Forward All Groups

Forward All Groups is the default Group filtering behavior (7.9.4) adopted by a given Port if the static and dynamic information held in the Filtering Database for that Port indicates a service requirement of Forward All Groups. Registering Forward All Groups as a service requirement (6.6.7) is intended to be used for two purposes:

a) To ensure that regions of the network containing only legacy devices can receive all multicast frames that originate in any other region of the network (except those that are explicitly disallowed by means of static configuration).
b) To allow successful operation of devices that require promiscuous reception, such as routers or network monitors, as discussed in I.2.

Ideally, all GMRP-unaware devices would be attached to LANs served by Bridge Ports operating with Forward All Groups as their default Group filtering behavior. As these LANs will attract all multicast traffic originating in any other LAN, such legacy LANs are ideally located in the same region of the network. As indicated in I.1.1, it may also be appropriate to attach these LANs to a Bridge that is close to the center of the network.

I.1.2.3 Use of Forward Unregistered Groups

Forward Unregistered Groups is the default Group filtering behavior (7.9.4) adopted by a given Port if the static and dynamic information held in the Filtering Database for that Port indicates a service requirement of Forward Unregistered Groups, but does not indicate a service requirement of Forward All Groups. One of the intended uses of Forward Unregistered Groups registration is to provide for migration from GMRP-unaware environments to GMRP-aware environments. The “pass-through” behavior for unregistered Groups will allow GMRP-unaware end stations to continue to operate on LANs serviced by Forward Unregistered Groups filtering behavior, as long as the multicast addresses they wish to receive are distinct from those used by GMRP-aware end stations (I.1.2.5 looks at the situation where the addresses used are not distinct). In general, it is more appropriate to place GMRP-unaware end stations on LANs served by Ports providing Forward All Groups as their default Group filtering behavior.

Forward Unregistered Groups could also be useful in circumstances where GMRP-aware devices are able to distinguish between a set of “legacy” multicast addresses, for which they do not register, and a set of “new” multicast addresses, for which they register. Again, this scenario is only workable if the GMRP-unaware devices use only the “legacy” multicast addresses.

I.1.2.4 Use of Filter Unregistered Groups

Filter Unregistered Groups, which requires explicit registration of Group membership in order for an end station to receive frames destined for a Group, is the default Group filtering behavior (7.9.4) adopted by a given Port if the static and dynamic information held in the Filtering Database for that Port does not indicate a service requirement of Forward All Groups or Forward Unregistered Groups. It is essentially designed to operate with GMRP-aware end stations only. It is largely unsuitable for operation on any LAN that contains GMRP-unaware devices, unless the network administrator is prepared to manually configure the GARP administrative control parameters of all such Ports to meet the known requirements of all connected GMRP-unaware devices.

An alternative approach would be to include on any shared media LAN that contained GMRP-unaware devices a “proxy member” that joined an appropriate set of Groups on behalf of the end stations on its LAN. However, the network administrator must again be prepared to manually configure the “proxy member” in order for this to work, so this solution is little different from manually configuring the Filtering Databases.

In general, therefore, it would appear to be inappropriate to design a migration strategy around the use of Filter Unregistered Groups, unless the physical configuration of the LAN allows segregation of GMRP-aware and GMRP-unaware end stations.

I.1.2.5 Use of a common set of addresses

A major danger in migration from GMRP-unaware environments to GMRP-aware environments is that, if both types of end stations are recipients of a common set of MAC Addresses, there is the possibility that the GMRP-unaware end stations will become disenfranchised as a result of the GMRP-aware end stations registering membership of a Group with that MAC Address. How likely this is to happen depends greatly upon the environment concerned. Given the fact that the use of the GMRP registration mechanisms will, in many cases, be tightly coupled to new applications and software implementations, the real impact of this problem may prove to be smaller in practice than it might appear to be in theory.

However, given the danger, any migration environment needs to be analyzed in order to determine whether the problem actually exists. Strategies for addressing the problem are again likely to revolve around the approaches discussed in I.1.2.2, I.1.2.3, and I.1.2.4, namely:

a) Segregation of legacy end stations on Forward All Groups mode LANs.
b) Manual configuration of the Filtering Database.
c) Use of “proxy member” end stations.

I.2 Interoperability with higher-layer multicast protocols and related issues

It is clearly necessary for GMRP to coexist and interoperate with devices in a Bridged Local Area Network that may implement higher-layer multicast protocols. The primary example here will be an IP Router that supports IP multicast and IGMP (IP Group Management Protocol); however, other examples may be encountered that exhibit similar characteristics.

Although not a higher-layer protocol issue per se, it may also be necessary to accommodate devices that monitor the multicast traffic in a network, which rely for their operation on promiscuous reception.

I.2.1 IP multicast

An IP router needs to receive multicast frames sent to any of the addresses in the multicast address range designated for IP multicast use.

A workable approach is to ensure that all Bridge Ports connected to the same LAN as the router adopt a default Group filtering behavior of Forward All Groups. This can be achieved either by static configuration of the Filtering Database for the Ports concerned, or by the router issuing a REGISTER_SERVICE_REQUIREMENT primitive, specifying Forward All Groups, to its GMRP Participant, which in turn will send a GMRP PDU containing a JoinIn or JoinEmpty message (12.9.4.2) for Forward All Groups registration. For all Bridge Ports whose default Group filtering behavior is Forward Unregistered Groups or Filter Unregistered Groups that are attached to the same LAN (including any Bridges indirectly attached, via Bridges providing only Basic Filtering Services), this has the effect of forcing them to change their default Group filtering behavior to Forward All Groups. This has the effect of ensuring that all Bridges in the network will forward multicast frames in the direction of the Router, regardless of whether those frames are destined for registered Groups. This approach will work regardless of the number of routers that may exist in the Bridged Local Area Network; however, it has the characteristic that the LAN that the router is attached to will “see” all multicast traffic generated anywhere in the network, regardless of whether it is of interest to the router.

Ideally the router would be connected to a Bridge by a LAN with only GMRP-unaware end stations attached to it, thus ensuring that the GARP-aware devices in the network obtain the maximum benefit from the Group filtering services. It may also be appropriate to attach this LAN to a Bridge that is close to the center of the network, as indicated in the discussion of legacy devices in I.1.1.

GMRP-aware end stations involved in IP multicast reception must be capable of using GMRP to manage their membership of Groups as required by their use of IP multicast. For example, in order to join a given IP multicast Group using IGMP, the end station must perform a GMRP Join for that Group before issuing the IGMP Join.

NOTE—Although described in the context of IP multicast, the approach described here may be appropriate for routers implementing other protocol architectures that support multicast.

I.2.2 Monitoring multicast traffic

The addition of the dynamic multicast filtering capability defined in this standard to a Bridged Local Area Network means that network monitors that would hitherto have been aware of all multicast traffic in the network (other than traffic constrained by static filters) may now be unaware of some multicast traffic, depending upon the point of attachment of the monitoring device and the pattern of Group registration that exists at that point in the network. The situation for multicasts will therefore be very similar to the situation for unicasts, where the dynamic entries in the Filtering Database results in some unicast traffic not being visible on some LANs.

Should it be desired to monitor all multicast traffic in the network, the simplest approach, which results in reception of all traffic except traffic destined for Groups for which static filtering information prevents membership, is for the monitor to ensure that all Bridge Ports connected to the same LAN as the monitor adopt a default Group filtering behavior of Forward All Groups, as described in I.2.1.

The ideal configuration for this scenario is either that the monitor is connected to a LAN that is already served by Ports that have adopted Forward All Groups filtering behavior, or that it is connected to a Bridge by a LAN with no other end stations, Bridges, or routers connected to it. It may also be appropriate to attach this LAN to a Bridge that is close to the center of the network, as indicated in the discussion of legacy devices in I.1.1.

It should be noted that unless the monitor is attached to a LAN where all Bridge Ports serving the LAN normally adopt Forward All Groups behavior (for example, a legacy LAN, as discussed in I.1.1), this approach may result in changes to the normal distribution of multicast frames in the network (i.e., changes to the distribution that would occur in the absence of the filtering mode changes introduced by the monitor itself).

Annex J

(informative)

RSTP Migration

J.1 Overview of protocol changes

RSTP (Clause 17) is explicitly designed to be compatible with the Spanning Tree Algorithm and Protocol (STP), as specified in Clause 8 of IEEE Std 802.1D, 1998 Edition and prior revisions of the standard. Computation of the Spanning Tree is identical between STP and RSTP. Protocol changes include:

a) Definition of a new Protocol Version number (version 2) for use with RSTP.
b) Definition of a new BPDU type (BPDU type 2) to distinguish RST BPDUs from Configuration and Topology Change BPDUs.
c) Inclusion of the Port Roles (Root Port, Designated Port, and Backup Port) in the computation of Port State (Discarding, Learning, and Forwarding). In particular, a new Root Port transitions rapidly to Forwarding.
d) Signalling to neighboring Bridges of a Bridge Port’s desire to be Designated and Forwarding, and explicit acknowledgement by the neighboring Bridge on a point-to-point link. This allows the Port State to transition to Forwarding without waiting for a timer expiry.
e) Acceptance of messages from a prior Designated Bridge even if they conveyed “inferior” information. Additionally, a minimum increment to the Message Age is specified so that messages propagating in this way cannot “go round in circles” for long.
f) Improvements in the propagation of topology change information so that the information does not have to be propagated all the way to the Root Bridge and back before unwanted learned source address information is flushed from the Filtering Databases.
g) Origination of BPDUs on a Port by Port basis, instead of transmission on Designated Ports following reception of information from the Root.

In addition to the changes to the state machines described in Clause 17, the following are required in order to support these changes:

h) Revised specification of timer values to accommodate changed behavior in the cases where neighboring Bridges do not support RSTP, and the forward delay timers do actually run to completion. The default timer values are chosen to work well; however, some care may be needed in environments where timers have been tuned to their minimum values.
i) Detection of point-to-point links (see 6.4.3) to allow selection of the procedures to indicate “Designated wanting to become Forwarding” (referred to as “propose” and “proposed” in the state machines) and “Yes, go ahead” (referred to as “agree” and “agreed” in the state machines). The adminPointToPointMAC and operPointToPointMAC parameters (6.4.3) allow point-to-point links to be identified.
j) Specification of BPDU message formats that include the information necessary to signal designated indications and confirmations.

J.2 BPDU formats

The BPDU formats used in IEEE Std 802.1D, 1998 Edition and prior versions of the standard were designed to permit easy extensions to spanning tree protocol(s). The basis for the intended backwards compatibility was that an implementation of version X of the protocol should interpret version >X BPDUs as if they were version X, ignoring any parameters and/or flags added by the more recent version, and interpret version <=X BPDUs exactly as specified for the version concerned. For this to work, new versions of the protocol were allowed to add new parameters and/or flags, but not to redefine the semantics or encoding of any of the parameters and flags of previous versions. Adoption of this approach would lead a correctly implemented version 0 device to ignore the protocol version field altogether, and also to ignore any parameters and/or flags that were not part of the version 0 protocol specification.

Unfortunately, while the 1998 and prior revisions of IEEE Std 802.1D are correctly specified in this regard, the interpretation of the words in the standard has not been consistent; consequently, there are implementations of spanning tree protocol that will discard BPDUs that do not carry protocol version 0, or that carry additional flags over and above those specified for version 0. The wording in Clause 9 has been made more explicit to ensure that this problem is not repeated in future implementations.

To ensure interoperability between RSTP Bridges and version 0 (STP) Bridges, it has therefore been necessary not simply to define a new protocol version number and additional parameters, but also to allow an RSTP Bridge to detect the presence of an STP Bridge on each Port and use version 0 BPDUs (see the BPDU Migration State Machine in 17.24). If an STP Bridge is detected, the information necessary to transition a Designated Port to Forwarding rapidly cannot be exchanged; however, the key element of rapidly transitioning new Root Ports to Forwarding is retained.

NOTE—The protocol version for RST BPDUs is version 2; version 1 BPDUs (defined for IEEE Std 802.1G, now withdrawn) are accommodated by a place-holder of zero length for version 1 protocol information.

The operation of the BPDU Migration State Machine is as follows. If a new Bridge is added to a LAN, it will start by transmitting a version 2 (RSTP) BPDU. For an initial period it will accept and process any BPDU format, but reception of version 0 BPDUs will not cause it to change the BPDU format it will transmit. If all other Bridges attached to the LAN are RST Bridges, they will see the version 2 BPDU and will send version 2 BPDUs themselves, if they need to transmit. However a version 0 Bridge, if present, will persist in sending version 0 BPDUs. Any version 0 BPDU received after the initial period causes RSTP to revert to transmitting version 0 BPDUs for a time. If after this time, a version 2 BPDU is seen or there is an explicit management request to do so, RSTP again transmits version 2 BPDUs.

A likely scenario is that remaining STP Bridge Ports will be Root Ports or Alternate Ports. In this case, when a new style Designated Port checks to see if the legacy Bridges have been removed from the LAN (by sending version 2 BPDUs), the legacy Bridges will be silent until they time out the existing Root and attempt to become Designated. This will cause the RSTP Bridge to send version 0 BPDUs, and the STP Bridge Port(s) will not erroneously enter the Learning or Forwarding states.

One subtlety of the chosen approach arises in the case of an STP Bridge that discards BPDUs on the analysis of the Flags field (the “Agreed” case in the RST BPDU sets a new flag not defined in version 0). This would cause such an STP Bridge to attempt to become Designated, as in the earlier scenario, allowing the RSTP Bridge to detect its presence and to revert to sending version 0 BPDUs at an early stage. This is preferable to discovering that version 2 BPDUs are discarded only in times of significant network change.

It should be noted that the approach chosen allows the determination of BPDU type to be made on a per-Port basis, allowing an RSTP Bridge to use version 0 BPDUs on some Ports, version 2 BPDUs on others.

Annex K

(informative)

Frame duplication and misordering

Although the Rapid Spanning Tree Protocol provides a single loop free path between communicating stations at any instant, RSTP can reconfigure the active topology of the Bridged Local Area Network in less time than it can take a frame to transit from its source to its destination(s). This annex explains how this can result in user data frame duplication and misordering and examines the possible QoS impact.

NOTE—The ForceVersion parameter provided by RSTP allows network administrators to choose slower reconfiguration, thereby trading off service availability against the risk of duplication and misordering.

K.1 Background

In a network of Bridges implementing the Spanning Tree Protocol (STP), specified in the 1998 and earlier editions of this standard, delays in transitioning Ports to Forwarding ensure that frames in transit prior to a network reconfiguration are delivered or discarded before being forwarded on the new active topology. In such a network, the only source of misordered or duplicated frames is a “magically healed” LAN between two Bridges, for example, as a result of accidental interconnections between shared media hubs.

RSTP, by design, can transition a Port to Forwarding very rapidly. A Root Port transition can occur as rapidly as the hardware can manage it, and a Designated Port transition in the time it takes to transmit two frames from separate stations on a single LAN. It is conceivable that a reconfiguration can take place, and a Port be made Forwarding as a result, while frames forwarded on the prior active topology are still in transit.

K.2 Frame duplication

A unicast frame whose destination address has been learned by the Bridges that can forward it, can only be buffered for transmission on one Port of one Bridge at any one time, and cannot be duplicated.

A unicast frame whose destination address has not be learned, can be flooded by some Bridges, and therefore buffered on multiple outbound Ports at the same time, and can be duplicated during network reconfiguration. Figure K-1 provides an example. In this network fragment, the Root Bridge is assumed to be somewhere to the left of Bridge A’s Port A1, and the Port Path Costs of the three Bridges result in the active topology shown, with Port B3 Discarding and the remaining Ports all Forwarding.

If the configuration is stable, a unicast frame arriving at A1 and destined for an unlearned unicast destination reachable through B2 is flooded by Bridge A to its Ports A2 and A3, and by Bridge C to its Ports C2 and C3. As B3 is Discarding, only the frame reaching B1 is transmitted through B2, and the destination receives a single copy of the frame.

During reconfiguration the following sequence of events can occur:

a) A receives the frame through A1 and transmits copies of it through A2 and A3.
b) B and C each receive a copy and queue them for transmission through B2, C2, and C3.
c) B2 transmits its copy of the frame to its destination.
d) B detects that LAN A2-B1 has failed, and immediately makes B3 Forwarding as its new Root Port.
e) C2 transmits its copy of the frame, B receives and transmits this second copy through B2.

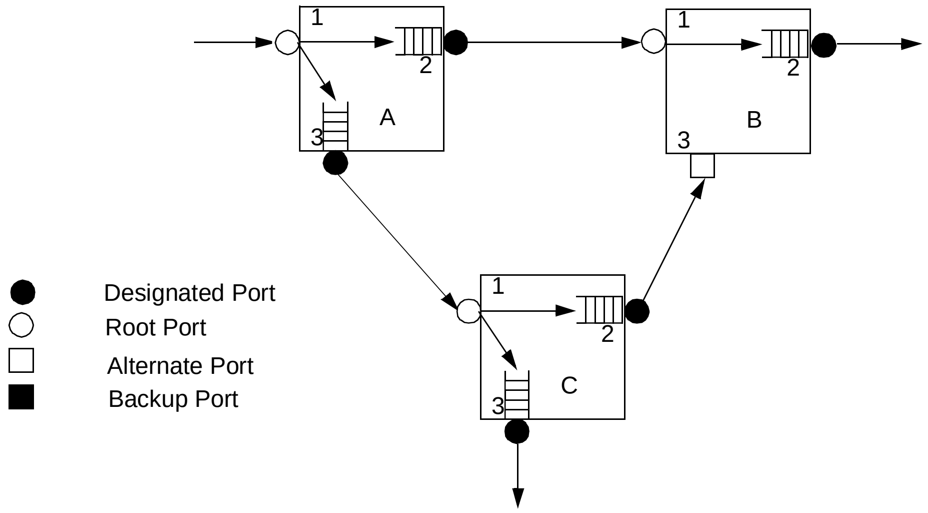


Figure K-1—Frame duplication scenario

The destination station, reachable through B2, receives the same frame twice. However, any subsequent response from the destination station would cause the address to be learned. While this reconfiguration will itself cause addresses to be flushed from the Bridge's filtering databases of Bridges, the flushing commences after the Port becomes Forwarding, and will not increase frame duplication unless there are further changes in the active topology.

Multicast frames are all potentially subject to duplication on reconfiguration events, as they can be buffered at multiple outbound Ports. Frame duplication does not cause problems in known multicast LAN protocols.

The risk of frame duplication depends upon network configuration, frequency of reconfiguration, equipment implementation details, and the characteristics of the network traffic that determine how long each frame is likely to be buffered in each bridge, and therefore cannot be quantified without reference to the a particular network.

K.3 Frame misordering

A change in the active topology between two communicating end stations can result in frame misordering, as a frame sent after reconfiguration can experience a lower transit delay, being queued at fewer Bridge Ports or for less time. Figure K-2 provides an example. In this network fragment a station connected to A1 is communicating with a station connected to B2. Prior to reconfiguration, A2 is an Alternate Port and is Discarding; all traffic between the two stations is relayed by Bridge C. During reconfiguration the following sequence of events can occur:

a) Frame F1 is received through A1, buffered and transmitted through A3, received through C1 and buffered awaiting transmission through C2.
b) A detects that LAN A3-C1 has failed, and immediately makes A2 Forwarding as its new Root Port.
c) Frame F2 is received through A1, transmitted through A2 to B1, and then transmitted through B2.
d) F1 is transmitted through C2 to B3, and then transmitted through B2.

Clearly, the receiving station connected to B2 sees the frames in the reverse of the order in which the originating station transmitted them.

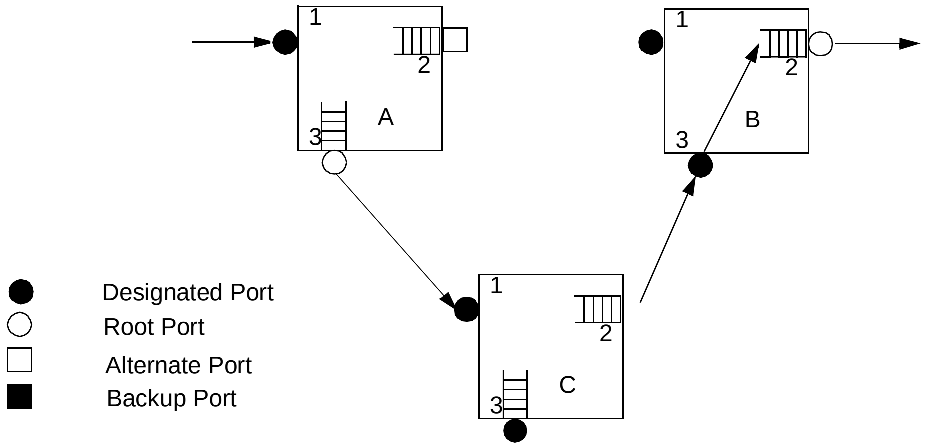


Figure K-2—Frame misordering scenario

As with frame duplication, the overall risk of frame misordering depends upon network configuration, frequency of reconfiguration, equipment implementation details, and the characteristics of the network traffic that determine how long each frame is likely to be buffered in each bridge, and therefore cannot be quantified without reference to the a particular network.

Some LAN protocols, for example LAT, LLC2, and NETBEUI, are sensitive to misordering, so even a low incidence of misordering could result in perceived problems in networks that support these protocols.

K.4 Other considerations

The possibility that frames may be stored “in transit” on old routes after a reconfiguration has completed means that there is also a possibility that addresses will be mis-learned, leading to a risk of denial of service for the addresses concerned. Hence, it is necessary to run short aging timers for a period after a reconfiguration to allow such frames to be delivered or discarded.

The Bridge model described in IEEE Std 802.1D/IEEE Std 802.1w does not include the concept of queueing on input to a Port; however, practical Bridge designs generally include some input queueing. While this is not a solution to the effects described above, a Bridge should flush any input queue associated with a Port that becomes Disabled. This behavior is consistent with the Bridge model; if a frame is in an input queue, it has not been received as far as the Bridge Port is concerned, and therefore should not be received after the Port becomes Disabled.