IEEE *Draft* P802.17/D0.1

Part 17: Resilient Packet Ring Access Method & Physical Layer Specifications—

Media Access Control (MAC) Parameters, Physical Layer Interface, and Management Parameters

Sponsor

LAN MAN Standards Committee
of the
IEEE Computer Society

This standard defines the medium access control characteristics, physical layer interface methods and layer management parameters for the Resilient Packet Ring (RPR) access method for shared physical or logical ring topologies. A set of protocols for detecting and initializing the shared ring configuration, recovering from failures, and regulating fair access to the shared medium are also described. Specifications are provided for interface to a number of physical layers, comprising 1 Gb/s, 2.488 Gb/s, 9.953 Gb/s and 10 Gb/s rates. System considerations and management information base (MIB) specifications are also set down herein.

This draft, D0.1, is being circulated for review and comment by the P802.17 Working Group. The formal expiration date of this draft is March 18, 2002.

Copyright © 2002 by the Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue
New York, NY 10016-5997, USA

All rights reserved. This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to change. USE AT YOUR OWN RISK! Because this is an unapproved draft, this document must not be utilized for any conformance/compliance purposes. Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only. Prior to submitting this document to another standards development organization for standardization activities, permission must first be obtained from the Manager, Standards Licensing and Contracts, IEEE Standards Activities Department. Other entities seeking permission to reproduce this document, in whole or in part, must obtain permission from the Manager, Standards Licensing and Contracts, IEEE Standards Activities Department.

IEEE Standards Activities Department
Standards Licensing and Contracts
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA

List of Special Symbols.

Editor's Note: *to be removed prior to final publication*
For the benefit of those who have received this document by electronic means, what follows is a list of special symbols and operators. All special symbols and operators are taken from the "SYMBOL" font set supported on most common computer systems. If any of these symbols or operators fail to print out correctly on your machine, the editors apologize, and hope that this table will at least help you to sort out the meaning of the resulting funny-shaped blobs and strokes.

Special symbols and operators

Printed Character	Meaning	Frame 6.0 character code	Font
∗	Boolean AND	ALT-042	Symbol
+	Boolean OR, Arithmetic addition	ALT-043	Symbol
^	Boolean XOR	^	Times
!	Boolean NOT	ALT-033	Symbol
<	Less than	ALT-060	Symbol
≤	Less than or equal to	ALT-0163	Symbol
=	Equal to	ALT-061	Symbol
≠	Not equal to	ALT-0185	Symbol
≥	Greater than or equal to	ALT-0179	Symbol
>	Greater than	ALT-062	Symbol
⇐	Assignment operator	Ctrl-q \	Symbol
∈	Indicates membership	ALT-0206	Symbol
∉	Indicates nonmembership	ALT-0207	Symbol
±	Plus or minus (a tolerance)	Ctrl-q 1	Times
°	Degrees (as in degrees Celsius)	ALT-0176	Symbol
∑	Summation	ALT-0229	Symbol
—	Big dash (Em dash)	Ctrl-q Shft-q	Times
–	Little dash (En dash)	Ctrl-q Shft-p	Times
†	Dagger	Ctrl-q Space	Times
‡	Double dagger	Ctrl-q ‘	Times
µ	Micro	Ctrl-q 5	Times
Ω	Omega	ALT-087	Symbol
λ	Lambda	ALT-0108	Symbol

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Contents

1. Overview 1

1.1 Scope 1
1.2 Purpose 2
1.3 Third-party operation 2
1.4 Services 3
1.5 Network Properties 3
1.6 Resilience 7
1.7 Spatial reuse 7
1.8 Bandwidth management features 9
1.9 Fairness 9
1.10 Jitter and Delay Considerations 10
1.11 Physical Layer Independence 10
1.12 Application Areas 10
1.13 Conformance Requirements 10
1.14 Notation 10

2. Normative references 13

3. Terms and definitions. 17

4. Abbreviations and acronyms 29

5. Medium Access Control (MAC) reference and service model 31

5.1 Scope 31
5.2 Overview of MAC Services 31
5.3 MAC Peer-to-Peer Services 33
5.4 MAC services to the MAC Client Layer 34
5.5 Physical layer service access point (PSAP) definition 38
5.6 Management layer service access point 39
5.7 MAC Reference model 41

6. Medium Access Control 45

6.1 Transit/Transmit buffer 45
6.2 Transmit and forwarding operation 46
6.3 Receive operation 48
6.4 Transit operation 50
6.5 Circulating packet detection (stripping) 50
6.6 Wrapping of data 51
6.7 Pass-thru mode 51

7. MAC physical interface 53

7.1 Overview 53
7.2 SONET/SDH network 53
7.3 Ethernet 54

8. Frame formats 57

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.1 Overview ... 57
8.2 RPR packet header format ... 58
8.3 Overall packet format ... 59
8.4 RPR control packet format ... 61
8.5 RPR Fairness Frame Format ... 62
8.6 Order of bit transmission ... 62
8.7 Invalid RPR frame ... 62
8.8 Elements of a tagged RPR frame ... 63
8.9 Elements of RPR frame with customer separation ID ... 64

9. MAC fairness ... 67

9.1 Overview ... 67
9.2 Congestion detection ... 69
9.3 Interoperability between single/dual transit buffer MACs ... 70
9.4 Threshold settings ... 70
9.5 RPR fairness packets ... 71

10. Topology Discovery ... 75

11. Protection ... 77

11.1 Wrap protection ... 77
11.2 Steering protection ... 80
11.3 Multicast consideration ... 81
11.4 Protection hierarchy ... 81
11.5 Protection message packet format ... 82
11.6 RPR protection protocol states ... 84
11.7 Protection protocol rules ... 84
11.8 RPR protection protocol rules ... 86
11.9 Protection state transition ... 88
11.10Failure examples ... 89

12. Operation administration and maintenance (OAM) ... 95

12.1 Overview ... 95
12.2 Fault management ... 97
12.3 RPR Activation/Deactivation of OAM ... 99
12.4 OAM frame handling during failures ... 100
12.5 OAM frame ... 101
12.6 OAM frame detection procedure ... 106
12.7 OAM frames support ... 106

13. Layer Management Entity Interface ... 107

13.1 Overview of the management model ... 107
13.2 Generic management primitives ... 108
13.3 MLME SAP interface ... 110
13.4 Ring Aggregation ... 115
13.5 PLME SAP interface ... 116

Annex A ... 119

Annex B ... 121

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex C 123

Annex D 157

Annex E 185

Annex F 227

Annex G 235

Annex H 237

Annex I 253

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

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

Part 17: Resilient Packet Ring Access Method & Physical Layer Specifications

1. Overview

Editor's Note: *To be removed prior to final publication.*
This clause could generally be made more useful by providing a more complete review of the RPR architecture and how components described in other clauses interact, and less emphasis on comparison of RPR to other technologies.

Editor's Note: *To be removed prior to final publication.*
The following content should be added to the overview:

1) service class descriptions:
2) unicast and multicast operation
3) shortest path selection for unicast traffic
4) OAMP suitable for service provider deployment
5) dynamic topology discovery
6) congestion reporting to MAC client
7) Qtag identifier to support RPR VLAN
8) customer separation identifier (CID)

Editor's Note: *To be removed prior to final publication.*
The term 'frame' and 'packet' are used interchangeably throughout the overview. There still needs to be some agreement on which one to use when.

1.1 Scope

Editor's Note: *To be removed prior to final publication.*
The phrase 'protocol and compatible interconnection' does not adequately define the scope of this specification. Other parts of IEEE 802 specify scope in more detail. The term 'proposal' is not applicable to the standard. Language should be improved as the meaning of 'protocol..... of data communication equipment' is not clear.

This proposal defines the protocol and compatible interconnection of data communication equipment via a ring-topology local and metropolitan area network using resilient packet ring access method.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1.2 Purpose

> ***Editor's Note:*** *To be removed prior to final publication.*
> *The phrase 'fiber optic and copper' is inconsistent with the specification of 'fiber' throughout this clause. The use of either 'fiber' or 'copper' is inconsistent with physical layer independence. The term 'interconnect' is not clear (e.g. is this the same as medium?). The language LAN/MAN/RAN/WAN is awkward. 'Support for three traffic classes' is vague without some specification of what distinguishes these classes. 'large number of stations' is vague. 'Ring master negotiation' is not defined. SONET BLSR is not defined (and steering is not similar to BLSR). These features of the protocol could be defined more precisely or the section could be omitted as it is not related to the specification.*

The purpose of this protocol is to provide a scalable LAN/MAN/RAN/WAN architecture with shared access method, spatial re-use, and resiliency through fault protection method. Pursuant to this, the protocol will:

a) Support a minimum data rate of 155Mb/s, scalable to higher speeds.
b) Support for dual counter rotating ring over fiber optic and copper interconnects.
c) Efficient use of bandwidth by the use of spatial reuse and minimal protocol overhead.
d) Support for three traffic classes.
e) Scalability across a large number of stations attached to a ring
f) "Plug and play" design without a software based station management transfer (SMT) protocol or ring master negotiation as seen in other ring based MAC protocols.
g) Weighted fairness among stations using the ring (each station can be assigned a proportion of the ring bandwidth).
h) Support for ring based redundancy (error detection, ring wrap, etc.) similar to that found in SONET BLSR specifications.
i) Provide media independent service interface from MAC to PHY layer.

1.3 Third-party operation

> ***Editor's Note:*** *To be removed prior to final publication.*
> *It is unclear how these three types of operation are related to the specification. It is not indicated what technical accommodations must be made to support the three modes of operations.*

The RPR standard deals with ring networks that can scale from LAN to WAN environments. These networks could be operated by private organizations over their own infrastructure, private organizations over the facilities of a third party service provider or the network of a third party service provider as a public service. All three types of operations are supported.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Reference to 802.16 is not permitted. Statement indicating that authentication and encryption are not necessary may be more appropriately placed in the scope section. The relationship between 'authentication and encryption' and 'third party operation' is not clear.*

Within the IEEE 802 family, most of the standards, such as the various versions of Ethernet, apply to local area networks, owned and operated by a single organization. In other cases, such as the 802.16 Broadband Wireless Access standard, a third-party operator is assumed. In that case, the open nature of the wireless medium makes it necessary to provide a high level of authentication and encryption, in order to restrict access to authorized users and to guarantee privacy of the transmissions.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *The meaning of 'impacted' is unclear. Does this mean that traffic generated by a user never affects the traffic of any other user, independent of service class? This is a strong restriction and should be made clearer if intended.*

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

In the case of RPR, the need for such security facilities is less, given that the difficulty of physical access to wireline facilities provides security absent in wireless systems. However, it is important to third-party service providers to be able to provide guarantees of service through service level agreements (SLAs). The level of service provided to one user, in terms of throughput, delay, and other characteristics, should not be impacted by the actions of other users.

Thus this standard provides for a variety of classes of service and provides mechanisms to prevent users from degrading the service seen by others

1.4 Services

Editor's Note: *To be removed prior to final publication.*
The text of this section describes applications rather than services. A section describing the MAC service interface should be added.

The applications that are expected to use an RPR network encompass the complete range of networking applications, including the familiar set of voice, video, and data. The resilient ring is ideal for critical applications that require high availability; since there are two paths between any two points on the ring, failure of a link need not prevent applications from running.

Typical uses of the network would include Internet access; in this case most traffic would exit from the RPR ring into a wide-area network or WAN for delivery to distant locations. Other uses would be for virtual private networks (VPNs) where bandwidth guarantees might be established to provide the equivalent of a hard-wired private line but with the cost savings associated with the ability of the network to recycle unused bandwidth to other users, or to allow a customer to burst at higher rates on an occasional basis

1.5 Network Properties

1.5.1 Network scale

Editor's Note: *To be removed prior to final publication.*
Scale issues appropriate for the standard would be the maximum length of a span and number of spans on a ring.Describing places where the MAN might be installed does not appear to be related to the standard itself. The statement 'relatively little loss of responsiveness' is vague. The current RPR architecture is not 'fail-save', it can recover from one fault.

RPR technology is optimized for the needs of metropolitan networks, in other words on a scale larger than a LAN but smaller than a typical WAN. Circumferences ranging up to several hundred kilometers are likely to be the most common. However, this does not preclude its use in other situations, for example as a building or campus backbone. It may also be used with ring circumferences ranging into thousands of kilometers with relatively little loss of responsiveness, should the fail-safe or bandwidth management properties of the network make it attractive to do so.

Editor's Note: *To be removed prior to final publication.*
'wide range' is vague. Is it possible to state upper and lower bounds based on some property of the architecture. 'DWDM fiber' (lambda?) seems misplaced here. The other protocols would make use of a lambda.

The data rates that can be accommodated by the RPR design cover a wide range. The protocols are designed to operate over a variety of physical layers, including SONET/SDH, Gigabit Ethernet (IEEE 802.3ab), and DWDM fiber. As higher-speed physical layers become available, it is expected that RPR will be able to work over them as well.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1.5.2 Topologies

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Redundant, all ring topologies have a ring structure. The term 'fortuitous loop' is not clear. A ring is a ring.*

This standard is intended for network configurations that have a ring topology, where there is a well-defined ring structure that offers two possible paths from any source to any destination. This is in contrast to a fortuitous loop in the cabling configuration, which is generally avoided in networking practice except as backup.

1.5.2.1 Dual ring

RPR uses a bidirectional ring. This can be seen as two symmetric counter-rotating ringlets. Most of the protocol finite state machines (FSMs) are duplicated for the two ringlets.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *In UPSR it is the destination station that selected the preferred signal. None of these acronyms are explained here, is it necessary to introduce these comparisons, especially if they are loose?*

The bidirectional ring allows for two protection mechanisms to be implemented in case of media or station failure, Ring wrap as in FDDI or SONET/SDH BLSR or source steering as in SONET UPSR, where the source station selects which ringlet will carry the packet.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *There is no explanation here of how 'inner' and 'outer' are assigned. On the ring, there is no physical notion of 'clockwise' and 'counterclockwise'. Are individual links configured as belonging to 'inner' or 'outer'? More explanation is needed.*

To distinguish between the two ringlets, one is referred to as the "inner" ringlet, the other the "outer" ringlet. The RPR protocol operates by sending data traffic in one direction (known as "downstream") and its corresponding control information in the opposite direction (known as "upstream") on the opposite ringlet.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *The term 'dual' add-drop multiplexer is not in common use and it's not clear that this analogy is needed in the specification. The term 'station' should always be used instead of 'station' and some instances have been changed here.*

The stations are able to send data on either ringlet, in other words clockwise or counter-clockwise. Generally the shorter of the two possible paths to a given destination is used, based on the topology discovery scheme, but this is not required, due for instance to congestion or malfunction on one of the links. The structure of an individual station can be thought of as a dual add-drop multiplexer, as shown for a dual ring in Figure 1–1. The inputs to the switch include:

— Receive from upstream
— Local transmission queues

Outputs are:

— Packets for this station
— Transmit downstream

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editor's Note: *To be removed prior to final publication.*
Figure 1 missing.

Note that multicast and broadcast packets are copied to the local receiver and transmitted downstream.

Figure 1–1—Switching functions at a dual-ring station.

Editor's Note: *To be removed prior to final publication.*
The title above uses the terminology 'dual-ring station'. Why is this terminology used? Aren't all RPR stations dual ring-stations.

The overall structure of the system is as shown in Figure 1–2.

Editor's Note: *To be removed prior to final publication.*
Figure 2 is missing.

Figure 1–2—Dual-ring MAN; individual stations are as shown in Figure 1–1.

Editor's Note: *To be removed prior to final publication.*
The next paragraph is a discussion of the advantages of a dual-ring with shortest path routing. It is not appropriate for the body of the specification and should be remove to a note or an annex, or deleted. 'Single fiber' is medium specific.

The use of a dual ring has substantial advantages in terms of the capacity of the system. Since there are two paths to each destination, it is possible to send traffic over the shorter path rather than the longer one; hence the longest path taken by any packet is half the ring circumference. If the destinations are randomly distributed along the ring, then the average path length is half of this, or a quarter of the circumference. Given that packets are stripped at the destination station (See "Spatial reuse" on page 7.) extensive spatial re-use is possible, with the total capacity of the dual ring approaching 8 times that of a single fiber.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1.5.2.2 Shared medium

> ***Editor's Note:*** *To be removed prior to final publication.*
> *The first sentence is only true for multicast/broadcast traffic. This information is not needed in the specification. It says that RPR supports a broadcast mode, which has been said elsewhere. The header reads 'shared medium' but the text is about 'broadcast medium'. Discussion of upper layer protocols is outside the scope of this standard.*

An RPR ring acts as a broadcast medium, in which a single transmission is capable of reaching all stations on the network. This means that the large number of mechanisms that have come into use with broadcast networks such as Ethernet will still work with RPR. Examples of these include the ARP protocol, the spanning tree protocol (IEEE 802.1D), and Layer 3 protocols in general.

Even under protection-switching conditions, where the rings operate as a bidirectional bus, the broadcast property still holds. Protection switching is an internal function of the RPR system and does not impact higher layers.

1.5.3 Packet-based operation

> ***Editor's Note:*** *To be removed prior to final publication.*
> *There is no 'current networking practice'. Choice of byte, cell, frame,or packet depends on many factors. It is not clear that this furthers the discussion.*

The basic unit of data on the RPR ring is a packet, consistent with current networking practice. This follows the precedent of the IEEE 802.5 token-passing ring and the FDDI ring. It does differ, however, from the 802.6 distributed queue dual bus ring, which was based on cells, and SONET/SDH rings, which operate on a TDM basis with 8-bit granularity.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1.6 Resilience

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Is it 'resilience' or 'resiliency'. We should be consistent.Replace 'fiber' with 'link' or 'span' and 'equipment' by 'station'. What is the meaning of 'standby status'? The sentence 'During normal operation, all paths carry traffic' is difficult to interpret. Paths go from source station to destination station. Under normal conditions a path is used only when there is traffic from a source to a particular destination and the selected path is the shorter of the two alternative. So, it's not clear that all paths carry traffic.*

The redundancy provided by two or more paths to the same destination provides resiliency in the face of fiber or equipment failure. This standard provides resiliency without the need to assign half of the fiber links to standby status. During normal operation, all paths carry traffic.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Additional clarification is needed here. The ring continues to function properly only in the face of a single failure. Throughput cannot be degraded for committed bandwidth.*

On the occurrence of network faults, due either to link failure equipment malfunction, the RPR system can continue to function. While throughput may be degraded, all functioning stations continue to operate on the ring.

1.6.1 Source steering

Given that the ring provides two routes to any destination, if one is not operational, the other can be used. The source MAC simply must be aware of the existence of the failure; for the duration of the outage it can send all data over the operational path.

1.6.2 Wrapping protection

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Terms and definitions currently uses the term 'segment' instead of 'span'. Span may be better.*

A RPR ring is composed of two counter-rotating ringlets. If an equipment or fiber facility failure is detected, traffic going towards and from the failure direction is wrapped (looped) back to go in the opposite ringlet. Wrapping takes place on the stations adjacent to the failure to re-route the traffic away from the failed span

1.7 Spatial reuse

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Not the 'aggregate bandwidth', but the 'bandwidth utilization'. Again, comparison to other standards is probably not necessary.*

Spatial Reuse is a concept used in rings to increase the overall aggregate bandwidth of the ring. This is possible because unicast traffic is only passed along ring spans between source and destination stations rather than the whole ring as in earlier ring based protocols such as token ring and FDDI.

Figure 1–1 below outlines how spatial reuse works. In this example, station 1 is sending traffic to station 4, station 2 to station 3 and station 5 to station 6. Having the destination station strip unicast data from the ring allows other stations on the ring which are downstream to have full access to the ring bandwidth. In the

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

example given this means station 5 has full bandwidth access to station 6 while other traffic is being simultaneously transmitted on other parts of the ring.

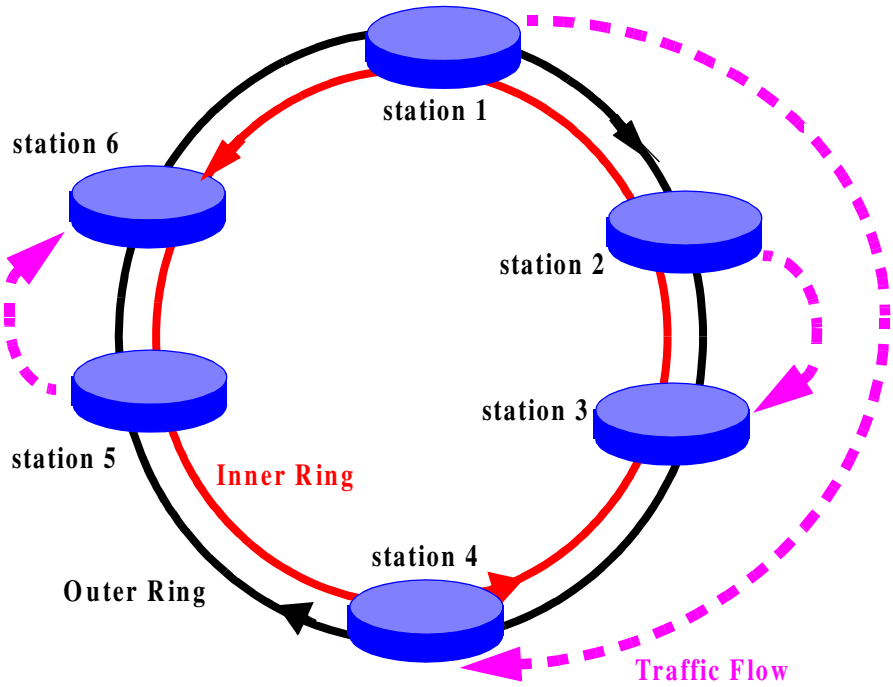


Figure 1–1—Global and [illegible]al Reuse

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Cannot use color figures.*

1.7.1 Destination stripping

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Destination removal of frames is called 'destination stripping', not 'spatial reuse'. Destination stripping allows spatial reuse. Again, the contrast to other protocols is not appropriate in the specification except in a note or annex.*

RPR achieves a high degree of link utilization by having the destination station remove packets from the ring. This is called spatial reuse because it provides empty space on the ring which that station or another station down the fiber can use to send additional data. This is in contrast to the 802.5 Token-passing Ring or FDDI, in which packets which have already been received continue on the ring until they are removed by their original source station.

There is a provision for dealing with failed or non-existent destinations. A time-to-live or hop count field in the header makes it possible to detect packets circulating excessively and to remove them.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1.7.2 Multicast traffic

> ***Editor's Note:*** *To be removed prior to final publication.*
> *This is a chapter on multicast traffic, but there are not corresponding sections on broadcast and unicast traffic. Also, this sections appears to include broadcast.*

An exception to destination stripping is made for multicast and broadcast packets. Since there is no single destination to remove the packet, stripping is done by the original sender.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Also remove broadcast frames in the same way.*

While the time-to-live field is decreased to zero the multicast packet also will be stripped from ring.

1.8 Bandwidth management features

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Stations don't cooperate to maximize traffic. They cooperate to maximize utilization of available bandwidth. This section contained many instances of the term 'node' that have been changed to 'station'.*

In RPR systems, stations cooperate to maximize traffic on the rings. They advertise information about current loads so that other stations may determine how much traffic they can send through that station.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *If it is the 'heart' of the system, that's a good reason to provide a thorough overview here. What is an RPR 'system'?*

The bandwidth management algorithm is the heart of the RPR system; it is described in detail in Clause 11 and 12.

1.9 Fairness

> ***Editor's Note:*** *To be removed prior to final publication.*
> *'is' may be to strong. The ring behaves like a shared medium.*

Since the ring is a shared media, some sort of access control is necessary to ensure fairness and to bound latency. Access control can be broken into two types which can operate in tandem:

Global access control - controls access so that everyone gets a fair share of the global bandwidth of the ring.

Local access control - grants additional access beyond that allocated globally to take advantage of segments of the ring that are less than fully utilized.

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Hard to parse the sentence below. What does it mean that global and local access are required?*

As an example of a case where both global and local access are required, refer again to Figure 1–1. stations 1, 2, and 5 will get 1/2 of the bandwidth on a global allocation basis. But from a local perspective, station 5

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

should be able to get all of the bandwidth since its bandwidth does not interfere with the fair shares of stations 1 and 2

1.10 Jitter and Delay Considerations

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Section is missing text.*

1.11 Physical Layer Dependence

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Paragraph below unnecessary limits to fiber. Is PLCP the same as RS? Which are we using?*

The RPR design allows for wide latitude in the choice of physical layers. Various fiber-based layers, such as Gigabit Ethernet and SDH/SONET may be used. This standard includes Physical Layer Convergence Procedures (PLCP) to specify how RPR packets are carried over the physical layer, and how control information is passed between the layers. It is expected that new PLCP standards may be generated in the future as new physical layers come into use.

1.12 Application Areas

> ***Editor's Note:*** *To be removed prior to final publication.*
> *Services section already describes applications. This information should probably be placed in the scope, if it is needed at all.*

The applications environment for the resilient packet ring network is intended to be commercial and light industrial.

1.13 Conformance Requirements

> ***Editor's Note:*** *To be removed prior to final publication.*
> *PICS Proforma will be placed within the clause to which it is associated.*

Annex A will contain PICS Preform definitions for resilient packet ring network components.

1.14 Notation

This standard uses service primitives, finite state machines, state tables, and pseudo code, supplemented by prose descriptions and illustrative diagrams, to define the requirements of the protocol.

1.14.1 Service definition method and notation

The service of a layer or sublayer is the set of capabilities that it offers to a user in the next higher (sub)layer. Abstract services are specified here by describing the service primitives and parameters that characterize each service. This definition of service is independent of any particular implementation (see Figure 1–1).

Specific implementations may also include provisions for interface interactions that have no direct end-to-end effects. Examples of such local interactions include interface flow control, status requests and indications, error notifications, and layer management. Specific implementation details are omitted from this ser-

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

vice specification both because they will differ from implementation to implementation and because they do not impact the peer-to-peer protocols.

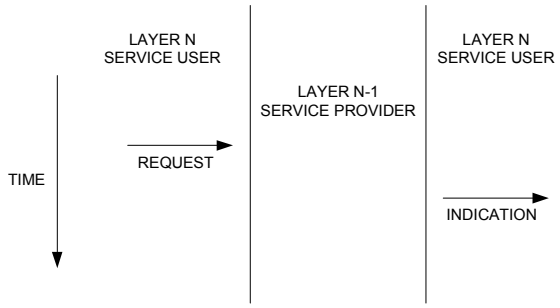


Figure 1–1—Service Definition

1.14.1.1 Classification of service primitives

Primitives are of two generic types:

a) REQUEST. The request primitive is passed from layer N to layer N-1 to request that a service be initiated.
b) INDICATION. The indication primitive is passed from layer N-1 to layer N to indicate an internal layer N-1 event that is significant to layer N. This event may be logically related to a remote service request, or may be caused by an event internal to layer N-1.

The service primitives are an abstraction of the functional specification and the user-layer interaction. The abstract definition does not contain local detail of the user/provider interaction. For instance, it does not indicate the local mechanism that allows a user to indicate that it is awaiting an incoming call. Each primitive has a set of zero or more parameters, representing data elements that shall be passed to qualify the functions invoked by the primitive. Parameters indicate information available in a user/provider interaction; in any particular interface, some parameters may be explicitly stated (even though not explicitly defined in the primitive) or implicitly associated with the service access point. Similarly, in any particular protocol specification, functions corresponding to a service primitive may be explicitly defined or implicitly available.

1.14.2 State diagram notation

The operation of a protocol can be described by subdividing the protocol into a number of interrelated functions. The operation of the functions can be described by state diagrams. Each diagram represents the domain of a function and consists of a group of connected, mutually exclusive states. Only one state of a function is active at any given time (see Figure 1–1)

Each state that the function can assume is represented by a rectangle. These are divided into two parts by a horizontal line. In the upper part the state is identified by a name in capital letters. The lower part contains the name of any ON signal that is generated by the function. Actions are described by short phrases and enclosed in brackets.

All permissible transitions between the states of a function are represented graphically by arrows between them. A transition that is global in nature (for example, an exit condition from all states to the IDLE or RESET state) is indicated by an open arrow. Labels on transitions are qualifiers that must be fulfilled before the transition will be taken. The label UCT designates an unconditional transition. Qualifiers described by short phrases are enclosed in parentheses.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

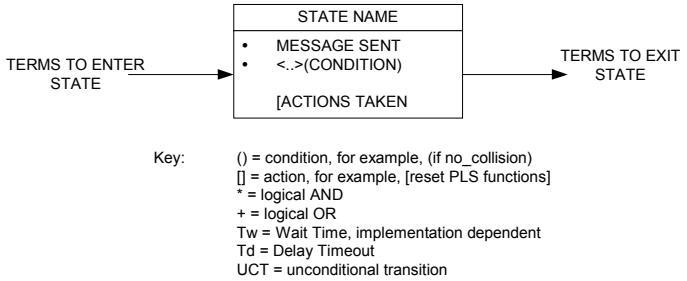


Figure 1–1—-State diagram notation example

State transitions and sending and receiving of messages occur instantaneously. When a state is entered and the condition to leave that state is not immediately fulfilled, the state executes continuously, sending the messages and executing the actions contained in the state in a continuous manner.

Some devices described in this standard are allowed to have two or more ports. State diagrams that are capable of describing the operation of devices with an unspecified number of ports, required qualifier notation that allows testing for conditions at multiple ports. The notation used is a term that includes a description in parentheses of which ports must meet the term for the qualifier to be satisfied (e.g., ANY and ALL). It is also necessary to provide for term-assignment statements that assign a name to a port that satisfies a qualifier. The following convention is used to describe a term-assignment statement that is associated with a transition:

a)) The character "**:**" (colon) is a delimiter used to denote that a term assignment statement follows.
b)) The character "**<=**" (left arrow) denotes assignment of the value following the arrow to the term preceding the arrow.

The state diagrams contain the authoritative statement of the functions they depict; when apparent conflicts between descriptive text and state diagrams arise, the state diagrams are to take precedence. This does not override, however, any explicit description in the text that has no parallel in the state diagrams.

The models presented by state diagrams are intended as the primary specifications of the functions to be provided. It is important to distinguish, however, between a model and a real implementation. The models are optimized for simplicity and clarity of presentation, while any realistic implementation may place heavier emphasis on efficiency and suitability to a particular implementation technology. It is the functional behavior of any unit that must match the standard, not its internal structure. The internal details of the model are useful only to the extent that they specify the external behavior clearly and precisely.

Editor's Note: *To be removed prior to final publication.*
Description of the format of C code examples and of the PICS Proforma should be included.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

2. Normative references

Editor's Note: *To be removed prior to final publication.*
The references currently cited in this clauses do not correspond to actual references made by the clauses. Each clause will specify in an editor's note, applicable references, including all standard bibliographic information and method by which the document can be obtained.

The following standards contain provisions which, 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 IEC and ISO maintain registers of currently valid international standards.

[**Editor's note**: The following references are lifted from other 802 specifications, and suitably modified (deletions). As IEEE 802.17 includes external PHY layer references, the respective standard must be added here as a normative reference. This paragraph is to be kept during draft process and deleted before publication]

ANSI X3T9 FDDI Specification

[Bellcore GR-1230, Issue 4, Dec. 1998, "SONET Bidirectional Line-Switched Ring Equipment Generic Criteria".

ANSI T1.105.01-1998 "Synchronous Optical Network (SONET) Automatic Protection Switching"

Malis, A. and W. Simpson, "PPP over SONET/SDH", RFC 2615, June 1999.

Simpson, W., "PPP in HDLC-like Framing", STD 51, RFC 1662, July 1994.

ANSI/TIA/EIA-568-A, Commercial Building Telecommunications Cabling Standard.CISPR 22: 1993, Limits and Methods of Measurement of Radio Interference Characteristics of Information Technology Equipment.[3]

IEC 60060 (all parts), High-voltage test techniques.[4]

IEC 60068, Basic environmental testing procedures.

IEC 60096-1: 1986, Radio-frequency cables, Part 1: General requirements and measuring methods and Amd. 2: 1993.

IEC 60793-1: 1995, Optical fibres—Part 1: Generic specification.
IEC 60793-2: 1992, Optical fibres—Part 2: Product specifications.
IEC 60794-1: 1996, Optical fibre cables—Part 1: Generic specification.

IEC 60794-2: 1989, Optical fibre cables—Part 2: Product specifications.

IEC 60825-1: 1993, Safety of laser products—Part 1: Equipment classi.cation, requirements and user's guide.

IEC 60825-2: 1993, Safety of laser products—Part 2: Safety of optical fibre communication systems.

IEC 60874-1: 1993, Connectors for optical fibres and cables—Part 1: Generic specification.

IEC 60874-10: 1992, Connectors for optical fibres and cables—Part 10: Sectional specification, Fibre optic connector type BFOC/2.5.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

IEC 60950: 1991, Safety of information technology equipment.

IEC 61000-4-3, Electromagnetic Compatibility (EMC)—Part 4: Testing and measurement techniques—Section 3: Radiated, radio-frequency, electromagnetic field immunity test.

IEC 61754-4: 1997, Fibre optic connector interfaces—Part 4: Type SC connector family.

IEEE Std 802-1990, IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture.[5]

IEEE Std 802.1F-1993 (Reaff 1998), IEEE Standards for Local and Metropolitan Area Networks: Common Definitions and Procedures for IEEE 802 Management Information.

IEEE P802.1Q/D11 (July 30, 1998), Draft Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks.[6]

IETF RFC 1155, Structure and Identification of Management Information for TCP/IP-based Internets, Rose, M., and K. McCloghrie, May 1990.[7]

IETF RFC 1157, A Simple Network Management Protocol (SNMP), Case, J., Fedor, M., Schoffstall, M., and J. Davin, May 1990.

IETF RFC 1212, Concise MIB Definitions, Rose, M., and K. McCloghrie, March 1991.

IETF STD 17, RFC 1213, Management Information Base for Network Management of TCP/IP-based internets: MIB-II, McCloghrie K., and M. Rose, Editors, March 1991.

IETF RFC 1215, A Convention for Defining Traps for use with the SNMP, M. Rose, March 1991

IETF RFC1662: PPP in HDLC Like Framing, W. Simpson, July 1994

IETF RFC 1901, Introduction to Community-based SNMPv2, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

IETF RFC 1902, Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2), Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

IETF RFC 1903, Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2), Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

IETF RFC 1904, Conformance Statements for Version 2 of the Simple Network Management Protocol

(SNMPv2), Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

IETF RFC 1905, Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2), Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

IETF RFC 1906, Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2), Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, January 1996.

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

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

IETF RFC 2271, An Architecture for Describing SNMP Management Frameworks, Harrington, D., Presuhn, R., and B. Wijnen, January 1998.

IETF RFC 2272, Message Processing and Dispatching for the Simple Network Management Protocol (SNMP), Case, J., Harrington D., Presuhn R., and B. Wijnen, January 1998.

IETF RFC 2273, SNMPv3 Applications, Levi, D., Meyer, P., and B. Stewart, January 1998.

IETF RFC 2274, User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3), Blumenthal, U., and B. Wijnen, January 1998.

IETF RFC 2275, View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), Wijnen, B., Presuhn, R., and K. McCloghrie, January 1998.

IETF RFC2615:PPP over SONET/SDH, A. Malis, W. Simpson, June 1999

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.[8]

ISO/IEC 2382-9: 1995, Information technology—Vocabulary—Part 9: Data communication.

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

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

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

ISO/IEC 8825: 1990, Information technology—Open Systems Interconnection—Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1).

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

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

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

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

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

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

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

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

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

ISO/IEC 11801: 1995, Information technology—Generic cabling for customer premises.

ISO/IEC 15802-2: 1995 [ANSI/IEEE Std 802.1B-1992 and IEEE Std 802.1k-1993], Information technology —Telecommunications and information exchange between systems—Local and metropolitan area networks —Common specifications—Part 2: LAN/MAN Management.

ISO/IEC 15802-3: 1998 [IEEE Std 802.1D, 1998 Edition), Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Common specifications —Part 3: Media Access Control (MAC) bridges.[9]

ITU-T Recommendation G.957 (1995) Digital line systems—Optical interfaces for equipments and systems relating to the synchronous digital hierarchy.[10]

ITU-T Recommendation I.430 (1995), Basic user-network interface—Layer 1 specification.

System Packet Interface Level 3 (SPI-3): OC48 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI3-01.0

System Packet Interface Level 4 Phase 1 (SPI-4.1): OC192 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI4-01.0

System Packet Interface Level 4 Phase 2 (SPI-4.2): OC192 System Interface for Physical and Link Layer Devices. Implementation agreement: OIF-SPI4-02.0

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3. Terms and definitions.

Editor's Note: *To be removed prior to final publication.*
Terms not used in the final version of the standard will be removed by the time of ballot on that document
An update to this clause, including previous comment resolutions, will be presented at the March plenary.

3.1 802.17: See IEEE Std. 802.17.

3.2 agent: [802.3-2000 1.4.30 (modified)] A network management entity (NME) which can be used to configure the station and/or collect data describing operation of that station.

3.3 <all-stations MAC address[1]: TBD >

3.4 backpressure: The sending of a *control frame* in the *upstream* direction, to stop or slow the flow of *data traffic*.

3.5 <bandwidth: Note: The term *bandwidth* is applicable to the *physical layer* and should **not** be used in reference to the *MAC layer.* The term *data-rate or capacity* is used instead.>

3.6 best-effort service (BES): A service not providing any *QoS* guarantee.

3.7 bit error ratio (BER): [802.3-2000 1.4.47] The ratio of the number of bits *received* in error to the total number of bits *received.*

3.8 bit rate: [ISO/IEC2382-09 9.03.01 (modified)] The speed at which bits are transferred.

3.9 bridge: [IEC2382-25 25.01.12 (modified)] A functional unit that interconnects two networks that use the same logical link control protocol but may use different *medium access control protocols*. *Local area networks (LANs)* and *metropolitan area networks (MANs)* are example of *networks* that a *bridge* may interconnect.

3.10 bridged network: [(C/LM) 10038-1993, 802.1G-1996 (modified)] A concatenation of individual *networks* interconnected by *MAC bridges.*

3.11 broadcast address: [ISO/IEC2382-25 25.01.13] A *group address* that identifies the set of all *stations* on the *network*.

3.12 broadcast: [802.5-1998 1.3.10] The act of sending a *frame* addressed to all *stations*.

3.13 buffer insertion ring (BIR): An *access technique* for ring media that gives absolute priority to passthru traffic except when transmission of an ingress frame is in progress.

3.14 buffer: An area of memory used for temporary storage of *frames*.

3.15 bursty (burstiness): Characterization of traffic as using the maximum data-rate of a channel only intermittently.

3.16 capacity: The maximum *data-rate* supported by a *medium* or *channel*.

3.17 channel: see *transmission channel*.

[1] Brian Holden to investigate.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.18 class of service (CoS): The categorization of traffic according to *relative* delivery priority.

3.19 closed user group (CUG): [09.08.14, 610.7-1995] A specified group of *network* users who are permitted communications among themselves but not with other *network* users.

3.20 committed burst size (Bc): [ITU I.233.1 A.5 (modified)] The maximum amount of data (in bits) that the *network* agrees to *transfer*, under normal conditions, during a time interval *Tc*.

3.21 committed information rate (CIR): [ITU I.233.1 A.8 (modified)] The information transfer rate which the network is committed to transfer under normal conditions. The rate is averaged over a minimum interval of time (*Tc*).

3.22 committed rate measurement interval (Tc): [ITU I.233.1 A.7 (modified)] The time interval during which the user is allowed to send only the committed amount of data (*Bc*) and the excess amount of data (*Be*).

3.23 congestion avoidance: [ITU I.233.1 A.11 (truncated)] Procedures initiated at or prior to the onset of mild congestion in order to prevent congestion from becoming severe.

3.24 congestion control: [ITU I.233.1 A.9] Real-time mechanisms to prevent and recover from congestion during periods of coincidental peak *traffic* demands or *network* overload conditions (e.g. resource failures). *Congestion control* includes both *congestion avoidance* and *congestion recovery* mechanisms.

3.25 congestion management: [ITU I.233.1 A.10] This includes *network* engineering, OAM procedures to detect the onset of congestion, and real-time mechanisms to prevent or recover from congestion. Congestion management includes, but is not limited to, *congestion control*, *congestion avoidance*, and *congestion recovery.*

3.26 congestion recovery: [ITU I.233.1 A.12 (truncated)] Procedures initiated to prevent congestion from severely degrading the end user perceived *quality of service*(s) delivered by the *network*.

3.27 congruent ringlets: *Ringlets* that share the same set of *stations*, but a distinct set of *links*, such that the order of station traversal via the links is identical or is exactly reversed.

3.28 control frame: A *frame* carrying only *MAC sublayer* control information.

3.29 control latency: Interval between the time that a *control frame* is sent from a *station* and the time that the effect of that *control frame* is observable at the issuing *station*.

3.30 conversation: [IEEE 100 (C/LM) 802.3 ad-2000] A set of *MAC frames transmitted* from one *end station* to another, where all of the *MAC frames* form an ordered sequence, and where the communicating *end stations* require the ordering to be maintained among the set of *MAC frames* exchanged.

3.31 copy (copying): Replication of an *inbound frame* by the *MAC sublayer* (independent of whether or not the *frame* is *stripped*).

3.32 customer separation: The property that data associated with one group of network users (e.g. a customer organization) is not communicated to a different group of *network* users.

3.33 <cut-thru: The *passthru* of a *frame* through a *station* such that the first bit of the *frame* is *retransmitted* before the last bit is *received*[2].>

3.34 cyclic redundancy check (CRC): A form of error check used to ensure the accuracy of *transmitting* a message. Note: The *CRC* is the result of a calculation carried out on the set of *transmitted bits* by the *trans-*

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

mitter. The *CRC* is encoded into the *transmitted signal* with the data. At the *receiver*, the calculation creating the *CRC* may be repeated, and the result compared to that encoded in the signal.

3.35 data delivery ratio (DDR): [FRF.13 section 5 (modified)] Reports the network's effectiveness in transporting offered data. The DDR is a ratio of successful payload octets received to attempted payload octets transmitted.

3.36 data frame: A *MAC frame* carrying data supplied by the *MAC client*.

3.37 data-rate: The rate at which information is transferred, measured in bits-per-second.

3.38 <data-stream (stream): [(C) 610.10-1994 (modified)] A continuous stream of data elements being *transmitted*.>

3.39 delivered duplicated frames: [ITU I.233.1 A.15 (modified)] A *frame received* at a *destination* such that the frame was not generated by the *source station* identified by the *source address* and the *frame* is exactly the same as a *frame* that was previously delivered to that *destination*.

3.40 delivered errored frames: [ITU I.233.1 A.14 (modified)] The number of *frames* for which the value of one or more of the bits in the *frame* is in error, or when some, but not all, bits in the *frame* are lost bits or extra bits (i.e. bits that were not present in the original signal).

3.41 delivered out-of-sequence frames: [ITU I.233.1 A.16 (modified)] A frame (Ft) arriving at a *destination station* after a *frame* F_{t+1}, F_{t+2}, F_{t+3}Ö., F_n in a sequence of *frames* F_1, F_2, F_3, Ö. , F_n sent from a *source station*.

3.42 destination station (destination): [from IEEE 100 (C/BA) 1355-1995 (modified)] The *station(s)* on a network that is(are) the intended recipient(s) of an *802.17 frame*.

3.43 discard eligibility (DE): [FRF glossary (modified)] A bit indicating that a *frame* may be discarded in preference to other frames if congestion occurs, in order to maintain the committed *quality of service* within the *network*.

3.44 downstream: The direction of data flow.

3.45 dual-ring: A *ring* composed of exactly two *congruent ringlets* having opposite orientations.

3.46 1.46 <dynamic bandwidth allocation: Candidate definition should be proposed. The T&D Ad Hoc noted that some people think of this as a synonym for fairness.>

3.47 1.47 <effective transfer rate: [ISO/IEC2382-09 9.05.22 (modified)] The number of bytes *transferred* between two points per unit time and accepted as valid at the *destination*.>

3.48 egress queue delay: TBD by Performance Ad Hoc

3.49 egress stripping: The removal of *frames* by the *egress station*.

3.50 egress: The direction towards the *MAC client* from the *ring* or *MAC sublayer.*_

3.51 encapsulation: A process by which an entity places a *header* and, optionally, a *trailer* on an *SDU*.

---

[2] LAN switches typically perform cut-thru of 802.3 frames after reception of the destination *MAC address* (first six bytes of the *frame*). LAN switch cut-thru is not described in 802.1D but is left as a device specific feature. The frame check is not performed for frames that are cut-thru.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.52 <encapsulating bridge[3]:> (TBD)

3.53 end station: [802.3-2000 1.4.111 (modified)] A station attached to a *network* that is an initial source or a final destination of MAC frames transmitted across that *network*. A network layer router is, from the perspective of the *LAN*, an *end station*; a *MAC Bridge*, in its role of *forwarding MAC frames* from one *LAN* to another, is not an *end station*

3.54 excess burst size (Be): [ITU I.233.1 A.6 (modified)] The maximum amount of data by which a user can exceed *Bc* during a time interval *Tc*. This data (*Be*) is delivered in general with a lower probability than *Bc*.

3.55 fairness: The assignment of *ring ingress rates* such that available *capacity* is shared according to a specified algorithm.

3.56 flow control: A *congestion control* mechanism allowing one *station* to communicate to another *station* the information that *frame transmission* should be reduced or halted in order to avoid *buffer* overrun, or other conditions associated with congestion, at the *receiving station* and allowing the resumption of normal levels *frame transmission* when the condition is resolved.

3.57 flow: The collection of *frames* associated with a *conversation* that can be identified by one or a combination of specific values carried in the *protocol headers* at the *MAC layer* or above.

3.58 rame check sequence (FCS): [IEEE 100 (C/LM) 802.12-1995] *A Cyclic Redundancy Check (CRC)* used by the *transmit* and *receive* algorithms to detect errors in the bit sequence of a *MAC frame*

3.59 frame delivery ratio (FTR): [FRF.13 section 4] The ratio of successful *frame receptions* to attempted *frame transmissions*.

3.60 frame transfer delay (FTD): [FRF.13 section 3] The difference in milliseconds between the time a *frame* exits a *source station* and the time the same *frame* enters the *destination station*.

3.61 frame transmission time: TBD by Performance Ad Hoc

3.62 frame: (see *MAC frame*)

3.63 global fairness: TBD by working group.

3.64 global spatial reuse: The utilization of *ring capacity* by *stations* on the *ring* when the *station* to which the *capacity* is assigned does not utilize that *capacity.*

3.65 group address: [ISO/IEC2382-25 25.01.15] An *address* that identifies a group of *stations* on a *network*.

3.66 guaranteed-service (GS): Service that assures conformance to specific QoS parameter values.

3.67 IEEE Std. 802.17 (802.17): The IEEE *resilient packet ring* standard.

3.68 inbound: The direction of *frame* arrival at a *station* from a *ringlet*.

3.69 individual address: [ISO/IEC2382-25 25.01.14] An *address* that identifies a particular *station* on a *network*.

[3] Bob Castellano to investigate.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.70 ingress queue delay: TBD by Performance Ad Hoc

3.71 ingress rate control: *Rate control* performed at the *ring ingress*.

3.72 ingress stripping: The removal of *frames* by the *ingress station*.

3.73 ingress: The direction from the *MAC client* towards the *ring* or *MAC sublayer.*

3.74 insert[4] **(insertion):** The placement of an *ingress frame* on the *ring* by a *station*.

3.75 interconnected rings: Non-*congruent rings* that intersect at one or more *stations*.

3.76 jitter: Variation in *delay* associated with the *transfer* of *frames* from one point in the *network* to another.

3.77 latency: The time required for information to be *transferred* between two points. Synonymous with *delay* for purposes of the 802.17 specification.

3.78 link aggregation group[5]**:** [IEEE 802.3-2000 1.4.154 (modified)] A group of *links* that appear to a *MAC client* as if they were a single *link*. All *links* in *a link aggregation group* connect the same pair of aggregation systems. One or more *conversations* may be associated with each *link* that is part of a *link aggregation group*.

3.79 link partner: The device at the opposite end of a *link* from the local *station*.

3.80 link: [IEEE 100 (C/LM) 802.5c-1991] A unidirectional physical and media connection between two *stations*.

3.81 local area network (LAN): [adapted from IEEE 100 (C/DIS) 1278.2-1995, 1278.3-1996[6]] A communications network designed for a user premises, typically not exceeding a few kilometers in length, and characterized by moderate to high data transmission rates, low delay, and low bit error rates.

3.82 local fairness: TBD by working group.

3.83 local spatial reuse: The utilization of common *ring capacity* by *stations* communicating across non-overlapping *segments* of the *ring*.

3.84 logical link control (LLC) sublayer: [C/LM 8802-5-1992s] That part of the *data link layer* that supports *media* independent *data link* functions, and uses the services of the *MAC sublayer* to provide services to the *network layer.*

3.85 lost frames: [ITU I.233.1 A.17 (modified)] A *frame* not delivered to the intended *destination* user within a specified time-out period, and the *network* is responsible for the non-delivery.

3.86 MAC client: The *protocol layer* (or *sublayer*) immediately above the *MAC sublayer.* Generally, the *network layer* or *logical link control (LLC) sublayer.*

3.87 MAC end-to-end delay: TBD by Performance Ad Hoc

[4] (C/LM) 11802-4-1994 is not applicable. That definition uses the term to specify device, rather than frame, insertion in the ring..
[5] 802.17 link aggregation is not necessarily identical to that specified by 802.1ae.
[6] 'moderate sized geographic area' replaced by 'user premises, typically not exceeding a few kilometers in length,'

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.88 MAC frame (frame): [IEEE 100 (C/LM) 802.12-1995] The logical organization of control and data fields (e.g., addresses, data, error check sequences) defined for the *MAC sublayer*[7]. Note: The term *frame* can be prefixed with an orientation (*ingress*, *egress*, *inbound*, *outbound*) or an operation (*inserted*, *copied*, *stripped*, *passedthru*).

3.89 management information base (MIB): [802.3-2000 1.4.163] A repository of information to describe the operation of a specific network device.

3.90 maximum frame size (MFS): The maximum number of bytes in a *frame*.

3.91 maximum transfer unit (MTU): [IEEE 100 610.7-1995 (modified)] The largest *payload* that can be *transferred* across a given *physical network* in a single *frame*.

3.92 medium: See *transmission medium*.

3.93 medium access control (MAC) sublayer: (1) [IEEE 100 (C/LM) 8802-5-1995] The portion of the *data link sublayer* that controls and mediates the access to the *ring*. **(2)** [802.3-2000 1.4.167] The *data link sublayer* that is responsible for *transferring data* to and from the *physical layer*[8]. **(3)** [ISO/IEC 15802-1] The MAC service provider.

3.94 medium access delay: TBD by Performance Ad Hoc

3.95 <medium[9] agnostic: Denotes a *MAC sublayer* that can operate with arbitrary *physical layer* alternatives, requiring a *reconciliation sublayer* for each specific *PHY* type supported.>

3.96 <medium interface connector (MIC): [802.5-1998 1.3.36 (modified)] A connector interface at which signal transmit and receive characteristics are specified for attaching stations.>

3.97 metropolitan area network (MAN): [IEEE 100 (C/LM) 8802-6-1994] A *network* for connecting a group of individual *stations* and *networks* [for example, *local area networks (LANs)*] located in the same urban area. Note: A *MAN* generally operates at a higher speed than the networks interconnected, crosses network administrative boundaries, may be subject to some form of regulation, and supports several access methods.

3.98 misdelivered frames: [ITU I.233.1 A.16 (modified)] A *frame transferred* from a *source* to a *destination* user other than the intended *destination* user. It is considered inconsequential whether the information is correct or incorrect in content.

3.99 <MTU transparency: The ability to *passthru frames* without regard to *MTU* size.>

3.100 multicast address: [ISO/IEC2382-25 25.01.16]. A *group address* that identifies a subset of the *stations* on a *network.*

3.101 multicast: The act of sending a *frame* addressed to a group of *stations*.

3.102 multi-ring: A *ring* composed of multiple *congruent ringlets*, at least two of which are *opposing ringlets*.

3.103 neighbor: [BH] A *station* that is exactly one *link* away from a given *station*.

---

[7] Omitted sentence: ìThe MAC frame may be constructed in either ISO/IEC 8802-3 or ISO/IEC 8802-5 format.

[8] This appears as a definition of medium access control, but it is clearly a definition of medium access control sublayer.

[9] Standards documents are split on whether this is medi*a* or medi*um*. Medium, the singular, seems more appropriate and is used here.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.104 network control host: [802.3-2000 1.4.176] A network management central control center that is used to configure *agents*, communicate with *agents*, and display information collected from *agents*.

3.105 network: A generic designation for a bridged LAN, MAN, RAN, or WAN.

3.106 opposing ringlet: A *ringlet* whose traffic circulates in the direction opposite that of a given *ringlet*.

3.107 outbound: The direction of *frame* departure from a *station* to a *ringlet*.

3.108 packet: A *frame* to which has been added those fields that are medium dependent[10].

3.109 <packetization delay: TBD by Performance Ad Hoc>

3.110 partition: One set of communicating *stations* on a *partitioned ring*.

3.111 partitioned ring: A *ring* having two or more points of failure resulting in two or more non-communicating sets of *stations*.

3.112 passthru buffer delay: TBD by Performance Ad Hoc

3.113 passthru delay[11]: TBD by Performance Ad Hoc

3.114 passthru queuing: A method of *passthru* in which *passthru traffic* is queued to allow *passthru* or *insertion* of *traffic* of higher priority and to allow a *transmission* in progress to complete.

3.115 passthru: The passing of a *frame* through a *station* via the *ring*[12].

3.116 path: The specific sequence of *stations* and *links* traversed by a *frame* in *passthru* between two *stations*.

3.117 <pause: [802.3-2000 1.4.209 (modified)] A mechanism associated with the *IEEE 802.3 MAC* specification for providing full duplex *flow control*.>

3.118 <payload agnostic: Denotes a *MAC sublayer* that is not sensitive to the contents of the *payload transferred* to/from the *MAC client*.>

3.119 physical layer (PHY): [(C/LM) 8802-5-1995] The layer responsible for interfacing with the *transmission medium*. This includes conditioning signals received from the *MAC* for *transmitting* to the *medium* and processing *signals received* from the *medium* for sending to the *MAC*.

3.120 plug-and-play: The requirement that a *station* perform *passthru*, *strip*, and *ring* control activities without manual intervention except for what may be needed for connection to the *ring*. The station may additionally *copy* and *insert frames*.

3.121 <port: (1) The point of *ingress* for *inbound frames* and the point of *egress* for *outbound frames* with respect to the *data station*. **(2)** [IEEE 100 (C/LM) 802.1G-1996, 8802-5-1995 (modified)] A signal interface provided by stations that is generally terminated at a *medium interface connector (MIC)*.[13]>

---

[10] This is the IEEE view of a packet. It is entirely different from the IEEE view of a packet as an L3 PDU.

[11] The term *station* is used to qualify the term *transit-delay* since the term *transit-delay* is used by *frame relay* to indicate *end-to-end transit delay.*

[12] Includes the case of wrapping, if supported.

[13] Removed 'Ports may or may not provide physical containment of channels' What, exactly, does this mean?

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.122 <preemption: The interruption of a *frame* in *transmission* for the purpose of *transmitting* a *frame* of higher priority.>

3.123 propagation delay: TBD by Performance Ad Hoc

3.124 <protocol agnostic: Denotes a *MAC sublayer* that can operate with arbitrary *upper-layer protocol* alternatives.>

3.125 protocol data unit (PDU): [802.5-1998 1.3.46] Information delivered as a unit between peer entities that contains control information and, optionally, data.

3.126 protocol implementation conformance statement (PICS): 1.3.47: A statement of which capabilities and options have been implemented for a given Open Systems Interconnection (OSI) protocol.

3.127 <protocol stack delay: TBD by Performance Ad Hoc>

3.128 <QTag prefix: [802.3-2000 1.4.222] The first four octets of an Ethernet-encoded Tag Header. The Ethernet-encoded Tag Header is defined in IEEE P802.1Q.>

3.129 quality of service (QoS): One or a combination of measurable properties (parameters) defining the requirements of a given data service.

3.130 rate control: Limitation of the *traffic rate* in bytes over a specified time interval.

3.131 receive (receipt, reception): The action of a station taking a frame from the medium.

3.132 reconciliation sublayer (RS): [adapted from IEEE 100 (C/LM) 802.3 –1998 modified] A mapping function that reconciles the *signals* at the media independent interface (MII) to the *media access control (MAC)* – physical signaling sublayer (PLS) service definitions.

3.133 regional area network (RAN): (1) A *network* for connecting a group of individual *stations* and *networks* [for example, *metropolitan area networks (MANs)*] located in multiple contiguous urban areas. (2) A *MAN* spanning multiple urban areas.

3.134 <residual error rate: [ITU I.233.1 A.13 (modified[14])] As applied to *MAC layer* service: (1 – (total correct MAC SDUs delivered)/(total offered MAC SDUs)).>

3.135 resilient packet ring (RPR): (1) A *connectionless* ring-based *MAC protocol* as defined by IEEE 802.17, appropriate for *LAN*, *MAN*, or *RAN* [15]deployment[16]. (2) A collection of *stations* conforming to the *resilient packet ring protocol,* and the links forming the *ring*.

3.136 ring end-to-end delay: TBD by Performance Ad Hoc

3.137 ring latency: TBD by Performance Ad Hoc

3.138 ring medium: The abstraction of a *ring* as a continuous closed path *transmission medium*.

3.139 ring segment (segment): The portion of a *ring* bounded by two *stations* interconnected by one or more *links*.

---

[14] Need help with this one.

[15] Why did we exclude WAN?

[16] Or should this be specifically the protocol standardized by IEEE802?

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.140 ring topology (topology): [IEEE 100 610.7-1995 (modified)] The logical and/or physical arrangement of *stations* on a *ring*.

3.141 ring: (1) The collection of *stations* and *links* forming a *resilient packet ring*. **(2)** The set of *congruent ringlets* forming a *resilient packet ring*.

3.142 ringlet: A closed unidirectional *path* formed by an ordered set of *stations*, and the *links* interconnecting *stations*, such that each *station* has exactly one *link* entering the *station* and one *link* exiting the *station*.

3.143 <round trip propagation time: > TBD by Performance Ad Hoc

3.144 segment: (see *ring segment*)

3.145 service data unit (SDU): [802.5-1998 1.3.59] Information delivered as a unit between adjacent entities that may also contain a PDU of the *upper layer.*

3.146 service level agreement (SLA): Contract between a *network* service provider and a customer that specifies, in measurable terms, what services the *network* service provider will furnish.

3.147 shared access: The capability of two or more *stations* to share the *capacity* of the *ring medium*[17].

3.148 simple-fairness: A class of *fairness algorithm* that assigns *equal* shares of *ring capacity.*

3.149 simultaneous access: The *insertion* of *traffic* onto the *ring medium* by two or more *stations* at the same instant in time.

3.150 source station (source): The *station* that originates an 802.17 MAC *frame* with respect to a *network.*

3.151 spatial reuse: The utilization of *ring capacity* by a *station* different from the *station* to which the *capacity* was nominally assigned.

3.152 <spatial reuse protocol (SRP): A protocol, described in IETF informational RFC 2892, August 2000.>

3.153 station (data station): [IEEE 100 1073.3.1-1994, 1073.4.1-1994, 8802-5-1995 (modified)] A device that may be attached to a *network* for the purpose of *transmitting* and *receiving* information on that *network*.

3.154 station management (SMT): [802.5-1998 1.3.66] The conceptual control element of a station that interfaces with all of the layers of the station and is responsible for the setting and resetting of control parameters, obtaining reports of error conditions, and determining if the station should be connected to or disconnected from the medium.

3.155 station passthru delay: TBD by Performance Ad Hoc

3.156 steering: The *transmission* of a *frame* on a specific *ringlet* at the *ingress station* based on knowledge of the *ring topology.*

3.157 store-and-forward: A method of *passthru* such that all bits of the *frame* are *received* and *buffered* before re*transmission* begins[18].

3.158 <stream: (see *data-stream*)>

---

[17] The definition of ring latency in ISO/IEC 2382-25 25.04.03 suggests that the ring is modeled as a shared medium even if it is not a continuous physical medium.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.159 strip (stripping): The removal of a *frame* from the *ring*.

3.160 <tagged MAC frame: [802.3-2000 1.4.269] A frame that contains a QTag Prefix.>

3.161 throttle: The sending of a *control frame* to a specific *station,* to stop or slow the flow of *data traffic*.

3.162 throughput: [ITU I.233.1 A.1(modified)] The number of data bits contained in the MAC frame payload successfully transferred from *source station* to *destination station* per unit time. A frame is successfully transferred if the *FCS* check for the frame is satisfied.

3.163 time-to-live (TTL): Value carried in the *protocol header* of a *frame* in order to allow the stripping of a *frame* that has *passedthru* a sufficient number of stations. The *TTL* value is generally set to an initial value at the source and decremented at each subsequent hop. The *frame* is *stripped* when the *TTL* value reaches zero.

3.164 topology: (see *ring topology*)

3.165 traffic class: A grouping of traffic that is to be processed by a distinct set of rules.

3.166 train: A collection of two or more contiguous *frames* on the ring.

3.167 transfer: [ISO/IEC2382-09 9.03.01 (modified)] The movement of an *SDU* from one layer to an adjacent layer. Also used generally to refer to any movement of information from one point to another.

3.168 <transfer rate: [ISO/IEC2382-09 9.05.21] The number of bytes *transferred* per unit time between two points.>

3.169 transmission channel (channel): [ISO/IEC2382-9 09.02.14] A means of transmission of a signal in one direction between two stations where the signals are physically or logically isolated from the signals in other channels.

3.170 transmission medium (medium): [IEEE 100 (C/LM) 8802-6-1994, 802.5-1998 1.3.34] The material on which *information signals* may be carried; e.g., optical fiber, coaxial cable, and twisted-wire pairs.

3.171 transmission: (see *transmit*)

3.172 transmit (transmission): [(C/LM) 802.5-1989s, 8802-5-1995 (modified)] The action of a *station* placing a *frame* on the *medium*.

3.173 transparent bridging: [(C/LM) 8802-5-1995] A *bridging* mechanism in a *bridged network* that is transparent to the *end stations*.

3.174 unicast: The act of sending a *frame* addressed to a single *station*.

3.175 unknown unicast: The act of sending a *frame* addressed to a single *station,* where the location in the *network* is unknown.

3.176 upper-layers: The collection of *protocol layers* above the *data-link layer.*

3.177 upstream: The direction opposite that of the *downstream* direction.

---

[18] The definition that appears in 09.07.13 ISO/IEC 2382-9 1995, 'A mode of operation of a *data network* in which data are temporarily stored before they are *retransmitted* toward the *destination*', is ambiguous, as it is not clear whether 'data' refers to a complete *frame* or some portion of a *frame*.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

3.178 <upstream neighbor's address (UNA): [802.5-1998 1.3.77 (modified)] The address of the *station* immediately *upstream* from a given *station*.>

3.179 verified frame: [802.5-1998 1.3.79 (modified)] A valid *frame* addressed to the *station*, for which the information field has met the validity check.

3.180 virtual LAN (VLAN): [IEEE 100 (C/LM 802.1Q-1998)] A subset of the active *topology* of a *bridged local area network*. Associated with each *VLAN* is a *VLAN Identifier (VID).*

3.181 virtual medium (VMedium): A logical partition of the *network* intended to provide *customer separation*.

3.182 weighted-fairness: A class of *fairness algorithm* that allows the assignment of unequal shares of *ring capacity.*

3.183 wide area network (WAN): [IEEE 100 (C/DIS) 1278.2-1995] A communications network designed for large geographic areas. Sometimes called *long-haul* network.

3.184 wrapping: In the case of a *dual ring*, the *transmission* of a *frame* on the ringlet opposing the ringlet on which it was received.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

4. Abbreviations and acronyms

Editor's Note: *To be removed prior to final publication.*
Before ballot for the final draft, the document will be searched for acronyms. Those and only those acronyms will appear in this clause.

This standard contains the following abbreviations and acronyms:

Bc	committed burst size
Be	excess burst size
BER	bit error ratio
BES	best effort service
BETC	best effort traffic class
BIR	burst information rate
CIR	committed information rate
CoS	class of service
CRC	cyclic redundancy check
CUG	closed user group
DDR	data delivery ratio
DE	discard eligibility
EIR	excess information rate
FCS	frame check sequence
FTD	frame transfer delay
FTR	frame delivery ratio
GS	guaranteed-service
LAN	local area network
LLC	logical link control
MAC	medium access control
MAN	metropolitan area network
MFE	MAC fairness entity
MFS	maximum frame size
MIB	management information base
MTU	maximum transfer unit
PDU	protocol data unit
PHY	physical layer
PICS	protocol implementation conformance statement
QoS	quality of service
RAN	regional area network
RPR	resilient packet ring
RPR-FA	resilient packet ring fairness algorithm
RS	reconciliation sublayer
SA	source address
SDU	service data unit
SLA	service level agreement
SMT	station management
Tc	committed rate measurement interval
TTL	time-to-live
VLAN	virtual LAN
VDQ	virtual destination queue
WAN	wide area network

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

5. Medium Access Control (MAC) reference and service model

5.1 Scope

This clause specifies the services provided by the MAC sublayer and the MAC Control sublayer to the client of the MAC (see Figure 5–1). MAC clients may include the Logical Link Control (LLC) sublayer, Bridge Relay Entity, or other users of ISO/IEC LAN International Standard MAC services (see Figure 5–1). The services are described in an abstract way and do not imply any particular implementations any exposed interface.There is not necessarily a one-to-one correspondence between the primitives and the formal procedures and interfaces.

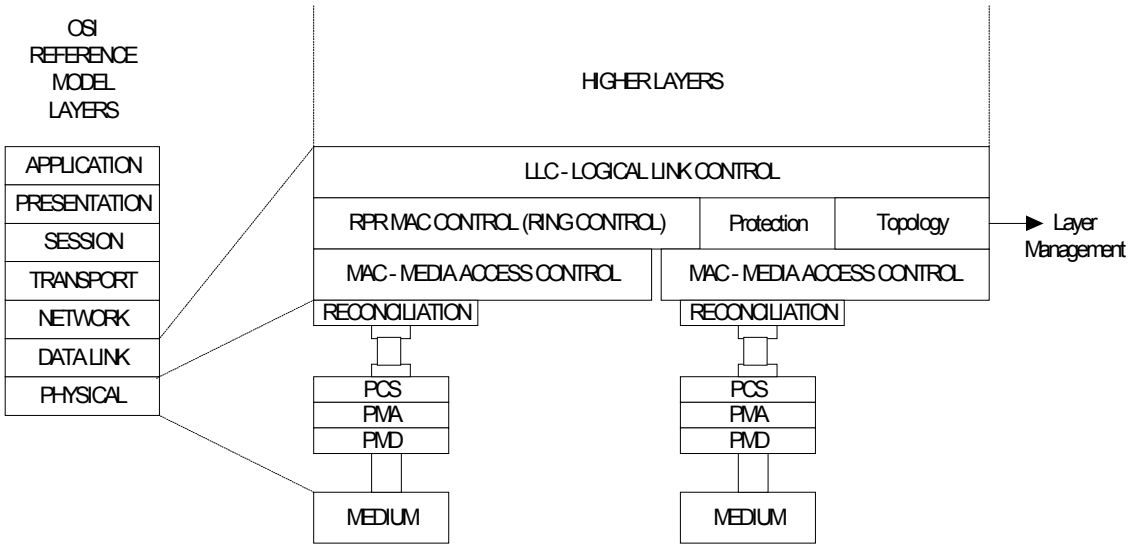


Figure 5–1—Service specification relation to the LAN model

5.2 Overview of MAC Services

The services provided by the MAC sublayer allow:

a) The local LLC sublayer in an end node to exchange data with peer LLC sublayer entities
b) The local LLC sublayer in an end node to exchange resilient packet ring parameters with local MAC entities.
c) The relay entity in a bridge to change data with local MAC entities in the bridge.
d) A non LLC MAC Client sublayer in an end node to exchange data with peer MAC client sub entities

The MAC sublayer is presents 5 service access points for the exchange of MAC client PDUs between MAC client entities. The service access points provide access to 5 logical channels at the MAC layer: reserved channel, high priority channel, medium priority channel, low priority channel and control channel.

5.2.1 Reserved Service

The 802.17 MAC provides a mechanism to reserve ring bandwidth. This bandwidth is invisible to the RPR fariness algorithm and must be wholly managed by the RPR MAC client. Traffic admitted in the other 3 traffic classes will not be able to utilized the bandwidth allocated to the Provisioned class, even when that bandwidth idle.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

By definition, if this service is offered in the RPR network, the network operator must set the provisioned bandwidth allocation consistently amongst all stations on the ring and then set allocation the provisioned bandwidth amongst nodes in the network. How this badwidth allocation is performed is the responsibility of the MAC client and beyond the scope of this specification.

The transmit data requests to the provisioned service access point are policed by the RPR MAC to meet the statically provisioned rate for this service. The service access point will also provide an indication to the MAC client of the status of the underlying channel. This information includes whether the service is currently operative (up or down).

The bandwidth allocated to the provisioned service may be set to zero allowing network designers the option of not offering such a service.

5.2.2 High Priority Service

The MAC provides a high priority delivery service. This service is intended to support application which require bounded end-to-end delay and jitter specifications.

The MAC assumes that traffic requesting high priority service will be shaped at ingress to meet provisioned values for CIR, BIR and EIR by the MAC client. The MAC sublayer implements a policing function as part of the high priority service to ensure that provisioned service parameters are not violated.

The high priority service is an engineered service and must be provisioned by the network designer.

The service access point also provides an indication to the MAC client of the status of the underlying channel. This information includes whether the service is currently operative (up or down) and whether there is dynamic backpressure from the media to indicate that traffic cannot currently be accepted.

5.2.3 Medium Priority Service

The Medium priority service is provided to implement a traffic class for applications which are not delay sensitive but which require BW guarantees.

It is similar in implementation to the High priority service, in that it expects the client to provide a shaped ingress traffic stream that conforms to provisioned CIR and EIR limits. However, traffic is treated differently with respect to the RPR-FA depending on whether it meets its CIR/EIR profile.

Those frames determined to be in-profile will be marked as such with a bit in the RPR header at ingress to the ring. These frames will not be counted as part of the RPR fairness algorithm at ingress to the ring or when transitting through stations on the ring. They are essentially invisible to the fairness algorithm in the same way that high priority packets are.

Those frames determined to be out-of-profile will be marked as such with a bit in the RPR header at ingress to the ring. Out-of profile frames will be counted as part of the RPR fairness algorithm both at ingress to the ring and while transitting stations on the ring.

Regardless of whether the frame is marked in- or out-of-profile, the frames are still sent on the low priority transit channel on the ring.

For in-profile traffic, applications using the medium priority service will not be blocked by ingress rate policing from the RPR fairness algorithm (like the high priority service) but will still incur the increased end-to-end delay of the low priority transit channel.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

For out-of-profile traffic, applications using the medium priority service receive the same service level as if they had used the low priority service access point.

The service access point also provides an indication to the MAC client of the status of the underlying channel. This information includes whether the service is currently operative (up or down) and whether there is dynamic backpressure from the media to indicate that traffic cannot currently be accepted.

5.2.4 Low Priority Service

The Low Priority service is provided to implement a Best Effort Traffic Class (BETC). It is transmitted on the MAC's Low Priority Transit Path and is intended for traffic not sensitive to end-to-end delay or jitter.

The service access point also provides an indication to the MAC client of the status of the underlying channel. This information includes whether the service is currently operative (up or down) and whether there is dynamic backpressure to indicate that traffic cannot currently be accepted.

5.3 MAC Peer-to-Peer Services

Since 802.17 MAC network is a ring based, shared medium network, each MAC provides transit service to frames that are not destined or sourced from the MAC client, hosted by that particu.lar MAC. This traffic passes through the MAC sublayer on one of 3channels: high priority, low priority or control. Note that transit packets in the Reserved class are combined with High Priority in terms of their transit treatment; likewise medium and low-priority classes are combined and treated as Low Priority.

5.3.1 High Priority Transit Channel

The MAC implements a high priority transit channel to support the High Priority traffic services. The high priority transit channel provides a worst-case per-station transit delay of one frame-time in order to bound the maximum delay for the network on the high priority service class.

The high priority transit channel does not support preemption of either the transit or ingress frames; any packet already in transmission continues to the end.

5.3.2 Low Priority Transit Channel

The MAC implements a low priority transit channel to support both medium and low priority service classes. All low priority traffic and medium priority traffic travels through the Low Priority Transit Channel on the ring.

The Low Priority Channels implements a lossless service on the ring.

5.3.3 Control Channel

The MAC implements a control channel through which control messages are passed between MAC entities on the RPR network. Traffic for this channel has the highest priority in terms of scheduling at ingress to the ring and does not participate in nor is policed by the RPR fairness mechanism.

Control traffic on the ring transit path is treated the same as traffic in the High Priority channel for the purposes of transit path scheduling and ring access (i.e. it has priority over all ingress traffic).

The MAC provides no policing of traffic destined for this channel except for inter-station synchronization purposes.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

5.4 MAC services to the MAC Client layer

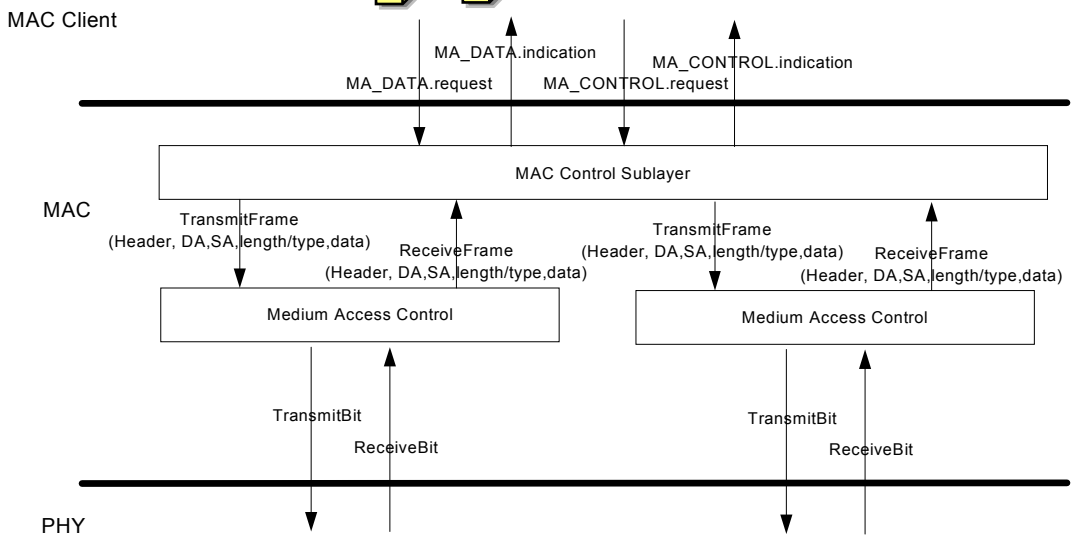


Figure 5–1—MAC service model

5.4.1 Overview of the interactions

Four service primitives are defined for the LLC interfaces.

— MA_DATA.request
— MA_DATA.indication.
— MA_CONTROL.request (used by MAC Control sublayer).
— MA_CONTROL.indication (used by MAC Control sublayer).

The formats for the M_DATA.indications and M_DATA.requests are the same as formats for MA_DATA.indication and MA_DATA.requests, except for the addition of an optional parameter for the FCS. This parameter may be used to preserve the FCS when bridging between LANs using like formats.

5.4.2 Basic services and options

The MA_DATA.request, MA_DATA.indication service, MA_CONTROL.request and MA_CONTROL.indication primitives described in this subclause are mandatory.

5.4.3 Detailed service specification

5.4.3.1 MA_DATA.request

5.4.3.1.1 Function

This primitive defines the transfer of data from a MAC client entity to a single peer entity or multiple peer entities in the case of group addresses.

5.4.3.1.2 Semantics of the service primitive

The semantics of the primitives are as follows:

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
MA_DATA.request (header,
         destination_address,
         source_address,
         m_sdu,
         service_class,
         ringlet_id)
```

The header parameter may specify one or the other ring medium, priority, Time To Live (TTL), and unicast or multicast. The destination_address parameter may specify either an individual or a group MAC entity address. It must contain sufficient information to create the DA field that is pre-appended to the frame by the local MAC sub- layer entity and any physical information. The source_address parameter, if present, must specify an individual MAC address. If the source_address parameter is omitted, the local MAC sublayer entity will insert a value associated with that entity. The m_sdu parameter specifies the MAC service data unit to be transmitted by the MAC sublayer entity. There is sufficient information associated with m_sdu for the MAC sublayer entity to determine the length of the data unit. The service_class parameter indicates the quality of service requested by the MAC client. The ringlet_id parameter, if present, allows the MAC client to optionally specify the desired ring on which to transmit the m_sdu. The MAC shall obey this request except when the ringlet status shows that it is down for a protection event.

5.4.3.1.3 When generated

This primitive is generated by the MAC client entity whenever data shall be transferred to a peer entity or entities. This can be in response to a request from higher protocol layers or from data generated internally to the MAC client, such as required by Type 2 LLC service.

5.4.3.1.4 Effect of receipt

The receipt of this primitive will cause the MAC entity to insert all MAC specific fields, including header, DA,SA, and any fields that are unique to the particular media access method, and pass the properly formed frame to the lower protocol layers for transfer to the peer MAC sublayer entity or entities.

5.4.3.1.5 Additional comments

The RPR MAC protocol provides four qualities of service in service_class requested.

5.4.3.2 MA_DATA.indication

5.4.3.2.1 Function

This primitive defines the transfer of data from the MAC sublayer entity (through the MAC Control sub-layer) to the MAC client entity or entities in the case of group addresses.

5.4.3.2.2 Semantics of the service primitive

The semantics of the primitive are as follows:

```
MA_DATA.indication (header,
         destination_address,
         m_sdu,
         ringlet_id,
         reception_status)
```

The header parameter may specify one or the other ring medium, priority, Time To Live (TTL), and unicast or multicast. The destination_address parameter may be either an individual or a group address as specified

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

by the DA field of the incoming frame. The source_address parameter is an individual address as specified by the SA field of the incoming frame. The m_sdu parameter specifies the MAC service data unit as received by the local MAC entity. The reception_status parameter is used to pass status information to the MAC client entity. The ringlet_id parameter indicates, to MAC clients which optionally use the information, which ringlet the m_sdu was received from.

5.4.3.2.3 When generated

The MA_DATA.indication is passed from the MAC sublayer entity (through the MAC Control sub-layer) to the MAC client entity or entities to indicate the arrival of a frame to the local MAC sublayer entity that is destined for the MAC client. Such frames are reported only if they are validly formed, received without error, and their destination address designates the local MAC entity.Frames destined for the MAC Control sublayer are not passed to the MAC client if the MAC Control sublayer is implemented.

5.4.3.2.4 Effect of receipt

The effect of receipt of this primitive by the MAC client is unspecified.

5.4.3.2.5 Additional comments

If the local MAC sublayer entity is designated by the destination_address parameter of an MA_DATA.request, the indication primitive will also be in oked by the MAC entity to the MAC client entity. This characteristic of the MAC sublayer may be due to unique functionality within the MAC sublayer or characteristics of the lower layers (for example, all frames transmitted to the broadcast address will invoke MA_DATA.indication at all stations in the network including the station that generated the request).

5.4.3.3 MA_CONTROL.request

This primitive defines the transfer of control requests from the MAC client to the MAC Control sublayer.

5.4.3.3.1 Function

This primitive defines the transfer of control commands from a MAC client entity to the local MAC Control sublayer entity.

5.4.3.3.2 Semantics of the service primitive

The semantics of the primitive are as follows:

```
MA_CONTROL.request (header
         destination_address,
         opcode,
         request_operand_list)
```

The destination_address parameter may specify either an individual or a group MAC entity address. It must contain sufficient information to create the DA field that is preappended to the frame by the local MAC sublayer entity. The opcode specifies the control operation requested by the MAC client entity. The request_operand_list is an opcode-specific set of parameters. The valid opcode and their respective meanings are described in Table 1— on page 37.

5.4.3.3.3 When generated

This primitive is generated by a MAC client whenever it wishes to use the services of the MAC Control sublayer entity.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 1—Control Request Opcodes

Opcode	Operand	Meaning
0x00	none	No Request
0x01	none	Request Network Topology
0x02	Service_Class	Request Service Status
0x03	Station_MAC_Address	Request Station Configuration
0x04	Station_MAC_Address	Request Transit Path Congestion Status
0x05	Station_MAC_Address	Request Current Topology Database
0x06	none	Pause Message
0x07-0xFF	TBD	TBD

5.4.3.3.4 Effect of receipt

The effect of receipt of this primitive by the MAC Control sublayer is opcode-specific.(See Clause TBD.)

5.4.3.4 MA_CONTROL.indication

5.4.3.4.1 Function

This primitive defines the transfer of control status indications from the MAC Control sublayer to the MAC client.

5.4.3.4.2 Semantics of the service primitive

The semantics of the primitive are as follows:

```
MA_CONTROL.indication(header,
         opcode,
         indication_operand_list)
```

The elements of the indication_operand_list are opcode-specific, and specified in Table 2.

5.4.3.4.3 When generated

The MA_CONTROL.indication is generated by the MAC Control sublayer under conditions specific to each MAC Control operation.

5.4.3.4.4 Effect of receipt

The effect of receipt of this primitive by the MAC client is unspecified.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 2—Control Indication Opcodes

Opcode	Operand	Meaning
0x01	Network Topology Data Structure	Network Topology Change
0x02	Service_Class, Status (ok_to_send, do_not_send)	Service Status Change
0x03	configuration_parameter_list	Request Action Configuration
0x04	normalized_bandwidth_value	Request Transit Port Congestion Status
0x05-0xFF	TBD	TBD

5.5 Physical layer service access point (PSAP) definition

5.5.1 PSAP

The IEEE 802.17 MAC supports the following Physical Service Access Point (PSAP) primitives:

— PHY_LINK_STATUS.indication
— PHY_DATA.request
— PHY_DATA.indication
— PHY_DATA_VALID.indication
— PHY_READY.indication

5.5.1.1 PHY_LINK_STATUS.indication

This interface provides a means to indicate some the status of the physical link to the MLME. The setting of this MLME MIB attribute can cause the MLME to perform an action and/or request that an action be performed by the MAC.

The semantics of the primitive are as follows:

```
PHY_LINK_STATUS.indication (
          link_status
          )
```

The link_status parameter may have the values of OK, DEGRADE, or FAIL. All physical layers shall support OK and FAIL. Support for generation of DEGRADE is optional.

5.5.1.2 PHY_DATA.request

This interface defines the transfer of an octet of data from the MAC to the RS.

The semantics of the primitive are as follows:

```
PHY_DATA.request (
          output_unit
          )
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The output_unit parameter contains an octet_of_data, or DATA_COMPLETE.

5.5.1.3 PHY_DATA.indication

This interface defines the transfer of an octet of data from the RS to the MAC.

The semantics of the primitive are as follows:

```
PHY_DATA.indication (
          input_unit
          )
```

The input_unit parameter contains an octet_of_data.

5.5.1.4 PHY_DATA_VALID.indication

This interface indicates whether the parameter of PHY_DATA.indicate contains valid data.

The semantics of the primitive are as follows:

```
PHY_DATA_VALID.indication (
          data_valid_status
          )
```

The data_valid_status parameter may have the values of VALID or NOT_VALID.

5.5.1.5 PHY_READY.indication

This interface indicates whether the PHY is ready to accept a new MAC frame.

The semantics of the primitive are as follows:

```
PHY_READY.indication (
          ready_status
          )
```

The ready_status parameter may have the values of READY or NOT_READY.

5.6 Management layer service access point

5.6.1 MLSAP

The IEEE 802.17 MAC supports the following Management Layer Service Access Point (MLSAP) primitives:

— MLME_GET.request
— MLME_SET.request

5.6.1.1 MLME_GET.request

This primitive requests the value of an attribute of the MLME MIB from the MAC.

The semantics of the primitive are as follows:

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
MLME_GET.request (
          mib_attribute
          )
```

The mib_attribute parameter contains an attribute of the MLME MIB.

5.6.1.2 MLME_SET.request

This primitive requests that an attribute of the MLME MIB, residing within the MAC, be set to a given value.

The semantics of the primitive are as follows:

```
MLME_SET.request (
          mib_attribute,
          mib_attribute_value
          )
```

The mib_attribute parameter contains an attribute of the MLME MIB. The mib_attribute_value contains the requested value for mib_attribute. If the given mib_attribute is associated with a specific action, then the action shall be performed by the MAC.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

5.7 MAC reference model

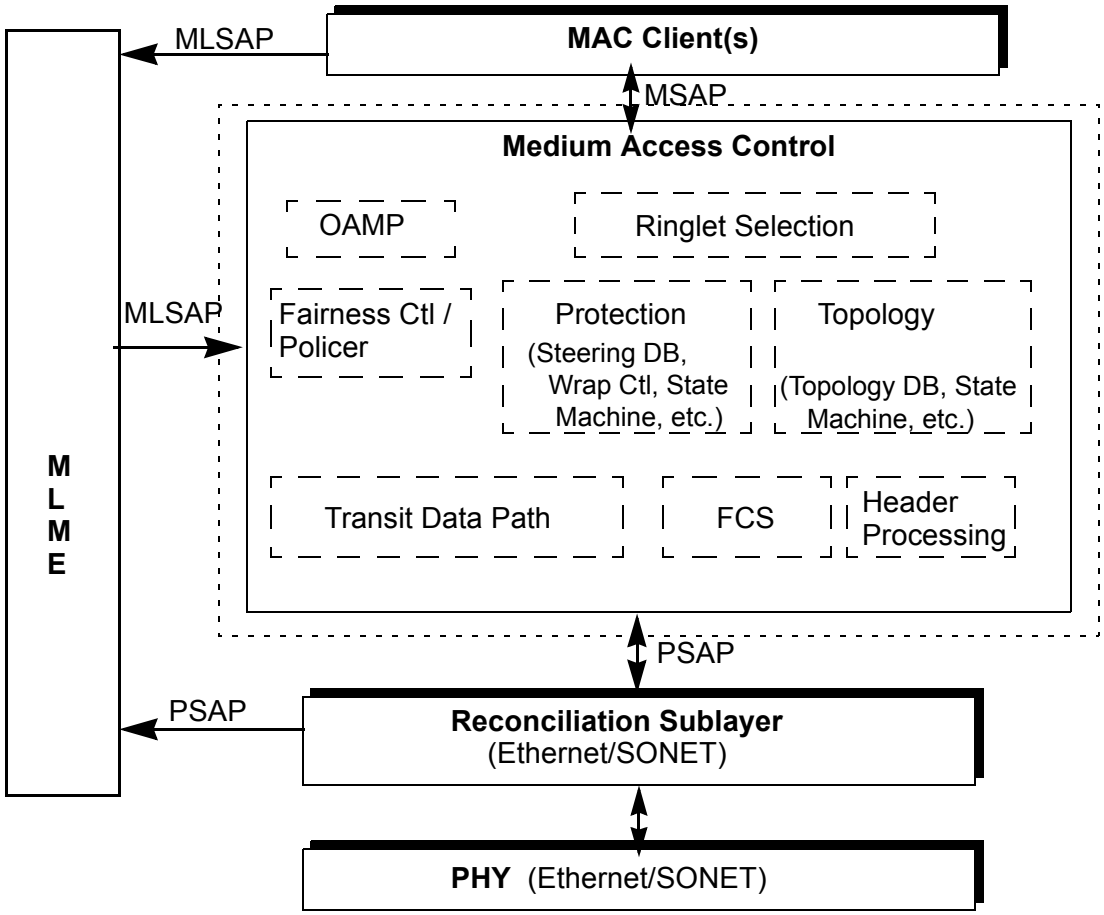


Figure 5–1—MAC reference model, showing internal MAC functions

5.7.1 PHY

The 1Gb/s and 10 Gb/s Ethernet Physical Layers and GFP SONET Physical are referenced, Synchronous HDLC is described relative to the referenced standard. It understood that different Physical Layers will provide different services. The differences are reconciled by each PHY specific Reconciliation Sublayer.

5.7.2 Reconciliation Sublayer

The Reconciliation Sublayer is part of the Physical Layer, and provides a uniform, reconciled service interface to the MAC Layer. There is one Reconciliation Sublayer Entity for each Physical Layer interface. In addition to providing a common interface for MAC/PHY data, the Reconciliation Sublayer also presents a common interface for management and control of the PHY that is needed by the MAC.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

5.7.3 RPR Medium Access Control

The RPR Medium Access Control layer provides the access control for the physical layer. It also controls the transit path through the MAC. Its functions include flow control, buffering, protection, topology discovery, and bandwidth allocation. this is the end

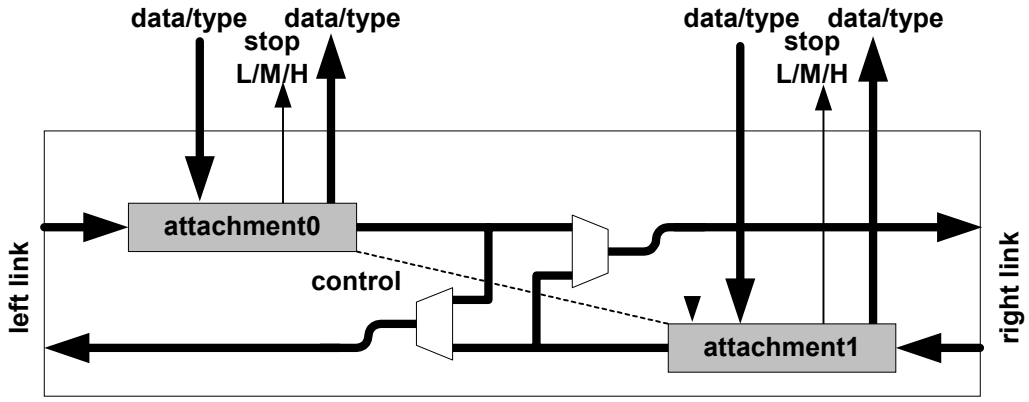


Figure 5–1—MAC client interface

5.7.3.1 Rate Control and Pacing

The Rate Control and Pacing function of the MAC sublayer controls the rate at which frames are transmitted by the MAC and coordinates this control with the other MAC sublayers on the ring.

An RPR MAC shall police access to each service by the MAC client in order to ensure that medium access and bandwidth provisioning rules are obeyed. If the client chooses to disregard feedback from the MAC on service availability and issue a DATA.request, the MAC shall accept the request but return an indication to the effect that immediate transmission is not possible. The MAC shall not transmit the packet until the packet is allowed to be transmitted.

There is one shaper each for HP, MP, and LP traffic. The shapers are simple token buckets, and if a bucket become empty or negative, the RPR MAC communicates with the MAC client with the CONFIRM primitive using the parameters STOP_HIGH, STOP_MED and STOP_LOW. If the client sends the traffic anyway, the RPR MAC may not schedule the client until the token bucket has a token in it. The detail shaper function is described in 6.2. .

5.7.3.2 Protection

The Protection function of the MAC sublayer directs frames to the appropriate ringlet based on the protection database. It also provides the protection state machine for the local MAC and coordination of this control with the other MAC sublayers on the ring.

5.7.3.3 Topology Awareness

The Topology function of the MAC sublayer manages the topology database. It also provides the topology state machine for the local MAC, and coordination of this control with the other MAC Control sublayers on the ring.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

In its canonical form, an 802.17 network consists of dual, counter-rotating ringlets. The MAC can optionally present two views of the network to the MAC client: a flat view of the network, in which the MAC sublayer hides the dual-ring-based topology from the client, or a topology-aware view in which allows the MAC client to make data and control requests for specific ringlets. Topological information is collected via a MAC sublayer entity process known as topology discovery and is made available to the client via a request to the MAC Layer Management Entity, MAC Control Indicate.

5.7.3.4 Ringlet Selection

Ringlet selection is optionally specified by trusted client. Otherwise, it is the responsibilty of MAC to decide the ringlet based on the current ring status for a particular destination address.

5.7.4 MAC Layer Management Entity

The MAC Layer Management Entity is an independent entity that resides outside of the MAC Layer in a separate management plane. The MLME contains the Management Information Base for the MAC Layer, and provides Get and Set operations on the MIB to MLME SAP user-entities. The MLME also provides actions upon the MAC Layer as a result of the invocation of Set.request primitives.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

6. Medium Access Control

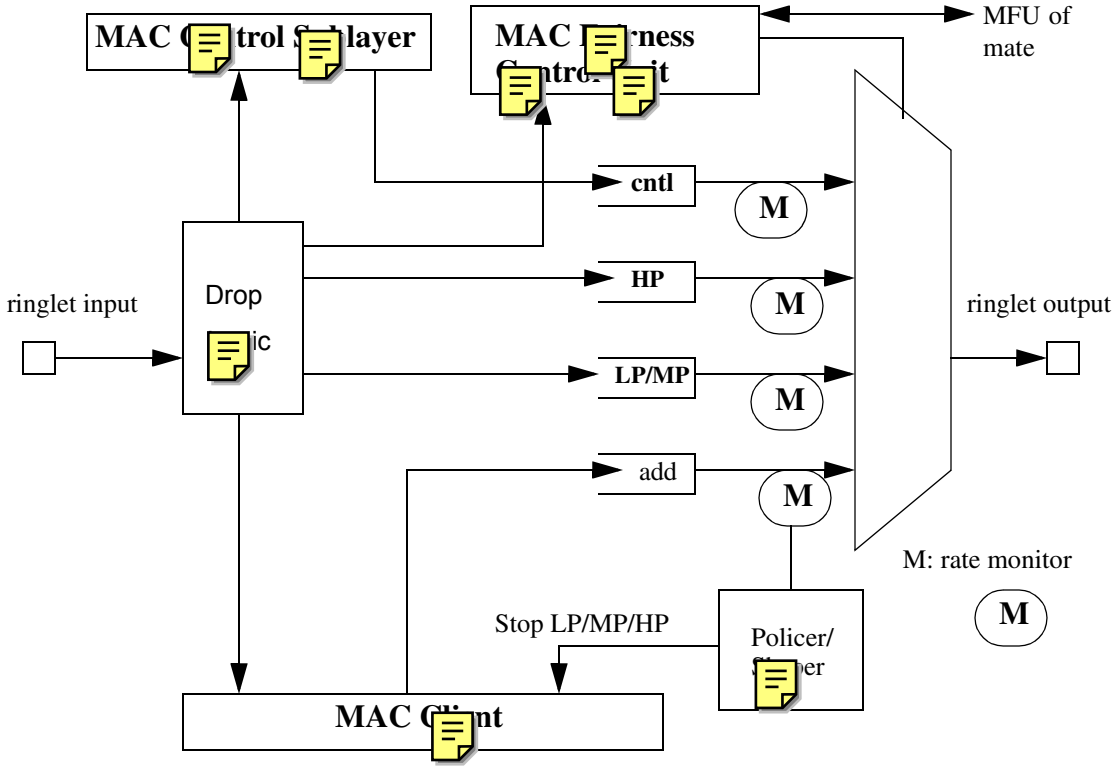


Figure 6–1—MAC data path

6.1 Transit/Transmit buffer

To be able to detect when to transmit and receive packets from the ring, RPR MAC makes use of a transit buffer as shown in Figure 6–1 below. There are two possible implementations, using either one or two transit buffers. The minimum requirement is the use of the high priority transit buffe; the inclusion of the second transit buffer is optional. In the two-transit-buffer case, traffic is eparated into three priorities, high, medium and low. High priority packets are placed in one fifo queue, while medium and low priority utilize another. In the single-transit-buffer case, only one buffer is used for all three priorities.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

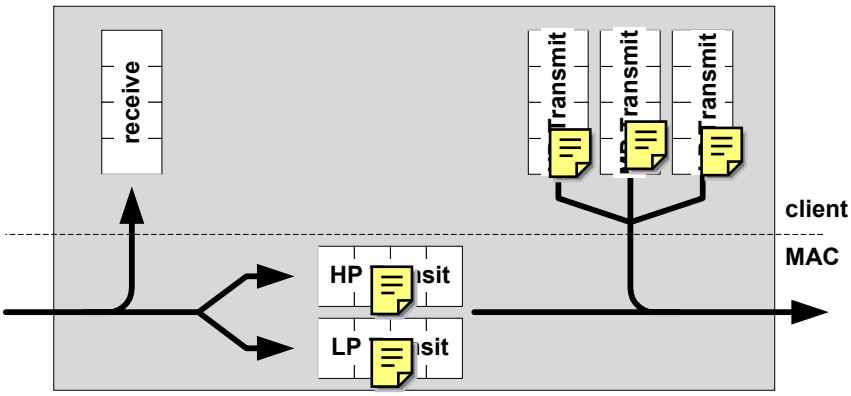


Figure 6–1—Transit / Transmit Buffer Design

6.2 Transmit and forwarding operation

An RPR MAC can transmit data packets from five possible queues:

1) High priority packets from the high priority transit buffer.
2) Medium or low priority packets from the low priority transit buffer
3) High priority packets from the client Tx high priority fifo.
4) Medium priority packets from the client Tx medium priority fifo.
5) Low priority packets from the client Tx low priority fifo.

Note that medium priority traffic is assigned a Committed Information Rate (CIR). Traffic within the CIR is treated as if it is high priority traffic while it is being accounted to the CIR. Traffic above the CIR will be treated as low priority, and will be referred to as EIR MP traffic, or eMP. The RPR MAC decides which traffic to send based on a priority scheme, which differ between single- and dual-transit-buffer implementations.

6.2.1 Single buffer implementation

In a single transit buffer implementation, transit traffic is always sent first, regardless of priority. High, medium and low priority transmit traffic will then be sent in priority order. All three classes of transmit traffic will be subjected to rate shapers. LP and eMP transmit traffic will also be limited to the fair rate governed by the RPR-fa rules.

6.2.2 Dual buffer implementation

In a dual transit buffer implementation, high priority forwarded data always gets sent first. High priority transmit data may be sent as long as the Low Priority Transit Buffer (LPTB) is not almost full. Medium Priority transmit traffic within CIR is scheduled after each priority transmit packet, and can be sent as long as the LPTB is not almost full. The value of the threshold below which medium priority packets can be sent is an implementer's option.

EIR medium priority transmit traffic residing in the medium priority transmit queue and low priority transmit traffic are sent next, assuming their combined rate does not exceed the fair rate governed by the RPR-FA rules, and the LPTB has not exceeded a low priority threshold. There is an dd_rate_ok equation to check if

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

EIR medium transmit/low priority is allowed to add traffic into ring. If the check is failed, the LPTB will be serve instead to avoid the starvation problem for LPTB traffic.

All three types of transmit traffic are also subjected to rate shapers.

If nothing else can be sent, low priority packets from the low priority transit buffer are transmitted. This decision tree is shown in Figure 6–1.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

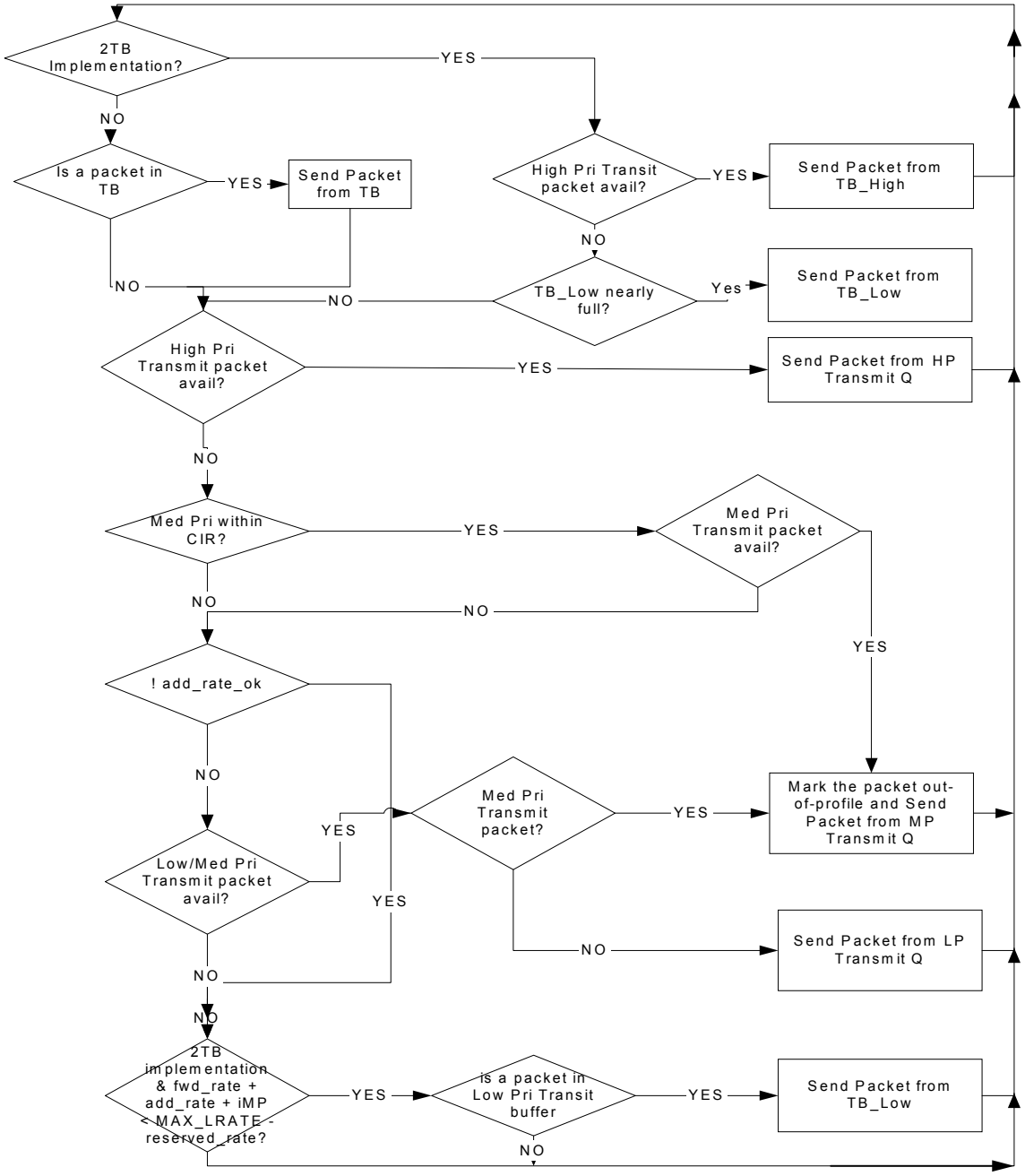


Figure 6–1—RPR transmit flowchart

6.3 Receive operation

Receive packets entering a node are copied to the receive buffer if a destination address (DA) match is made. If the DA matched packet is also a unicast, then the packet will be stripped. If the packet does not DA match or is a multicast and the packet does not source address (SA) match, then the packet is placed into

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

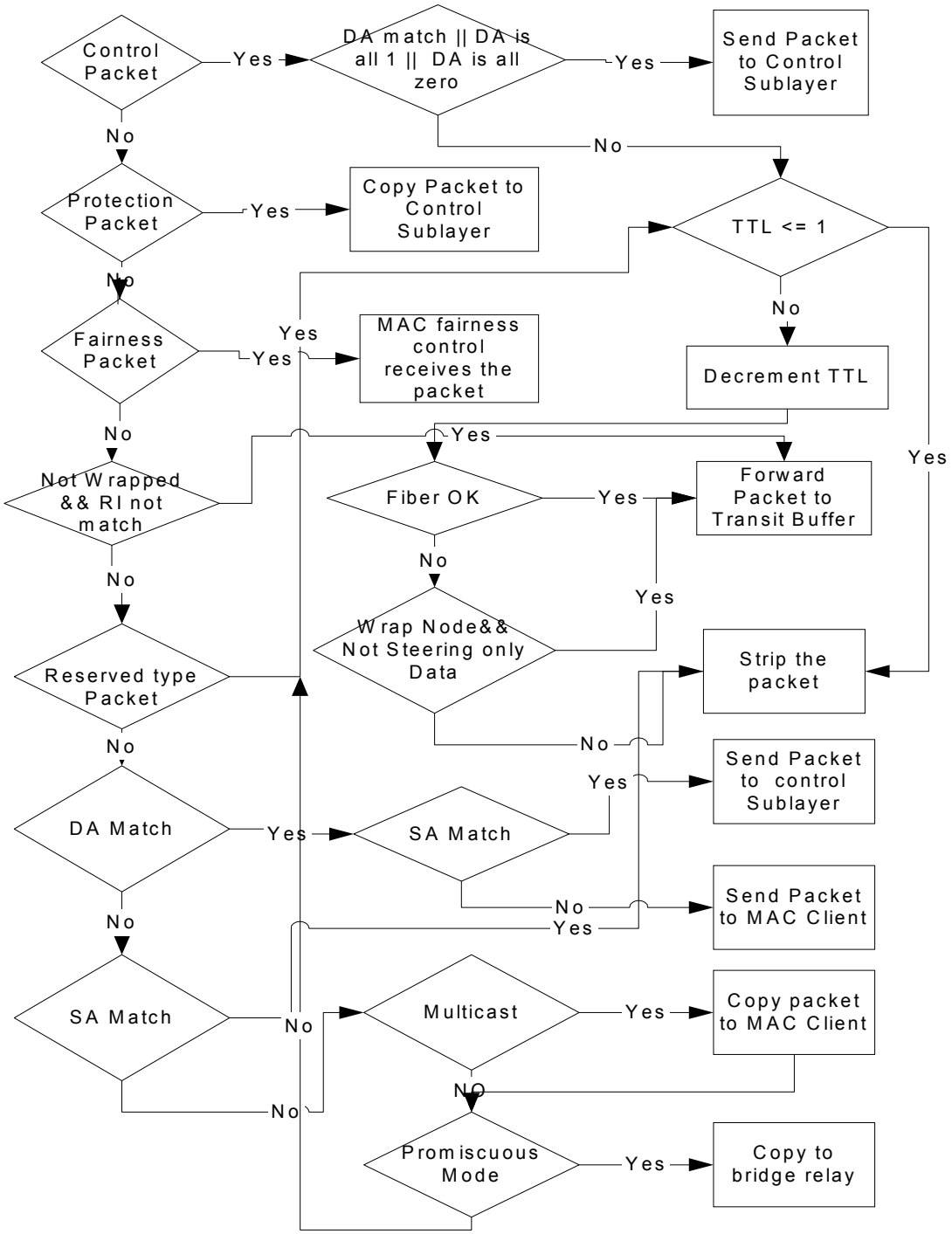


Figure 6–1—RPR Receive Flowchart

the Transit Buffer (TB) for forwarding to the next node if the packet passes the Time to Live and Header Error Check (HEC) tests.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

6.4 Transit operation

A series of decisions based on the type of packet (mode), source and destination addresses is made on the MAC incoming packets. Packets can either be control or data packets. Protection messages are broadcast to all nodes on the ring. Control packets may be hop-by-hop, broadcast or unicast destined to a station. Hop-by-hop packets are given an all-zero destination address and a hop count (TTL) of 1. Broadcast control packets use an all-ones destination address. Control packets are stripped once the information is extracted. The source and destination addresses are checked in the case of data packets. The rules for reception and stripping are given below as well as in the flow chart in Figure 6–1.

1) Strip unicast packets at the destination station.
2) Do not process packets other than for TTL and forwarding if ring identifier bit is not matched for the direction in which they are received unless the node is wrapped.
3) Do not process packets other than for TTL and forwarding if the type is not supported by the node (e.g. reserved types).
4) Transit packets will be discarded if there is a HEC error.
5) Transit packet will be optionally discarded if there is a FCS error.
6) Packets accepted by the host due to destination address match may be optionally discarded at the MAC if there is an FCS error.
7) Type 4 protection messages are broadcast and should always be copied to the MAC control sublayer.
8) Type 5 control messages are accepted and stripped if there is a DA match, a all-one DA, or a all-zero DA.
9) Packets with source address and ring identifier bit match should be stripped. If the node is wrapped and source address matches then the packet should be stripped.
10) Steering only data will be stripped in the wrapped node.
11) Conditionally decrement TTL on receipt of a packet, discard if it gets to zero; do not forward. The conditions to decrement TTL are as follows: always decrement unless the ring is in the wrap state (anywhere) and the ring id in the packet an in the MAC do not match.

Notes:

i) FCS error packets should be passed to the client when there is a DA match. It is the client's responsibility whether to accept or drop the error packet. The CRC will be overwrited with the newly invert CRC to indicate the FCS error.
ii) Conditionals (if statements) in Figure 6–1 branch to the right if true and branch down if false.

6.4.1 Transit operation in a bridge (promiscuous mode)

When the RPR MAC is part of a bridge, all data packets are copied to the Bridge Relay Entity and forwarded to the transit buffer.

Optional behaviors to improve bridging performance include the use of a MAC filtering database to hold the DA and SA of stations that are located behind the bridge: In this case the DA and SA of the packet can be checked to determine if the packet is to be dropped or stripped. If the addresses are not found in the database then the same rules as promiscuous mode apply.

6.5 Circulating packet detection (stripping)

Packets continue to circulate when transmitted packets fail to get stripped. Unicast packets are normally stripped by the destination station or by the source station if the destination station has failed. Multicast packets are only stripped by the source station. If both the source and destination stations drop out of the ring

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

while a unicast packet is in flight, or if the source node drops out while its multicast packet is in flight, the packet will rotate around the ring continuously.

The solution to this problem is to have a TTL or hop count field in each packet that is set to the number of nodes in the ring. As each node forwards the packet, it decrements the TTL. If the TTL reaches zero it is stripped off of the ring. In order to allow 256 nodes on a wrapped ring, the TTL is not decremented when the packet is on the opposite ring and the ring is still wrapped. Once the ring unwraps, TTL decrements are performed on all packets. This catches the case where the packet is stuck on the wrong ring.

The ring identifier bit is used to qualify all stripping and receive decisions. This is necessary to handle the case where packets are being wrapped by some node in the ring. The sending node may see its packet on the reverse ring prior to reaching its destination so must not source strip it.

A potential optimization would be to allow ring identifier bit independent destination stripping of unicast packets. One problem with this is that packets may be delivered out of order during a transition to a wrap condition. For this reason, the ring identifier bit should always be used as a qualifier for all strip and receive decisions.

6.6 Wrapping of data

Normally, transmitted data is sent on the same ring to the downstream neighbor. However, if a node is in the wrapped state, transmitted data is sent on the opposite ring to the upstream neighbor. Packets of type 0x3 are marked for steering only, and when they reach a wrap point they are stripped.

6.7 Pass-thru mode

An optional mode of operation is pass-thru mode. In pass-thru mode, a node transparently forwards data. The node does not source or sink packets. It may optionally decrement the TTL and adjust the HEC but does no other modifications to the packets that it forwards. Data should continue to be sorted into high and low priority transit buffers with high priority transit buffers always emptied first. The node does not source any control packets (e.g. topology discovery or protection switch protocol) and basically looks like a signal regenerator with delay (caused by packets that happened to be in the transit buffer when the transition to pass-thru mode occurred). A node can enter pass-thru mode because of an operator command or due to a error condition such as a software crash

The justification for continuing with the TTL decremented operation is to prevent a packet from being delivered twice if the node that sourced the packet is the node that goes into passthru. This could cause packets to be stripped early when topology discovery has determined that the ring contains fewer stations and adjusts the TTL value down in magnitude.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

7. MAC physical interface

Editors' Notes: *To be removed prior to final publication.*

References:
TBD.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002 Initial draft document for RPR WG review.

7.1 Overview

Resilient packet ring (RPR) is media independent. RPR frames may be sent over different physical media.

7.2 SONET/SDH network

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that the following two paragraphs should be deleted from Clause 7, and reviewed for possible inclusion in Clause 11 (Protection) or an informative annex.

RPR may also connect to a SONET/SDH ring network via a tributary connection to a SONET/SDH add-drop multiplexer (ADM). The two RPR rings may be mapped into two STS-Nc connections. SONET/SDH networks typically provide fully redundant connections, so RPR mapped into two STS-Nc connections will have two levels of protection. The SONET/SDH network provides layer 1 protection, and RPR provides layer 2 protection. In this case it is recommended to hold off the RPR Signal Fail protection message triggers (which correspond to failures which can be protected by SONET/SDH) using a programable hold-off timer in order to allow the SONET/SDH network to protect. Only if a failure persists for a time exceeding the programmable hold-off time (indicating SONET/SDH protection failure) should the protection switch protocol take place.

Since multiple protection levels over the same physical infrastructure are not very desirable, an alternate way of connecting RPR over a SONET/SDH network is configuring SONET/SDH without protection. Since the connection is unprotected at layer 1, RPR would be the sole protection mechanism.

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that the following three paragraphs be deleted in their entirety, since they provide no normative information, and it is not clear how to include them in a specification.

Hybrid RPR rings may also be built where some parts of the ring traverse over a SONET/SDH network while other parts do not.

Connections to a SONET/SDH network would have to be synchronized to network timing by some means. This can be accomplished by locking the transmit connection to the frequency of the receive connection (called loop timing) or via an external synchronization technique.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Connections made via dark fiber or over a wavelength-division multiplexed (WDM) optical network should utilize internal timing as clock synchronization is not necessary in this case.

7.2.1 Byte-synchronous HDLC framing

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> ***Editors suggest that the following two paragraphs be replaced with specific references to framing requirements using byte-synchronous HDLC-like framing. Comments and suggestions are welcome.***

Flag delimiting on SONET/SDH uses the octet stuffing method defined for byte-synchronous HDLC. The packet delimiter flags (0x7E) are required for SONET/SDH links but may not be necessary for RPR on other media types. An end-of-packet is delineated by a flag, which might also be the next packet's starting flag. If the data appears to be a flag (0x7E) or an escape character (0x7D) anywhere inside of a packet, the data must be marked with an escape character.

SONET/SDH framing plus byte-synchronous HDLC packet delimiting allows RPR to be used directly over fiber or through an optical network (including WDM equipment).

7.2.2 GFP framing

Generic Framing Procedure (GFP) is a standard method of delineating octets of variable length payload into octet-synchronous payload envelope as defined in ANSI T1.105.02 and ITU-T G.707, G.709. GFP defines a frame format for protocol data units (PDUs) transferred between GFP initiation and termination points, as well as the mapping procedure for client signals into GFP.

The GFP delineation scheme uses a header error check (HEC) polynomial. The latest GFP standard can be found in ITU specification G.7041.

The preferred method for GFP to encapsulate 802.17 frames is to use the [illegible] header definition as defined by the Extension Header Identifier (EXI) with a [illegible] Payload Identifier (UPI) for RPR payload.

The SONET/SDH Reconciliation Sublayer (RS) and the MAC sublayer provide an optional capability to propagate the PDU length value to optimize the forwarding of PDU from one SAP to another.

GFP allows RPR packets to be transported over SONET/SDH and OTN networks.

7.3 Ethernet

RPR frames can be sent over Ethernet physical media.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that this clause be substantially re-written using the following outline as a guide:

1. Overview, including scope, objectives, layer diagram, functional diagram/description, and notation conventions
2. Service interface definition
3. Detailed description
4. PICS

Comments with specific suggestions and/or concerns are welcome.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8. Frame formats

8.1 Overview

This clause defines in detail the frame structure for data communication systems using the RPR MAC. It defines the syntax and semantics of the various components of the MAC frame.

Four frame formats are specified in this clause:

a) A data RPR frame format,
b) A control RPR frame format, and
c) An extension of the basic RPR frame format for tagged RPR frames, i.e., frames that carry QTag
d) An extension of the basic RPR frame format for CID RPR frames, i.e., frames that carry CID

This section describes the frame formats used by RPR. Packets can be sent over any point to point link layer (e.g. SONET/SDH, point to point ETHERNET connections). The maximum transfer length (MTU) is 9216 octets.

These limits include everything listed in Figure 8–1 but are exclusive of the frame delineation (e.g. for RPR over SONET/SDH, the flags used for frame delineation are not included in the size limits).

The following frame format does not include any layer 1 frame delineation. For RPR over POS, there isan additional flag that delineates start and end of frame.

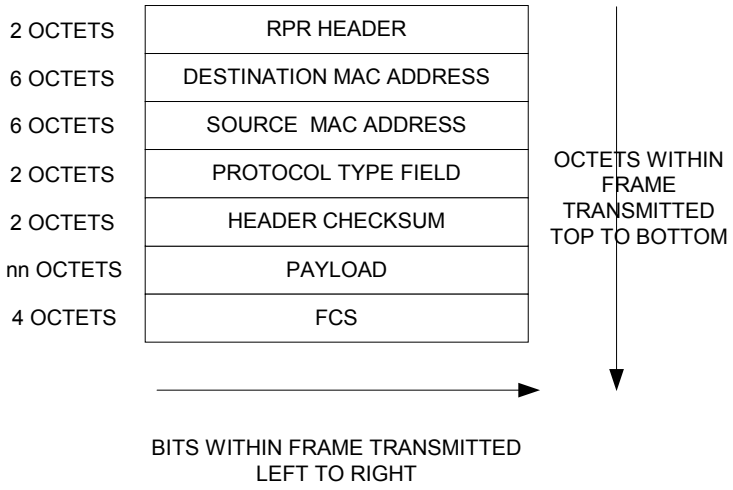


Figure 8–1—RPR frame format

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.2 RPR packet header format

Each packet has a fixed-sized header. The packet header format is shown in Figure 8–1

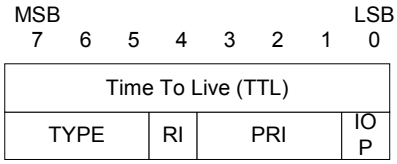


Figure 8–1—RPR Packet Header Format

The fields are described below:

8.2.1 Time To Live (TTL)

This 8-bit field is a hop count that must be decremented every time a node forwards a packet. If the TTL reaches zero the packet is not forwarded. While the packet is traveling on a different ringlet (wrapped) due to failure protection, TTL is not decremented. The 8-bit field allows for a total of 256 nodes on a ring.

8.2.2 Type field

This 3-bit field identifies he mode of the packet. The following modes are defined in Table 3:.

Table 3—Type Values

Value (bin)	Description
000	Reserved
001	Reserved
010	Reserved
011	Steering only data
100	Protection Control packet
101	Control packet
110	Fairness packet
111	Data packet

These modes will be further explained in later sections.

8.2.3 Ring Identifier

This bit indicates the ringlet onto which the frame was originally transmit.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 4—Ring Identifier

	Ring Identifier
inner ringlet	0
outer ringlet	1

8.2.4 Priority field (PRI)

This 3-bit field indicates the priority level of the RPR packet (0 through 7). The higher the value, the higher the priority. Since there are only two queues in the transit buffer (HPTB and LPTB) a packet is treated as either low or high priority once it is on the ring. Each node sets a threshold value for distinguishing high and low priority packets. For consistency between nodes, by default only priority-7 packets shall be queued in the high-priority transit buffer. The remaining packets shall be queued in the low-priority transit buffer. However, the full 8 levels of priority in the RPR header may be used prior to transmission onto the ring (transmit queues) as well as after reception from the ring (receive queues).

8.2.5 IOP

This bit is used to mark whether the packet is in or out of profile. If the bit is not set for a medium priority packet, the packet is within the Committed Access Rate or within profile(0). Otherwise, the packet is excess Medium priority traffic (eMP) or out of profile(1). Excess Medium priority traffic or out of profile(1) traffic is treated as low priority traffic in the RPR fairness algorithm.

8.3 Overall packet format

The overall packet format is as shown in Figure 8–1.

8.3.1 IEEE 802.17 address field

a) Each address field shall be 48 bits in length.
b) The least **>>>NO it is the MOST<<<**significant bit of the most significant (first transmitted) octet is used to identify the destination address either as an individual or as a group address. If this bit is 0, it shall indicate that address field contains an individual address. If the bit is 1, it shall indicate that the address field contains a group address. In the source address field, this bit is reserved and set to 0.

A MAC sublayer address is one o two types:

a) Individual Address. The address associated with a particular station on the ringlet.
b) Group Address. A multi-destination address, associated with one or more stations on the ringlet..

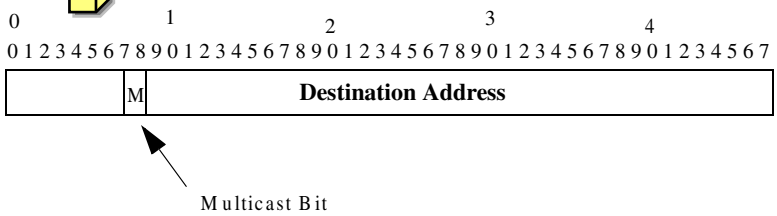


Figure 8–1—Group address bit position

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

1) Mutlticast-Group Address. An address associated by higher-level convention with a group of logically related stations. Multicast addresses have the most-significant bit of the first octet set to 1.
2) Broadcast Address. An address that denotes the set of all stations on a given ringlet. All 1's in the destination address shall be predefined to be the Broadcast Address.

8.3.2 Destination address

The destination address is a globally unique 48 bit IEEE 802.17 address.

8.3.3 Source address

The source address is a globally unique 48 bit IEEE 802.17 address.

8.3.4 Protocol type/Length field

When the value of this field is greater than or equal to 1536 decimal (equal to 0600 hexadecimal) the Type Field indicates the nature of the MAC client protocol (Type interpretation). When less than 1536, the field is interpreted as the length of the packet. The length and type interpretations of this field are mutually exclusive.

The value of the Type Field is obtained from the IEEE Type Field Registrar. Additional values for RPR are given in Table 5.

Table 5: Defined protocol type

Value	Protocol Type
TBD	RPR Control
TBD	Payload with Customer ID

8.3.5 HEC field

This is a 16 bit HEC. The generator polynomial is:

$$\text{HEC-16} = x^{16} + x^{12} + x^{5} + 1$$

The HEC is computed over the RPR header, destination address, source address and protocol type. Single-bit error correction by the receiver is optional.

8.3.6 FCS

The frame check sequence (FCS) is a 32-bit cyclic redundancy check (CRC) as specified in RFC-1662 and is the same CRC as used in Packet over SONET (POS - specified in RFC-2615). The generator polynomial is:

$$\text{CRC-32} = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x^{1} + 1$$

The FCS is calculated starting from the octet following the HEC to the end of packet. The inital value for CRC calculation is an all-ones value.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.4 RPR control packet format

If the type field is set to 101 then this indicates a control message. RPR control packet could be a hop-by-hop ,broadcast or point-to-point unicast message. If the control packet is a hop-by-hop message, the destination address field for control packets should be set to 0's.The TTL is not relevant but it preferably should be set to one. If the control packet is broadcast, the TTL shall be set as the number of node on the ring for broadcast control packet. The destination address shall be set as broadcast address which all is 1.The source address field for a control packet should be set to the source address of the transmitting node.

The control packet format is shown in Figure 8–1.

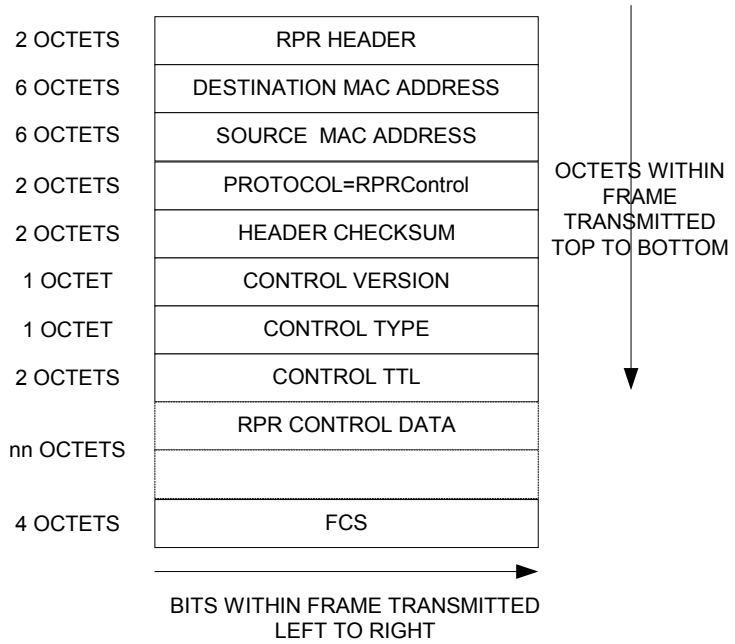


Figure 8–1—Control Packet Format

The priority (PRI) value in RPR header of control packet shall be set to 0x7 (all one's) except for some OAM control packet which might be set to any PRI value as indicated by the control command.

8.4.1 Control ver

This one octet field is the version number associated with the control type field. Initially, all control types will be version 0.

8.4.2 Control type

This one octet field represents the control message type. Table 6 contains the currently defined control types.

8.4.3 Control TTL

The Control TTL is a control layer hop-count that must be decremented every time a node forwards a hop-by-hop control packet. If a node receives a control packet with a control TTL <= 1, then it should accept the packet but not forward it.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 6—Control types

Control Type	Description
0x01	Topology Discovery
0x02	Protection message
0x03	OAM control packet
0x04 - 0xFF	Reserved

8.4.4 Payload

The payload is a variable length field dependent on the control type.

8.5 RPR Fairness Frame Format

RPR Fairness Frame as shown in Figure 8–1 is sent to MAC neighbors to convey data for the fairness algorithm specified in Clause 12. The IOP bit in Fairness frame serves as a parity bit for the header.

2 OCTETS	RPR HEADER (TYPE=0x6)
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	FAIRNESS CONTROL HEADER
2 OCTETS	CONTROL VALUE
4 OCTETS	FCS

Figure 8–1—RPR Fairness Frame Format

8.6 Order of bit transmission

Each octet of RPR frame, with the exception of the FCS, is transmitted high-order bit first.

8.7 Invalid RPR frame

An invalid RPR frame is defined as one that meets at least one of the following conditions

a) HEC does not match with the frame as received.
b) (Optional) FCS does not match with the frame as received.

The contents of invalid RPR frames shall not be passed to the Client or MAC control sublayers. The occurrence of invalid MAC frames may be communicated to network management.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.8 Elements of a tagged RPR frame

The tagged RPR frame format is shown as in Figure 8–1. This format is an extension of the RPR frame format.

2 OCTETS	RPR HEADER
6 OCTETS	DESTINATION MAC ADDRESS
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	PROTOCOL TYPE=0x8100
2 OCTETS	HEADER CHECKSUM
2 OCTETS	IEEE 802.1Q VLAN TAG
2 OCTETS	PAYLOAD TYPE FIELD
nn OCTETS	MAC CLIENT DATA
4 OCTETS	FCS

Figure 8–1—Tagged RPR frame format

8.8.1 Protocol Type/Length field

The Protocol Type/Length field of a tagged RPR frame always uses the Type interpretation, and contains the 802.1Q Protocol Type: a constant equal to 0x8100.

8.8.2 Tag control information field (informative)

The Tag control information field is subdivided as follows in Figure 8–1

MSB 7	6	5	4	3	2	1	LSB 0
USER PRIORITY			C FI	VID(11:8)			
VID(7:0)							

Figure 8–1—Tag control information field

a) 3-bit User Priority field
b) Canonical Format Indicator (CFI), and
c) 12-bit VLAN Identifier

The structure and semantics within the Tag control information field are defined in IEEE 802.1Q.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.8.3 Payload Type

The payload type field followed by the control information field contains the original protocol type from RPR frame prior to add the QTag prefix. The value for this field is the same as Ethernet type defined in 802.3.

8.9 Elements of RPR frame with customer separation ID

The payload of RPR frame might have a customer separation ID (CID) as shown in Figure 8–1.

2 OCTETS	RPR HEADER
6 OCTETS	DESTINATION MAC ADDRESS
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	PROTOCOL TYPE=CID
2 OCTETS	HEADER CHECKSUM
4 OCTETS	CID
2 OCTETS	PAYLOAD TYPE FIELD
nn OCTETS	MAC CLIENT DATA
4 OCTETS	FCS

Figure 8–1—RPR with CID

8.9.1 Protocol Type/Length field

The Protocol Type/Length field of a CID RPR frame always uses the type interpretation, and contains a TBD value assigned by IEEE Registration Authority.

8.9.2 Customer separation ID field

This 32 bit field might be used to as an identifier of customers to separate customer traffic.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

8.9.3 Payload type

This field shall be used as type interpretation for the payload. All Ethernet type values

Table 7:

Value	Protocol T
0x8100	VLAN [illegible]ged [illegible]me

might be applied here. Also, additional type values may be requested for RPR frames in the future.

8.9.3.1 Tagged within CID RPR frame

2 OCTETS	RPR HEADER
6 OCTETS	DESTINATION MAC ADDRESS
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	PROTOCOL TYPE=CID
2 OCTETS	HEADER CHECKSUM
4 OCTETS	CID
2 OCTETS	PROTOCOL TYPE = 0x8100
2 OCTETS	IEEE 801.1Q Vlan Tag
2 OCTETS	PAYLOAD TYPE FIELD
nn OCTETS	MAC CLIENT DATA
4 OCTETS	FCS

Figure 8–1—Tagged frame within CID RPR frame

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

9. MAC fairness

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
fair rate:
decay interval:
TB_HI_THRESHOLD:
TB_LO_THRESHOLD:
weighting function:
virtual destination queueing:
multi-choke:
add_rate:
forward_rate:
reserved_rate:
max_rate
allowed rate:
access delay:

Abbreviations:

Revision History:
Draft 0.1 February 2002 Initial draft document for RPR WG review.

9.1 Overview

Editors' Notes: *To be removed prior to final publication.*

The overview section needs to be rewritten in the standard format per directions from the chief editor. Missing things in the overview section include:
- Scope
- Goals and objectives
- Relationship to other clauses (done using the MAC reference model) - refer to the clause discussing the service interface.
- Functional block diagram and contents of clause
- Notational conventions

More details are needed on the service interface with the MAC client with appropriate references to the service interface definitions clause.

Detailed state diagrams are needed that describe:
the events that cause fairness messages to be sent;
the actions when a fairness message is received;
congestion and congestion subsided, etc.

The PICS needs to be added.

The RPR-fa is a local fairness mechanism that enforces fairness among the stations on the ring. It applies to only to LP and excess Medium priority (eMP) traffic coming from the MAC client.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

There is no MAC fairness entity in the MAC reference model. Need to make sure that this is consistent withy whateever we define in the reference model. I think the present version calls it a fairness control (FCU).

The RPR-fa is implemented within a control entity called the MAC Fairness Entity (MFE). The MFE does not need to understand the ring topology, however it utilizes the implicit topology information (i.e. TTL value) passed by the MAC client to perform fairness and policing functions.

The fairness algorithm consists of the following three functions:

d) Determining when the congestion threshold is crossed and when the congestion has subsided
e) Determining the fair rate for advertisement
f) Determine the station's allowed rate

Each station is assigned a weight, which allows the user to allocate more ring bandwidth to certain stations in times of congestion. This is referred to as the *weighted fairness* property of RPR-fa.

Editors' Notes: *To be removed prior to final publication.*

In the paragraph below, I will be removing the references to implementations (e.g add_rate, etc.) and moving it to a NOTE-- following the paragraph where I will describe implementation examples including this one.

In RPR-fa, a node advertises a fair rate (e.g. add_rate) to upstream nodes via the opposite ring. The fair rate counter is run through a low pass filter function and divided by a weighting function (e.g. local station weight). The low-pass filter stabilizes the feedback, and the division by weight normalizes the transmitted value to a weight of 1.0. When the upstream stations receive an advertised fair rate, they will adjust their transmit rates so as not to exceed the advertised value (adjusted by their respective weights).

Propagation of the advertised value to other nodes on the ring is done using fairness messages as described in Clause 9.5. There are 2 types of fairness messages -- Type 1 and Type 2.

Type 1 messages are propagated hop-by-hop and contain the SA of the most congested node. If a node experiences congestion, it will create a Type 1 fairness message to advertise its fair rate. A station that receives an advertised value, and that is also in the congested state, shall propagate the minimum of its normalized low pass filtered advertised fair rate and the received fair rate. Type 1 messages are processed by the MFE and the information contained is also passed to the MAC clients.

Type 2 messages are broadcast and contain the SA of the node that originated the message. They are not processed by the MFE, but the information is passed to the MAC client.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The next 4 paragraphs describe client behavior for a multi-choke implementation. Since it represents a combination of tutorial information and implementation guidelines, it is better moved to an informative annex.*

If the TTL of the data packet from the client is less than the number of hops to the node that generated the fairness message, then the client can take advantage of the available bandwidth on the ring; otherwise, if it uses a TTL value that would allow the packet go beyond the node that generated the fairness message, the client will at least receive its fair share of the bandwidth from the most congested span that it contends for.

The client can also take advantage of Virtual Destination Queueing (VDQ) by utilizing the multi-choke concept of RPR-fa. VDQ combined with RPR-fa will increase the utilization.

The multi-choke concept deals with the case where a node wants to send traffic to a destination that is closer than a congested link. As an example, consider the case where node 1 wants to send traffic to node 2, and the link between nodes 2 and 3 is congested. RPR-fa will allow node 1 to send as much traffic as it wants to node 2, and will only limit traffic to nodes beyond the congested link to the fair rate.

In a multi-choke implementation of the RPR-fa, each client will track advertised fair rates for congested nodes. A node is allowed to send unlimited traffic to any node between itself and the first congested node (choke point). It can send traffic to nodes between the first and second choke point based on the first choke point's advertised fair rate. In general, a node can send traffic to a particular destination if it has satisfied the fair rate conditions for all choke points between itself and the destination. The theoretical maximum number of choke points is equal to the number of nodes on the ring.

9.2 Congestion detection

Congestion is declared by using the following criteria depending on the type of transit path design.

If the transit path has only a single buffer, then congestion is detected when:

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *Again, I think the part following the i.e. is a candidate for a NOTE--.*

a) the outgoing link rate passes a configured congestion threshold. The outgoing link rate is measured using a byte counter which is passed through a low pass filter before the comparison. The node will be considered to be congested until its outgoing link utilization falls below a configured threshold. i.e. if (add_rate + forward_rate) is more than (max_rate - reserved_rate) then it is congested, or
b) the access delay timer expires for LP and eMP packets

If the transit path has 2 buffers, then congestion is detected when:

a) the outgoing link rate passes a configured congestion threshold, or
b) access delay timer expires for LP and eMP packet, or
c) the depth of the low priority transit buffer reaches a congestion threshold.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

9.3 Interoperability between single/dual transit buffer MACs

If a ring consists of mixed RPR MACs (single and dual transit buffer nodes), the fairness scheme will need to interact without disadvantaging any other node on the ring. The actions required for the interaction between single transit and dual transit buffer nodes is the same as the interaction between similar nodes. Upon receiving a congestion message the RPR MAC shall reduce its allowed rate to the fair rate received in the fairness message multiplied by the local weight, and then forward the message upstream with the minimum of its own advertised rate and the received fair rate.

The RPR MACs should rate-shape their transmit traffic using the dynamic traffic shaping algorithm described in Clause 6.2. Also, in a mixed node ring configuration, dual transit buffer MACs should employ a traffic shaper to limit their outgoing traffic to a certain preset percentage of the line rate (default value is 95%) in order to allow the downstream single transit buffer MACs to meet their HP delay jitter requirement.

9.4 Threshold settings

Editors' Notes: *To be removed prior to final publication.*

We talk about TB_HI/TB_LO_THRESHOLD and how to set them, but the text never describes how they are actually used. They are probably used in the c-code. Need to pull in some description of this into the overview section. Also, since this is clearly an implementation guideline, we need to move it to an informative annex.

The high priority transit buffer needs to hold 2 to 3 MTUs or about 30 KB.

The adequate sizing of the low priority transit buffer and associated high and low threshold values (TB_HI_THRESHOLD, TB_LO_THRESHOLD) depends on the ring size and traffic profile of the ring. The recommended value for 10 Gbps rings with a ring diameter of 100 Km is 256 KB. For 10 Gbps, 1000 Km rings, the recommended value is 512 KB. For 10 Gbps, 3000 Km rings, 1 MB of low priority transit buffer is recommended.

The goal of setting the appropriate threshold values is to deliver best possible end-to-end delay for the low priority traffic without penalizing the high priority traffic.

The following guidelines can be used to determine the proper threshold values:

TB_LO_THRESHOLD should be set to about 25% of the total buffer available. Lower values will result in higher end-to-end delays for low priority data packets. If either low or high priority data traffic is extremely bursty, then a lower threshold value should be considered.

TB_HI_THRESHOLD should be set to about (total buffer size - 1MTU).

If the high priority data traffic has a bursty nature a more conservative (lower) value of TB_LO_THRESHOLD is recommended to avoid overflow of the low priority transit buffer.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

9.5 RPR fairness packets

9.5.1 When generated

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *"decay interval" has not been defined until now. Check other clauses for definition and refer to that, or define it over here.*
>
> *Also, for the second paragraph below, we should probably reference the protection clause if that is one of the ways for protection to get activated*
>
> *Also, need to talk about "When received", and "When not received" to complete this sub clause.*

RPR fairness packets are sent out periodically to propagate allowed rate information to upstream stations. RPR fairness packets also perform a keepalive function for the Protection entity. The recommended fair rate period is between the decay interval and 1 MTU transmission time.

If a receive interface has not seen a fairness packet within the keepalive time-out interval it will trigger an L2 keepalive time-out interrupt/event. The protection software will subsequently mark that interface as faulty and initiate a protection switch around that interface. The keepalive time-out interval should be set to 16 times the RPR fairness packet transmission interval.

9.5.2 Packet format

The fairness message format is as shown in Figure 9–1—.

2 OCTETS	RPR HEADER (TYPE=0x6)
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	FAIRNESS CONTROL HEADER
2 OCTETS	CONTROL VALUE
4 OCTETS	FCS

Figure 9–1—Fairness Packet Format

The Fairness Control Header is shown in Figure 9–2—. The fair rate is encoded as a 16-bit quantity within the "control value" field. A fair rate of all ones indicates a value of NULL.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

MSB 7	6	5	4	3	2	1	LSB 0
VERSION			RESERVED				
RESERVED							
CONTROL VALUE (15:8)							
CONTROL VALUE (7:0)							

Figure 9–2—Fairness Control Header

9.5.2.1 Version field (3 bits)

This field is to specify the version number of fairness packet. Table 8 shows the fairness message version values. Type 1 fairness message is used to implement basic RPR-fa. Type 2 fairness message is needed to support a multi-choke implementation.

Table 8—Version values

Value (binary)	Type of fairness packet	How is it used
000	Type 1 fairness packet	Follows RPR-fa fair rate rules, generated in every fair rate interval. SA is the MAC address of the most congested node in the fairness domain
001	Type 2 fairness packet	Generated every 10 fair rate intervals by every MAC with its SA and broadcast. The fair rate is passed to every client on the ring.
010 to 111	Reserved	For future use.

9.5.2.2 Reserved field (12 bits)

It is set to 0 for Type 1 and Type 2 fairness packets.

9.5.2.3 Length field (Optional 8 bits)

This is optional field within reserved field to specify the length of fairness packet. It is set to 0x00 for Type 1 and Type 2 fairness packets.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

9.5.2.4 Control value (16 bits)

Editors' Notes: *To be removed prior to final publication.*

We need some more explanation about the normalization factor.

This field carries the fair rate (total number of bytes/normalization_factor added to the ring by the node) to the upstream node when congestion is detected. The normalization factor is 1 for OC-48 and below, 16 for OC-192 and proportional thereafter. A value of 0xFFFF indicates the availability of up to line rate bandwidth

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

10. Topology Discovery

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

Material for this clause is yet to be adopted by the P802.17 WG.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

11. Protection

Editors' Notes: *To be removed prior to final publication.*

References:
To be added.

Definitions:
To be added.

Abbreviations:
To be added.

Revision History:
Draft 0.1, February 2002 Initial draft document for RPR WG review.

Resiliency is an important RPR objective. An RPR ring shall provide protection within 50ms of detection of a link or station failure. There are two possible protection mechanisms: wrapping and steering. All RPR stations shall provide support for steering, with support for wrapping optional. All stations within the same RPR ring shall choose the same protection mechanism.

Via the topology discovery protocol, every RPR station shall indicate if it supports wrapping protection or not. If all stations on an RPR ring are able to support wrapping protection, the choice of wrapping or steering for the protection mechanism shall be based on configuration. Otherwise, steering shall be selected as the protection scheme on the RPR ring.

A protection message shall be proposed to signal the failure between station within the same RPR ring. That message shall accommodate the necessary information for RPR to do wrap or steer protection.

Editors' Notes: *To be removed prior to final publication.*

The introduction/overview needs to be re-written to be descriptive rather than normative. It needs to provide, among other things, a description of the objectives met by the protection switching portion of the standard. It needs to put the protection section in context relative to the MAC reference model.

Editors' Notes: *To be removed prior to final publication.*

A section needs to be added to put this section in context with the remainder of the standard in terms of the MAC reference model.

11.1 Wrap protection

An RPR wrap capable ring is composed of two counter-rotating, single fiber ringlets. If an equipment or fiber facility failure is detected, traffic going towards and from the failure is wrapped (looped) back to go in the opposite direction on the other ring (subject to the protection hierarchy). Wrapping takes place on the stations adjacent to the failure, under control of the protection switch protocol. The wrap re-routes the traffic away from the failure.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The diagrams need to be modified to be colored in black and gray only and to follow editorial conventions for the standard.

.

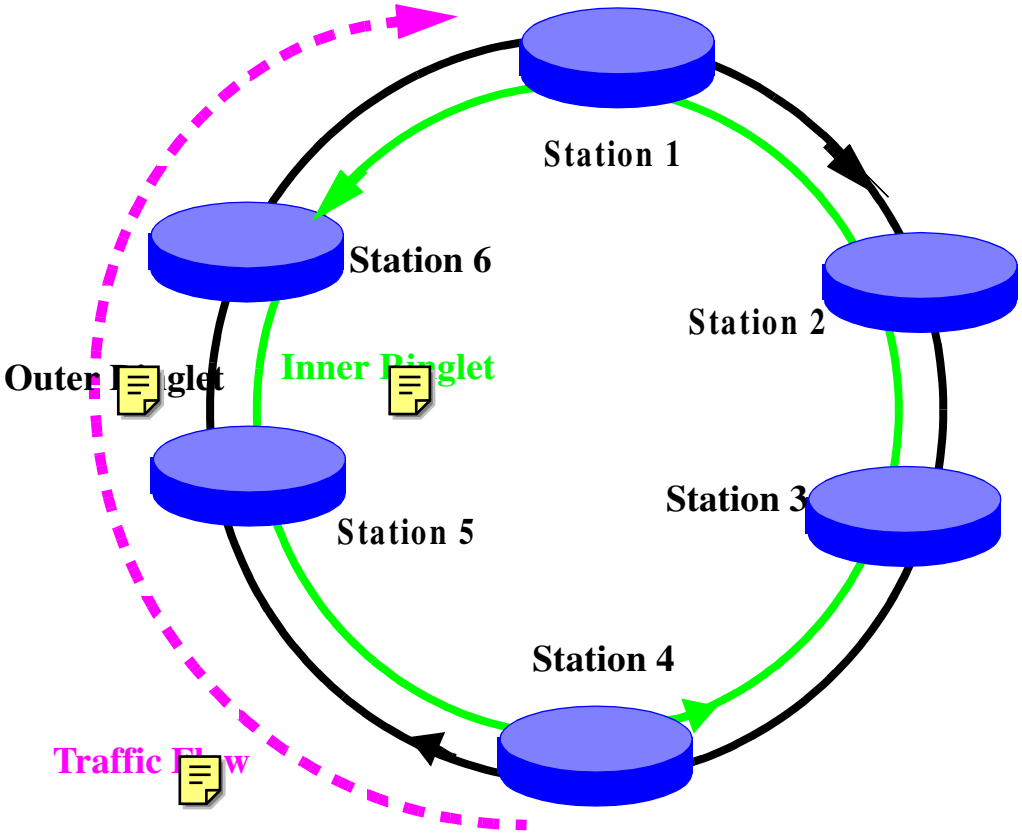


Figure 1—Data flow before fiber cut.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

An example of the data paths taken before and after a wrap is shown in Figure 1 and Figure 2, respectively. Before the fiber cut, Station 4 sends to Station 1 via the path S4->S5->S6->S1.

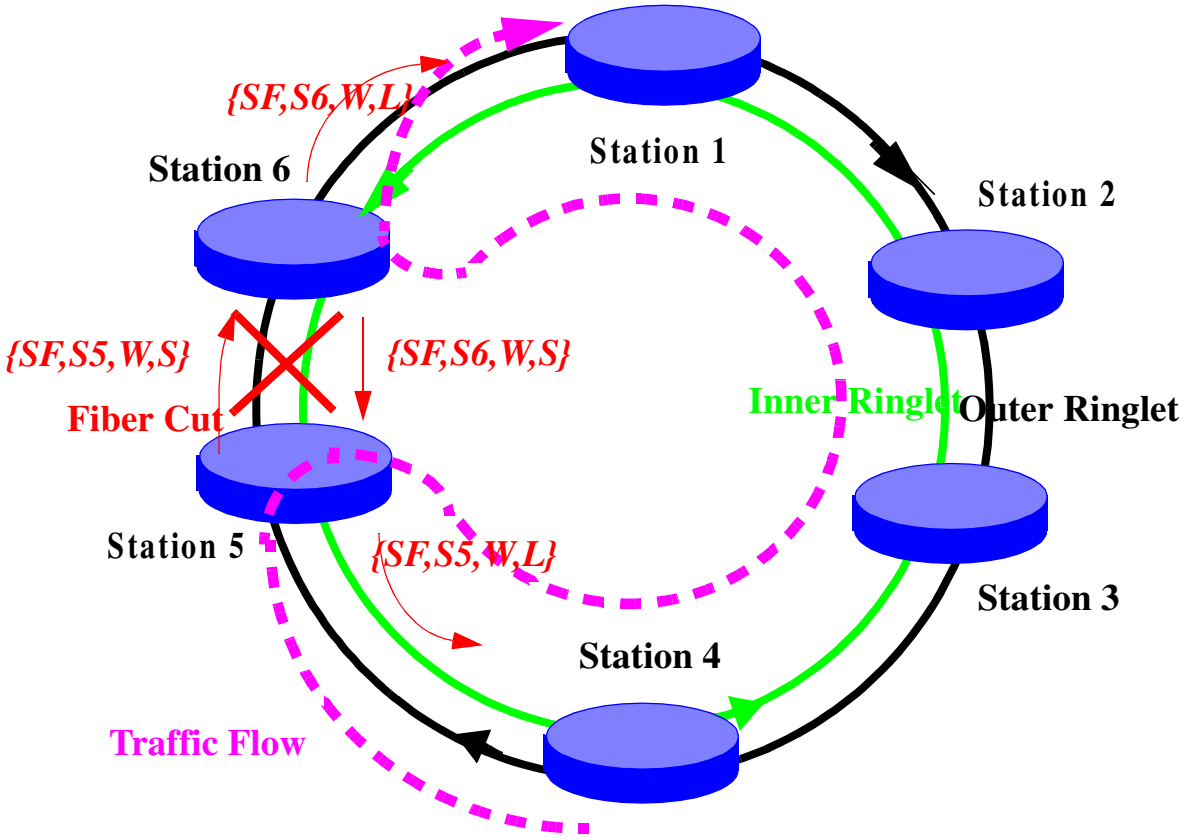


Figure 2—Data path after wrap

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

If there is a fiber cut between Station 5 and Station 6, Station 5 and Station 6 will wrap the inner ringlet traffic to the outer ringlet. After the wraps have been set up, traffic from Station 4 to Station 1 initially goes through the non-optimal path S4->S5->S4->S3->S2->S1->S6->S1.

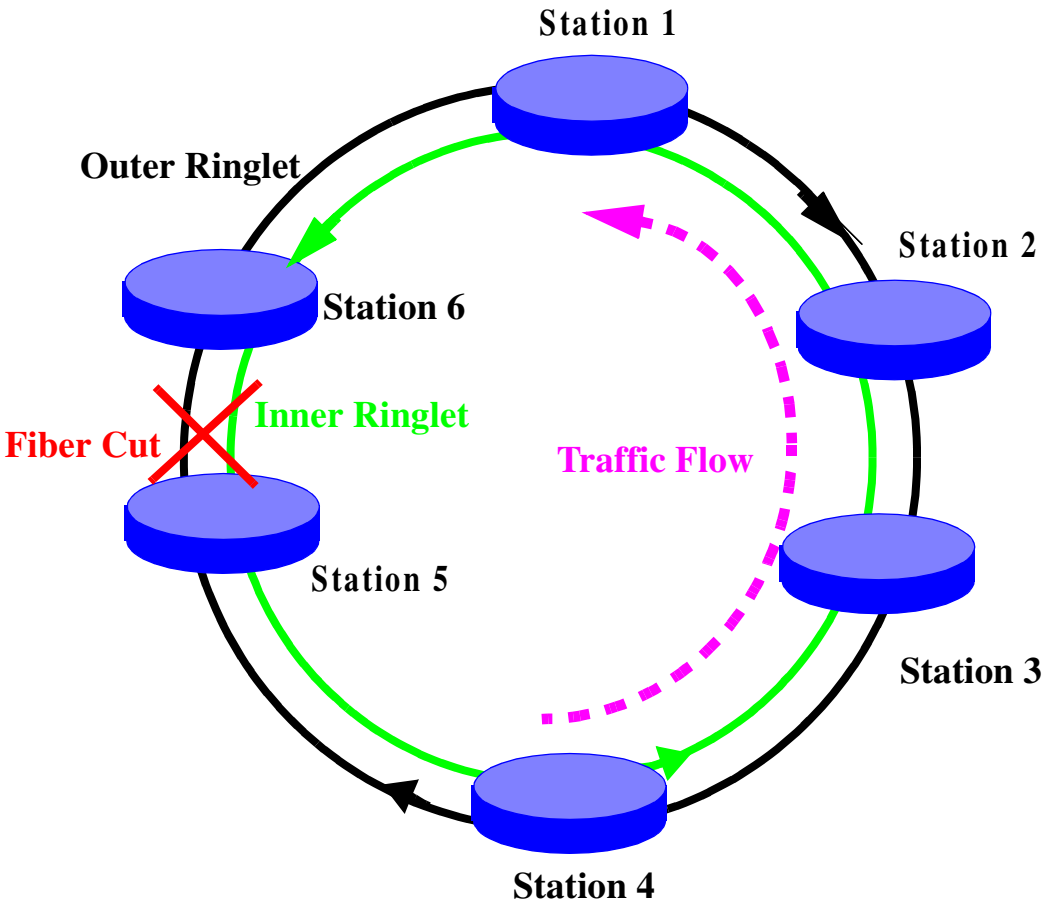


Figure 3—Data path after new topology discovery

Subsequently a new ring topology is discovered and a new optimal path S4->S3->S2->S1 is used, as shown in Figure 3. Note that the topology discovery and the subsequent optimal path selection are not part of the protection switch protocol.

The ring wrap is controlled through SONET BLSR [3][4] style protection switch signaling. It is an objective to perform the wrapping at least as fast as in SONET equipment.

11.2 Steering protection

For steering protection, a station shall not wrap a failed segment when a failure is detected. Instead, a protection request message is sent to every station to indicate a link failure, as in the wrap protection scheme. When stations receive a protection request message indicating a failure, their steering database will be updated accordingly. It is the responsibility of each source station to direct its traffic onto an inner or outer ringlet that avoids the failed link.

Packets that have been transmitted onto the ring, that are destined to a station beyond the point of failure, before the steering database is updated at the source station, will be dropped at the failure point, since there is no delivery mechanism available.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The text in Sections 14.1 and 14.2 is almost entirely descriptive. It would fit well in an overview section. The steering protection description needs to be expanded to correspond to the wrapping description.

11.3 Multicast consideration

For steering rings, when a link fails, multicast packets shall be sent in both directions, with the TTL set to the number of stations on the ring. Duplicate packets shall not be allowed to arrive at any station on the ring as a result of a protection condition ceasing to exist. For wrapping rings, multicast packets continue to be sent in a single direction, with the TTL set to the number of stations on the ring

Editors' Notes: *To be removed prior to final publication.*

A true normative section is needed that incorporates multicast requirements.

11.4 Protection hierarchy

The protection switch protocol processes the following request types (in the order of priority, from highest to lowest):

1) Forced Switch (FS): operator originated, performs a protection switch on a requested link.
2) Signal Fail (SF): automatic, caused by a media Signal Failure or RPR keep-alive failure, performs a protection switch on the effected link.
3) Signal Degrade (SD): automatic, caused by a media Signal Degrade (e.g. excessive Bit Error Rate), performs a protection switch on the effected link.
4) Manual Switch (MS): operator originated, like Forced Switched but of a lower priority.
5) Wait to Restore (WTR): automatic, entered after a link meets the restoration criteria following exit from SF or SD condition. Protection switch is then maintained for the WTR period before restoring traffic in order to prevent protection switch oscillations.

Editors' Notes: *To be removed prior to final publication.*

Protection triggers need to be included in a separate subsection. There is no explicit definition yet provided for these triggers in the context of the PHY and Reconciliation Sublayer sections of the draft.

The text of this section as written is descriptive rather than normative. The appropriate protions of the text need to be made normative.

There is also no clear listing of configurable parameters pertaining to both the wrapping and steering protocols.

For wrapping rings, as protection requests travel around the ring, the protection hierarchy is applied. If a requested protection switch is of a higher priority (e.g. a Signal Fail request is of higher priority than a Signal Degrade request), then the requested protection switch takes place and any lower priority protection

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

switches elsewhere in the ring are superseded and ignored. When a lower priority request is presented, it is not allowed if a higher priority request is present in the ring. The only exception is multiple SF and FS switches, which can coexist in the ring.

All protection switches are performed bidirectionally (protect at both ends of a segment for both transmit and receive directions, even if a failure is only unidirectional).

11.5 Protection message packet format

The protection switch message is used for signaling various link failures and degradations. The protection switch message packet format is outlined in Figure 4.

2 OCTETS	RPR HEADER(TYPE=0x4)
6 OCTETS	DESTINATION MAC ADDRESS
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	PROTOCOL=RPR Control
2 OCTETS	HEADER CHECKSUM
1 OCTET	CONTROL VERSION(0x0)
1 OCTET	CONTROL TYPE(0x2)
1 OCTET	Protection Message Octet
1 OCTET	Reserved
4 OCTETS	FCS

Figure 4—Protection switch packet format

The protection switch specific fields are detailed below.

11.5.1 Destination MAC Address

The Destination MAC Address is a pre-registered multicast address for protection switch packets. Protection switch messages are multicast in order to minimize the transmission delay.

11.5.2 Source MAC Address

The Source MAC Address is the MAC address of the originator of the protection message.

11.5.3 Protection Message Octet

The Protection Message Octet contains specific protection information. The format of the protection message octet is shown in Table 9.

The currently defined request types with values, hierarchy, and interpretation are as used in SONET BLSR [3], [4], except as noted.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 9: Protection message octet format

Bit	Value
0-3	**Protection Message Request Type** 1101 - Forced Switch (FS) 1011 - Signal Failure (SF) 1000 - Signal Degrade (SD) 0110 - Manual Switch (MS) 0101 - Wait to Restore (WTR) 0000 - No Request (IDLE)
4	**Path Indicator** 0 - Short (S) 1 - Long (L)
5-7	**Status Type** 010 - Protection Switch Completed - Traffic Wrapped (W) 000 - Idle

11.5.4 Protection message request types

The following is a list of the request types, from the highest to the lowest priority. All requests are signaled using protection control messages.

1) Forced Switch (FS - operator originated)
   Operator invoked protection switch on the specified link, steering or wrapping all traffic away from the link. An example use of this request type is to add another station to the ring in a controlled fashion.
2) Signal Failure (SF - automatic)
   Protection caused by a media “hard failure” or RPR keep-alive failure. SONET examples of SF triggers are: Loss of Signal (LOS), Loss of Frame (LOF), Bit Error Rate (BER) above a preselected SF threshold, and Alarm Indication Signal. Note that the RPR keep-alive failure provides end-to-end coverage, and as a result SONET Path triggers are not necessary.
3) Signal Degrade (SD - automatic)
   Protection caused by a media “soft failure”. SONET examples of SD triggers are: Line BER or Path BER above a preselected SD threshold.
4) Manual Switch (MS - operator originated)
   Like FS, but of lower priority. An example use of this request type is to force down a marginally operating link, but allow it to come back into use in the case of a more serious failure elsewhere on the ring.
5) Wait to Restore (WTR - automatic)
   Entered when a link meets the restoration threshold after an SD or SF condition disappears. The protection switch protocol waits for the WTR time-out before restoring traffic. An example use of this request type is to prevent protection switch oscillations.

11.5.5 Protection message path indicator

The Path Indicator indicates on which path a protection message is sent, long and short. Short path messages are sent towards the failed link through the opposite ringlet. They indicate a failure on the other ringlet on the link immediately preceding the station whose address is given in the source address of the protection request message. Long path messages are sent away from the failed link on the same ringlet. They indicate a

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

failure on the same ringlet on the link immediately preceding the station whose address is given in the source address of the protection request messages.

The protection control messages are shown in this document as:

{REQUEST_TYPE, SOURCE_ADDRESS, WRAP_STATUS, PATH_INDICATOR}

Editors' Notes: *To be removed prior to final publication.*

The text of this section as written is descriptive rather than normative. The appropriate protions of the text need to be made normative.

11.6 RPR protection protocol states

Each station in the protection protocol is in one of the following states for each of the rings:

11.6.1 Idle

In this mode a station is ready to perform a protection switch. This state is entered based on no protection request being present on links connecting to the station.

11.6.2 Wrapping/Steering

In this mode, a station is participating in a protection switch on a link connecting it to a neighboring station. This state is entered based on a protection request issued locally or based on a protection message received from a neighboring station.

Editors' Notes: *To be removed prior to final publication.*

The state diagram below needs to be expanded to cover the protection protocol rules as defined below, as the state diagram traditionally supercedes text from a normative standpoint. The rules will be molded into a description of the state diagram.

11.7 Protection protocol rules

11.7.1 RPR protection packet transfer mechanism

R T.1:

Protection packets are transferred in a broadcast packet format between stations on the ring. A received packet (payload portion) is passed to the receiving station's MAC control sublayer.

R T.2:

All protection messages are triggered by self-detect or user request. Protection messages are sent when the local protection state changes.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

11.7.2 RPR protection signaling and wrapping mechanism

R S.1:

Protection switch signaling is performed using protection control packets as defined in Figure 4.

R S.2:

A station executing a local request signals the protection request on both short (across the failed span) and long (around the ring) paths after detecting a link failure. The station will repeat the protection message every T1 timer until it receives its protection message. The default T1 timer is one second.

R S.3:

Protection control packets are never wrapped.

R S.4:

If the protection protocol calls for sending both short and long path requests on the same segment (for example if a station has all fibers disconnected), only the short path request should be sent.

R S.5:

A station on a wrapping RPR ring wraps and unwraps only as a result of a local request or a short path request. A station never wraps or unwraps as a result of a long path request. On wrapping rings, long path requests are used only to maintain the protection hierarchy. (Since the long path requests do not trigger wrapping protection, there is no need for destination addresses and no need for topology maps).

11.7.3 Example

Editors' Notes: *To be removed prior to final publication.*

The examples below need to be included in a descriptive Annex.

In Figure 5, Station A detects SF (via local request or self-detected request) on the span between Station A and Station B and starts sourcing {SF, A, W, S} on the outer ringlet and {SF, A, W, L} on the inner ringlet. Station B receives the protection request from Station A (short path request) and starts sourcing {IDLE, B, W, L} periodically.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

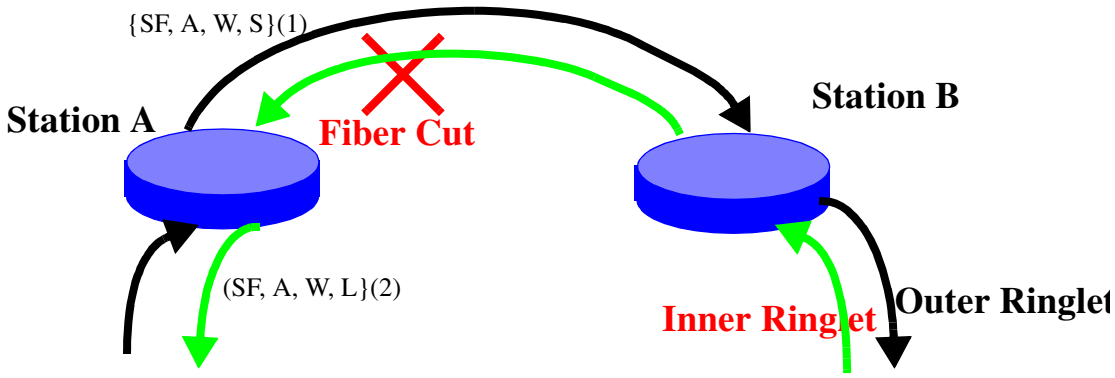


Figure 5—RPR protection switch signaling

11.8 RPR protection protocol rules

Editors' Notes: *To be removed prior to final publication.*

The rules below need to be reformatted to conform with the IEEE editorial template and style.

R P.1:

The protection request hierarchy values are listed in Table 10 (listed from highest priority to lowest priority). In general, a higher priority request preempts a lower priority request within the ring, with exceptions noted as rules. The 4 bit values shown in the table correspond to the REQUEST_TYPE field in the protection packet.

Table 10: Protection request type

Value
Protection request type 1101 - Forced S[illegible]h (FS) 1011 - Signal F[illegible]SF) 1000 - Signal [illegible]ade (SD) 0110 - Manua[illegible]h (MS) 0101 - Wait to [illegible]e (WTR) 0000 - No [illegible] (IDLE)

R P.2:

Requests >= SF can coexist. All requests above SF need to be cleared before the state is transferred into idle state.

R P.3:

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Requests < SF can not coexist with other requests. A higher priority request will preempt a lower priority request.

R P.4:

A station always honors the highest of {short path request, self detected request} if there is no higher long path message passing through the station.

R P.5:

When there is more than one request of priority < SF, the first request to complete long path signaling will take priority. However, a higher priority request can preempt the request as long as its long path signal is completed.

R P.6:

A station will strip a protection packet which was originally generated by the station itself (based on the source address).

R P.7:

When a station receives a long path request and the request is >= to the highest of {short path request, self detected request}, the station checks the message to determine if the message is coming from its neighbor on the short path. If that is the case, then it strips the message.

R P.8:

When a station receives a long path request, it strips the request if it is a wrapping/steering station with a request >= than that in the request; otherwise it passes it through.

R P.9:

Each station keeps track of the addresses of the immediate neighbors through topology discovery.

R P.10:

When a station which initially detected a failure discovers the disappearance of the failure, it enters WTR (for a user-configured WTR time-period). WTR can be configured with values in the range of 10-600 sec., with a default value of 60 sec.

R P.11:

When a station is in WTR mode, and detects that a new neighbor (as identified from the received short path protection message) is not the same as the old neighbor (stored at the time of wrap initiation), the station drops the WTR.

R P.12:

When a station is in WTR mode, and the source of a long path request is not equal to its neighbor on the opposite side (as stored at the time of wrap initiation), the station drops the WTR. This is the case when a new neighbor is added to the ring.

R P.13:

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

When a station receives a local protection request of type SD or SF and it cannot be executed (according to protocol rules), it keeps the request pending. (The request can be kept pending outside of the protection protocol implementation.)

R P.14:

If a local non-failure request (WTR, MS, FS) clears, and if there are no other requests pending, the station enters idle state.

R P.15:

If there are two link failures and two resulting WTR conditions on a single segment, the second WTR to time out brings both the links up. (After a WTR time expires, a station does not unprotect automatically, but waits until it receives idle messages from its neighbor on the previously failed segment.)

R P.16:

If a short path FS request is present on a given segment, and a SF/SD condition takes place on the same segment, a station shall accept and process the SF/SD condition ignoring the FS. (Without this rule, a single ended wrap condition could take place, wrapping on only one end of a segment.)

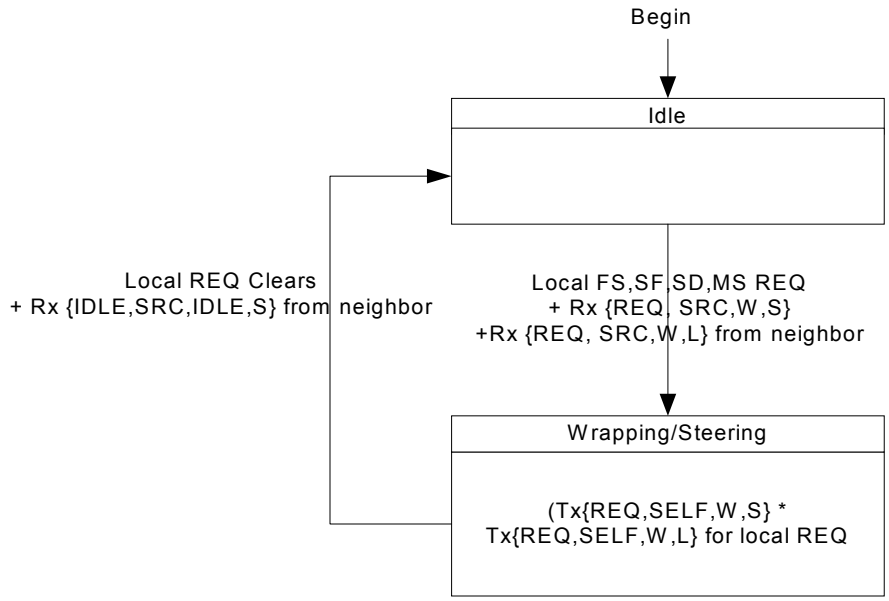
R.P.17:

If a station receives a protection message, it should update its topology accordingly and trigger a topology discovery process.

11.9 Protection state transition

Figure 6 shows the simplified state transition diagram for the protection protocol:

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.



Legend:
neighbor = node on the other end of the affected span
REQ is any request = FS,SF,SD,MA

Figure 6—Simplified protection state transitions diagram

11.10 Failure examples

11.10.1 Signal failure - single fiber cut scenario

This sample scenario is a ring of four stations: A, B, C, and D, with a unidirectional failure on a link from A to B, detected on B. The ring is in the Idle state (all stations are Idle) prior to the failure.

11.10.1.1 Signal fail scenario

1) Station B detects SF on the outer ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, B, W, S}towards A on the inner ringlet/short path, and transmits {SF, B, W, L}on the outer ringlet/long path.
2) Station A receives a protection request on the short path, and transitions to Wrapped state.
3) Steady state is reached.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

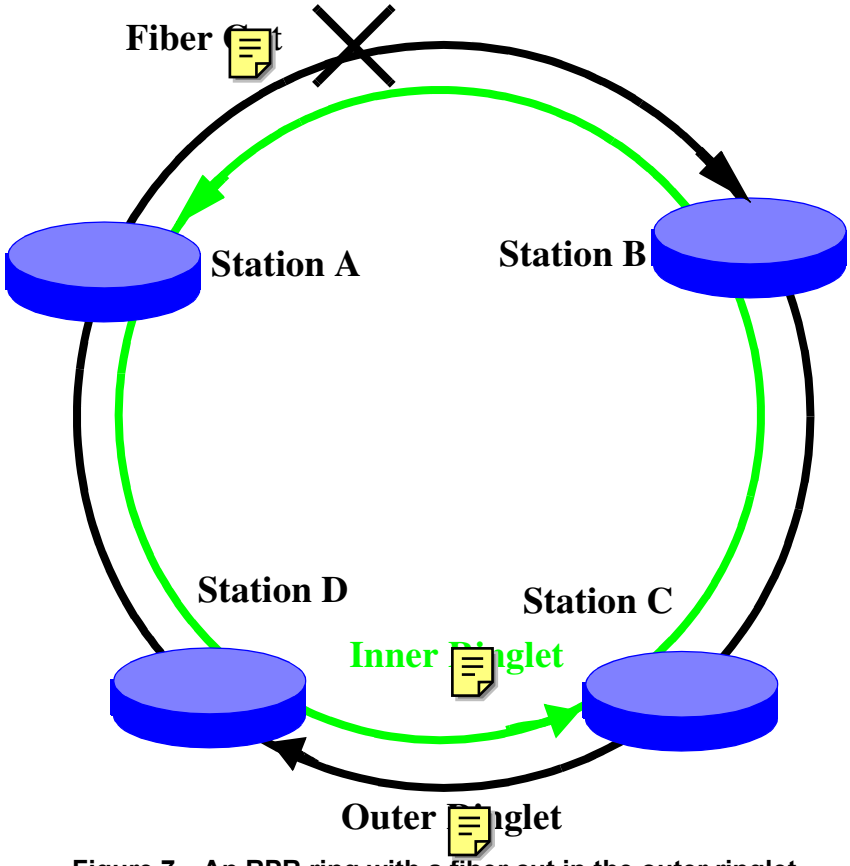


Figure 7—An RPR ring with a fiber cut in the outer ringlet

11.10.1.2 Signal fail clears

1) SF on Station B clears, Station B does not unwrap if it is in wrap state, sets WTR timer, transmits {WTR, B, W, S} on the inner ringlet, and transmits {WTR, B, W,L} on the outer ringlet.
2) Station A receives a WTR request on the short path, and does not unwrap.
3) Steady state is reached.
4) WTR times out on Station B. Station B transitions to idle state (unwraps), transmits {IDLE, B, I, S} on the inner ringlet, and transmits{IDLE,B,I.L} on the outer ringlet.
5) Station A receives {IDLE, B, I, L} and transitions to Idle.
6) Steady state it reached.

11.10.2 Signal failure - bidirectional fiber cut scenario

This sample scenario is a ring of four stations: A, B, C, and D, with a bidirectional failure between A and B. The ring is in the [illegible] state (all stations are [illegible] prior to the failure.

11.10.2.1 Signal fail scenario

1) Station A detects SF on the inner ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, A, W, S} towards B on the outer ringlet/short path, and transmits {SF, A, W, L} on the inner ringlet/long path.
2) Station B detects SF on the outer ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, B, W, S} towards A on the inner ringlet/short path, and transmits {SF, B, W, L} on the outer ringlet/long path.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

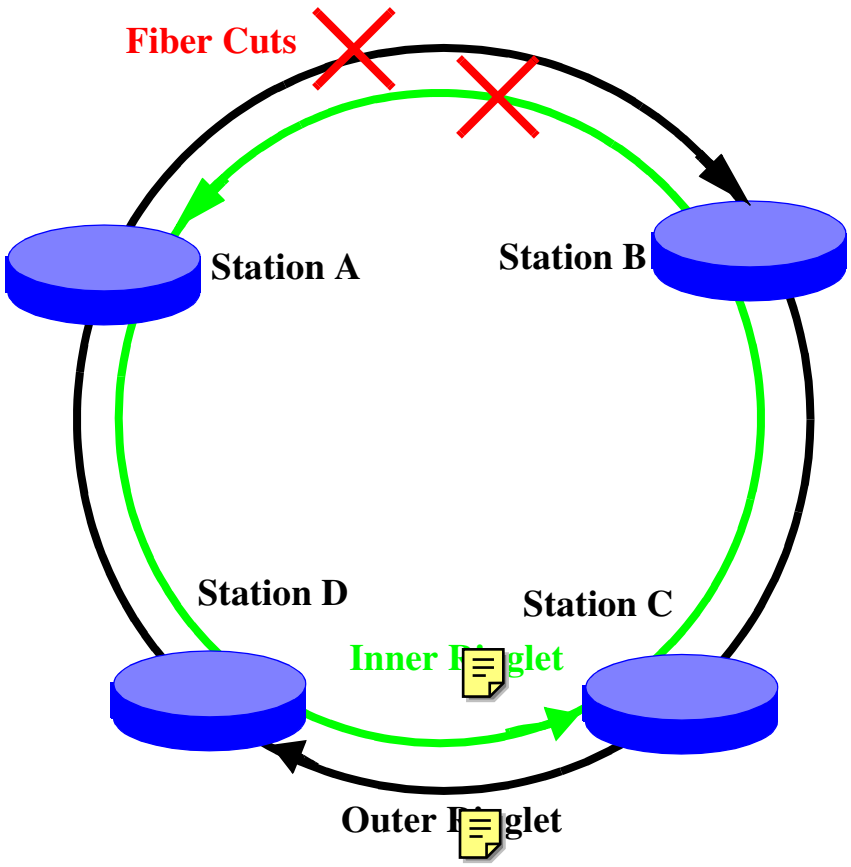


Figure 8—An RPR ring with bidirectional fiber cut

3) Steady state is reached.

11.10.2.2 Signal fail clears

1) SF on A clears, A does not unwrap, A sets the WTR timer, transmits {WTR, A, W, S} through the outer ringlet towards B, and transmits {WTR, A, W, L} on the long path through the inner ringlet.
2) SF on B clears, B does not unwrap. Since it now has a short path WTR request from A, it acts upon this request. It keeps the wrap, transmits {IDLE, B, W, S} towards A, and transmits {WTR, B, W, L} on the long path.
3) Stations C and D relay long path messages without changing the protection switch octet.
4) Steady state is reached.
5) WTR times out on A. A enters the idle state (drops wrap) and starts transmitting idle on both ringlets.
6) B sees an idle request on the short path and enters idle state.
7) Steady state is reached.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

11.10.3 Failed station scenario

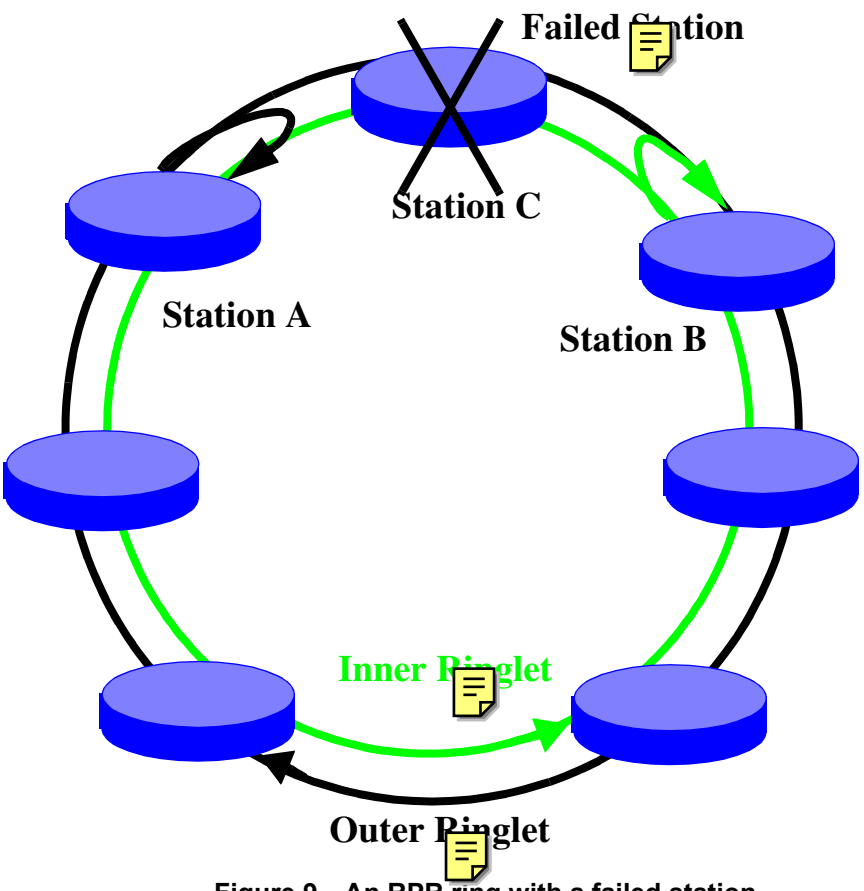


Figure 9—An RPR ring with a failed station

This sample scenario is a ring of four stations: A, B, C, and D, where station C fails. The ring is in the Idle state (all stations are Idle) prior to the failure.

11.10.3.1 Station failure (or fiber cuts on both sides of the station)

1) B detects SF on the outer ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, B, W, S} towards C on the inner ringlet/short path, and transmits {SF, B, W, L} on the outer ringlet/long path.
2) A detects SF on the inner ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, A, W, S} towards C on the outer ringlet/short path, and transmits {SF, A, W, L} on the inner ringlet/long path.
3) Steady state is reached.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

11.10.3.2 Failed station and one span return to service

Note: A station will usually return to service with one segment coming up after the other (with the time delta potentially close to 0). In this scenario, a station is powered up with the links connected and fault free.

1) Station C and a segment between A and C return to service (SF between A and C disappears).
2) Station C, not seeing any faults, starts to source idle messages {IDLE, C, I, S} in both directions.
3) The fault disappears on A, and A enters WTR state (briefly).
4) Station A receives an idle message from station C. Because the long path protection request {SF, B, W, L} received over the long span is not originating from the short path neighbor (C), station A drops the WTR.
5) Steady state is reached.

11.10.3.3 Second span returns to service

The scenario is like the bidirectional fiber cut fault clearing scenario described in 11.10.2.

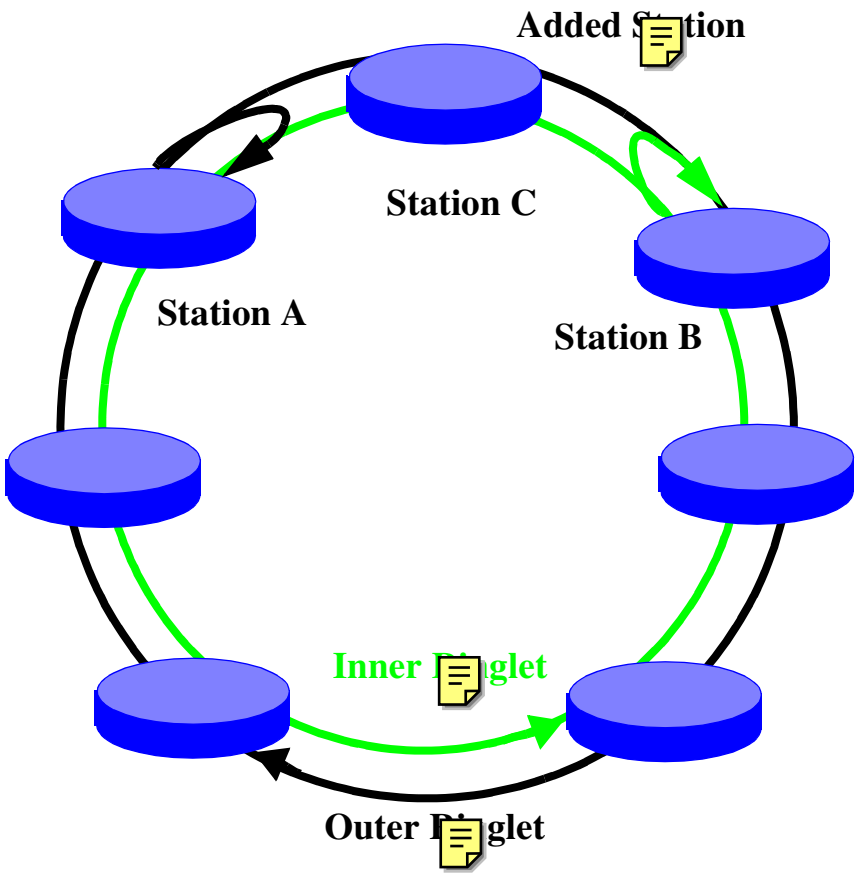


Figure 10—An RPR ring with a failed station

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

This sample scenario is a ring where stations A and B are initially connected. Subsequently, the links between stations A and B are disconnected, and a new station, C, is inserted.

11.10.3.4 Bidirectional fiber cut

1) Links are removed between stations A and B.
2) B detects SF on the outer ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, B, W, S} towards A on the inner ringlet/short path, and transmits {SF, B, W, L} on the outer ringlet/long path.
3) A detects SF on the inner ringlet, transitions to Wrapped state (performs a wrap), transmits {SF, A, W, S} towards B on the inner ringlet/short path, and transmits {SF, A, W, L} on the outer ringlet/long path.
4) As the stations on the long path between A and B receive a SF request, they enter a pass-through mode (in each direction), stop sourcing the Idle messages, and start transiting messages between A an B.
5) Steady state is reached.

11.10.3.5 Station C is powered up and fibers between stations A and C are reconnected

This scenario is identical to the returning a failed station to service scenario described in 11.10.3.2.

11.10.3.6 Second span put into service

Stations C and B are connected. The scenario is identical to the bidirectional fiber cut fault clearing scenario described in 11.10.2.2.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12. Operation administration and maintenance (OAM)

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

12.1 Overview

This section provides OAM&P (Operation, Administration, Maintenance and Provisioning) functionalists supported by RPR equipment.

Editors' Notes: *To be removed prior to final publication.*

We need to be consistent on whether we are defining "OAM" or OAM&P". The clause text does not mention any "provisioning". The outline does. The clause title doesn't. The overview does. This needs to be resolved.

Management functionals as pertinent to RPR are:

a) Configuration management
b) Fault management
c) Performance management
d) Security management (not addressed in this draft)

Editors' Notes: *To be removed prior to final publication.*

The mention of "security management" is in conflict with Clause 1, which explicitly excludes security. In addition, we either need a contribution to specify this security management, or else remove all mention of it from the text.

Configuration Management exercises control over, identifies, collects data from, and provides data to Network Elements (NEs) and the connections between NEs. Configuration Management is responsible for the installation of NEs, their interconnection into a network (configuration) and provisioning. [Note that it does not check and report configuration anomalies, this is fault management. Depending on the system used to configure the NE, it may not allow the user to enter an invalid value, but it cannot be used for checking configuration errors.]

Fault (or Maintenance) Management enables the detection, isolation, and correction of abnormal operation of the NEs and its network. It is responsible to detect and process any faults as well as to report it to the management system.

Performance Management evaluates and reports upon the behavior of NEs and the effectiveness of the network and NEs in the support of services. Performance Management provides mechanisms to measure ser-

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

vice quality, by monitoring the system performance. It also reports statistics information to the management system

In order to improve fault and performance management capability, e.g. to allow fault detection, some in-band OAM functions are envisaged.

OAM functions in a network are performed on hierarchical levels. Not all protocols support OAM, for example a physical layer based on SONET/SDH includes extensive OAM functions, while a physical layer based on Ethernet phys lacks OAM functions completely.

SONET/SDH has in-band OAM functionality specified in ITU G.783 [B3]. Based on G.783 specification, it is possible to detect Signal Fail (SF) in each span (SONET SPE or SDH VC). SONET/SDH interfaces used in RPR systems should support the OAM requirements as specified in Recommendation G.783 [B3].

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The statement about the properties of SONET/SDH (first sentence in above paragraph) needs to be removed. RPR cannot say what other standards do or don't do.*

The IEEE 802.3 standards for 1 GbE (802.3ac) and 10 GbE (802.3ae) LAN PHY interfaces have no in-band OAM functionality defined. The only information that is known is if the physical interface is available or not available. The unavailability of the physical interface directly translates into a Signal Fail (SF) condition for the MAC layer.

In-band OAM functionality for IEEE 802.3 standard 10 GbE (802.3ae) WAN PHY interface is TBD.

IEEE 802.17 RPR re-uses existing in-band OAM mechanisms provided by the PHY layers.

In-band OAM functionality used in upper layers is outside the scope of the IEEE 802.17 Standard.

12.1.1 OAM functions of the RPR layer

The OAM function in RPR is based on special frames sent between Stations. These frames correspond to flows. A flow is defined by the SA and DA.

Fault Management frames are used to indicate components, stations and ring failures, loss of continuity between ring stations, and to perform loopback operations.

Activation/Deactivation frames are used to activate or deactivate the transmission of continuity check frames. These frames allow coordinating the transmission and reception of continuity frames to avoid the generation of undesirable alarm indication.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The need for the activation/deactivation flow is TBD.*

The OAM frame types of the RPR layer are:

RDI - For reporting defect indications in the backward direction on a flow level

CC - For monitoring continuity of flows

LB - For on demand connectivity monitoring and fault localization on flows

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Activation/Deactivation - Deactivating/Reactivating CC.

Editors' Notes: *To be removed prior to final publication.*

The above list has to be reformatted.

12.2 Fault management

Fault management includes alarm surveillance, fault localization, fault correction and testing. Alarm Surveillance provides the capability to monitor failures detected in NEs. In support of alarm surveillance RPR NEs should perform checks on hardware and software in order to detect failures, and generate alarms for such failures. Upon detecting a failure, in addition to generating and sending alarms to systems, NEs should also send RDI in the backward direction, in order to notify the peer node that a failure has occurred (and some action is required).

Loss of Continuity (LOC) is the only defect defined at this stage that the NE has to detect. This is addressed by the use of a continuity check (CC) mechanism. CC also assists in fault localization, since it is possible to identify between which NEs the flow is interrupted. Another type of failure that RPR NEs may identify is software misconfiguration/failure. Such failure/misconfiguration can lead to invalid/unrecognizable header field value when the RPR frame is generated. Software checks can be performed on the RPR header to check for invalid/unrecognizable field value.

Fault Localization determines the root cause of a failure. In addition to the initial failure information, it may use failure information from other entities in order to correlate and localize the fault.

Fault Correction is responsible for the repair of a fault and for the control of procedures that use redundant resources to replace equipment or facilities that have failed. For RPR, in case of fiber cut or node failure, a protection switching is used to restore service.

Testing performs repair functions using some testing and diagnostic routines. Testing is characterized as the application of signals/messages and their measurement. Loopback is one example of a testing routine and can be activated upon request.

The fault management mechanisms are useful to check the reachability at the MAC layer between two RPR nodes on the ring, especially when there are some failures (e.g. one station on the ring steals packets addressed to some other stations) that are not detected at layer 1.

This section defines two Fault Management mechanisms.

The first mechanism is an on-demand in-service loopback mechanism (see section 1.2.3) used for troubleshooting the RPR network (reactive mechanism).

The second mechanism is a Continuity Check (CC) that is used for fault detection (proactive mechanism) by continuously running and warning the operator when there are some problems (see section 1.2.1).

As part of the Fault Management, a Remote Defect Indication (RDI) mechanism is also defined. It is used together with the CC mechanism to inform the source node of a flow that the destination node has detected a failure on that flow (see section 1.2.2). Currently, the only failure that can be notified is the Loss of Continuity (LOC).

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.2.1 RPR continuity check (CC)

Continuity check allows detecting per flow failures, such as one Station "stealing" the frames from another Station. It can also be used to verify connectivity in the protection path.

Continuity check frames transmission and reception can be activated either using activation/deactivation frames or by configuration. In order to avoid meaningless alarms, it is required that the CC frame insertion is activated in the source station before activating the CC frame detection in the destination station.

CC frames are sent with a periodicity of nominally 1 frame per second. CC frames use the highest priority class of service.

When the sink station does not receive any CC frame within a time interval of 3.5±0.5 seconds, it will declare a Loss of Continuity (LOC) defect. LOC shall be removed when a CC frame is detected.

Each side of the Station can have separate CC capability (ringlet specific CC). In this case the source station always sends the CC frames on the specified ringlet. The destination station will declare LOC on per ringlet basis.

It is also possible to activate CC without specifying the side (ringlet agnostic CC). In this case the source station sends the CC frames on the shortest path and steer the flow during ring failures. The destination station will declare LOC on per station basis.

It is recommended to activate CC bi-directional.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above recommendation should be converted to a Note.*

12.2.2 RPR remote defect indication (RDI)

The station detecting a Loss of Continuity (LOC) defect shall generate and send back to the source station of the failed flow, a RDI frame. The RDI frame shall be transmitted through the working path. The SA should be the detecting Station MAC address and the DA should be the MAC address of the station sourcing the failed flow, the ringlet on which the LOC was detected shall be indicated in the Defect Indication field.

The RDI frame shall be generated and transmitted as soon as possible after detection of the LOC defect, and shall be periodically transmitted during the defect condition. The generation frequency of the RDI frame shall be one frame per second.

The RDI frame generation shall be stopped as soon as the defect indication is removed.

The RDI frames shall be detected at the respective flow sourcing station. The RDI state shall be declared at the RDI frame detecting station as soon as a RDI frame is received. The RDI state is released when RDI frames are absent for 2.5±0.5 seconds.

12.2.3 RPR loopback capability

The IEEE 802.17 allows the management system to request an loopback operation to a specified destination in order to check the reachability of an RPR station.

The RPR loopback capability allows for a frame to be inserted at one station in the ring, and returned back by another station through the same or opposite ringlet, without impairing the normal flow operation. Loopback frames can be activated for each Class of Service.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The Loopback command can be sent through the shortest path (using the topology discovery), the clockwise or the counterclockwise ringlet. By default, if nothing is specified, the Loopback command is sent on the shortest path.

The Loopback source Station shall set the DA to the Loopback target MAC address and the SA to its own MAC address, and it shall set the function type to Loopback command.

The target Station shall perform the following operations:

a) Set the function type to Loopback response
b) Change the SA to its MAC address
c) Set the DA to the original Loopback frame SA
d) Change the ring ID
e) Copy all other received bytes to the transmit frame
f) Loopback the resulting frame according to the request filed.

The Loopback command can require the target station to reply either on the shortest path, or on the same ringlet it received the command, or on the opposite ringlet, or on the counterclockwise or on the clockwise ringlet. By default, if nothing is specified, the target station is required to reply on the shortest path.

The waiting time between the transmissions of consecutive Loopback frames on a flow shall be 5 seconds. The Loopback shall be considered unsuccessful if the Loopback frame is not returned to the source Station within seconds.

12.3 RPR Activation/Deactivation of OAM

The CC mechanism is an optional mechanism and should be activated in both the source and the destination station. The activation/deactivation of the CC mechanism on both stations can be done by the management system either on both stations or on only one of them. In the former case, the coordination between the beginning of the transmission and reception of CC frames is under the responsibility of the management system. In the latter case, an Activation/Deactivation procedure is defined to coordinate the activation between the source and destination stations.

Specifically, this initialization procedure may serve the following purposes:

a) To coordinate the beginning or end of the transmission and reception of CC
b) To establish the type of procedure
c) To specify relevant parameters (if required)

The initialization procedure may be performed either via configuration, or using the Activation/Deactivation frames.

If no response is received for an Activation/Deactivation frame within 5 seconds, the frame shall be resent. If no response is received after 3 attempts the operation shall be declared as failed.

A Station that does not support Continuity Check may respond to the relevant Activation messages with Activation Request Denied, or silently discard the Activation/Deactivation frame.

Editors' Notes: *To be removed prior to final publication.*

The need for the Activation/Deactivation OAM mechanism is an open issue. Contributions are needed.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.4 OAM frame handling during failures

Two protection schemes are used to protect the rings: Steer and Wrap.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The following description contains many statements about the functionality of protection switching and protection modes that should not appear in this clause; this clause is only concerned with how OAM works during the two different protection modes, not how the protection switching itself works. The protection-specific material should either be deleted or moved to the protection clause.*

12.4.1 Steer protection

Ringlet specific CC frames are always sent on the required ringlet irrespective of the failure conditions (they are never steered).

During a single failure of the ring all the affected flows that are using the ringlet specific CC will stop receiving the CC frames and LOC will be declared. The Steer on may mask LOC since a ring failure is declared in the path used by the CC flow.

If the ringlet specific CC was not activated in the protection path the steered path will not be protected by CC.

Ringlet agnostic CC frames are always sent on the shortest path and steered during ring failures.

During a single failure of the ring, all the affected flows that are using the ringlet agnostic CC will continue receiving the CC frames from the other ringlet and LOC will not be declared.

RDI frames are always sent on the shortest path and steered during ring failures.

Loopback frames (either commands or responses) can be sent on the shortest path or on a specified ringlet. When they have to be sent on a specified ringlet, they are always sent on such a ringlet. During ring failures that affect them, they are not steered and then lost. When they have to be sent on the shortest path, they are steered during ring failures.

12.4.2 Wrap protection

During a single failure of the ring the OAM frames are wrapped, so they will reach their original destination. No LOC will be declared and the flow will remain protected by CC.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.5 OAM frame

The OAM frame includes a common part and a function specific part. Figure 12–1 shows the general frame format.

2 OCTETS	RPR HEADER(TYPE=0x5)
6 OCTETS	DESTINATION MAC ADDRESS
6 OCTETS	SOURCE MAC ADDRESS
2 OCTETS	PROTOCOL=0x2007
2 OCTETS	HEADER CHECKSUM
1 OCTET	CONTROL VERSION(0x0)
1 OCTET	CONTROL TYPE(0x3)
2 OCTETS	CONTROL TTL
1/2 OCTET	OAM TYPE
1/2 OCTET	FUNCTION TYPE
41 OCTETS	FUNCTION SPECIFIC
4 OCTETS	FCS

Figure 12–1—OAM frame format

The OAM frames length is fixed. Padding is added to provide a minimum packet length of 42 bytes.

12.5.1 OAM Class Of Service

OAM CoS is indicated in the PRI field, and depends on the OAM frame type.

The following OAM frames use the highest priority class:

a) RDI
b) CC
c) Activation/Deactivation

Loopback frames use the CoS defined by the operation.

12.5.2 OAM type

The OAM type identifies the OAM group of the OAM frame. Table 11shows the possible values of the OAM type field.

12.5.3 Function type

This field indicates the actual function performed by this frame within the group indicated by the OAM Type. Table 11 shows the possible values of the Function Type field.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 11—OAM type field values

OAM type	Coding	Function Type	Code
Fault Management	0001	RDI	0001
		CC	0100
		LB Command	1000
		LB Response	1001
Activation/Deactivation	1000	CC	0001

12.5.4 Specific fields for OAM frames

The definition of the specific fields for the different OAM frames are provided in the sub clauses that follow.

12.5.4.1 Continuity check fault management frame

No fields are specified for the Continuity Check Fault Management frame.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *What are the default contents of the CC Fault Management frame? Are any default values required?*

12.5.4.2 RDI fault management frame

The function specific fields for RDI fault management frames are illustrated in Figure 12–1

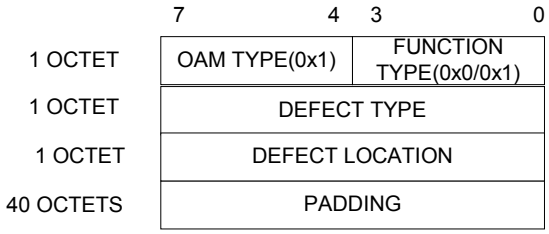


Figure 12–1—RDI fault management frame

12.5.4.2.1 Defect type

Optional field used to provide further information about the nature of the failure. Examples of this information are specified in Table 12.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *Table 14 is hardly an example; it needs to be a normative definition. Change above statement to "This field shall be encoded as in Table 14".*

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table 12—Defect Types

Defect Type	Coding
Defect not specified	11111111
Loss of Continuity (LOC) in the RPR layer.	00000000

12.5.4.2.2 Defect location

This field identifies the ringlet in which the LOC was detected. Examples of this information are specified in Table 13.

Editors' Notes: *To be removed prior to final publication.*

Table 15 is a normative definition. Change above statement to "This field shall be encoded as in Table 15".

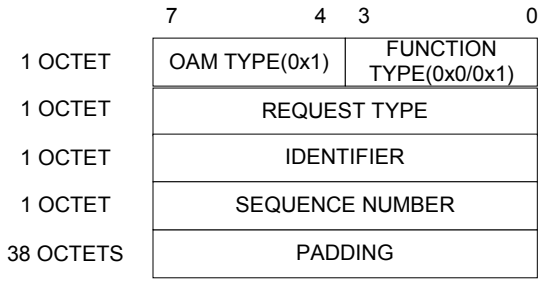
Table 13—Defect Location

Defect Location	Coding
Ringlet 0	00000000
Ringlet 1	00000001
Unspecified	11111111

12.5.4.3 Loopback frame

The function specific fields for Loopback frames are illustrated in Figure 12–1

Figure 12–1—Loopback frame



Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.5.4.3.1 Request type

The Request type field should be interpreted by the Loopback Station to decide through which interface the Loopback frame should be transmitted back to the source Station. Table 14 shows the possible values of the Loopback type and the required action.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *Table 16 is a normative definition. Change above statement to "The Loopback Request Type field shall be encoded as in Table 16".*

Table 14—Loopback Request Type values

Request Type	Action
0x00	Reply through shortest path
0x01	Reply through Inner ring
0x02	Reply through Outer ring
0x03	Reply on same ring
0x04	Reply on opposite ring

12.5.4.3.2 Identifier and sequence number

An identifier and a sequence number are generated for each loopback process so Stations can correlate loopback commands with loopback responses. The value of these fields in the loopback frame must match the value in the associated received frame. Consecutively generated identifiers and/or Sequence numbers should be different, in order to correctly correlate commands with responses.

12.5.5 Activation/Deactivation frame

The Function Type field for the Activation/Deactivation frame will be used to identify the functions. Only Function Type activation/deactivation is defined, other functions may be defined in the future.

The function specific fields for the Activation/Deactivation frame are illustrated in Figure 12–1

	7 … 4	3 … 0
1 OCTET	OAM TYPE(0x1)	FUNCTION TYPE(0x8)
1 OCTET	MESSAGE ID	
1 OCTET	IDENTIFIER	
1 OCTET	SEQUENCE NUMBER	
1 OCTET	DIRECTION OF ACTION	
38 OCTETS	PADDING	

Figure 12–1—Activation/Deactivation frame

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.5.5.1 Message ID

This field indicates the Message ID for activating or deactivating specific OAM functions. Code values for this field are shown in Table 15.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *Table 17 is a normative definition. Change above statement to "This field shall be encoded as in Table 17".*

Table 15— Message ID values

Message	Command/Response	Coding
Active	Command	00000001
Activation Confirmed	Response	00000010
Activation Request Denied	Response	00000011
Deactivate	Command	00000101
Deactivation Confirmed	Response	00000110

12.5.5.2 Identifier and sequence number

An Identifier and a Sequence number are generated for each Activation/Deactivation command so Stations can correlate commands with responses. The value of these fields in the response frame must match the value in the associated command frame. Consecutively generated Identifiers and/or Sequence numbers should be different, in order to correctly correlate commands with responses.

12.5.5.3 Direction of action

This field identifies the direction, or directions, of transmission to activate/deactivate OAM functions. The East (Inner ring sink, Outer ring source) and West (Inner ring source, Outer ring sink) notation is used to differentiate between the directions of transmission. This field is used as a parameter for the Activate and Deactivate messages. This field shall be encoded as shown in Table 16

Table 16—Direction values

Direction	Coding
East	00000010
West	00000001
Both	00000011
Unspecified	11111111
Not applicable	00000000

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

12.6 OAM frame detection procedure

OAM frames are detected through the following procedure (no specific ordering is implied):

a) Check RPR header to determine if it is a control frame of type OAM, and if it is for this station
b) Check the OAM type and Function type values according to Table 1.1 to determine the type of OAM frame received
c) Silently discard OAM frames with unsupported type or Function

12.7 OAM frames support

Editors' Notes: *To be removed prior to final publication.*

This subclause needs significant work.

All RPR compliant stations should support loopback OAM frames.

CC and RDI support is optional.

Activation/Deactivation support is optional

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

13. Layer Management Entity Interface

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

13.1 Overview of the management model

Editors' Notes: *To be removed prior to final publication.*

We need to remove ALL mention of PHY layer management from this clause. PHY layer management is outside the scope of P802.17: we do not define the details of any PHY (instead, we import the PHYs from other standards by reference), and the PHYs that RPR will use already have their own management methods and MIB objects, which we should not redefine. Therefore, the PLME and all text associated with the PLME should be deleted.

Both RPR MAC and all applicable PHY layers conceptually include management entities, called MAC sub layer management and PHY layer management entities (MLME and PLME, respectively). These entities provide the layer management service interfaces through which layer management functions may be invoked.

In order to provide correct RPR MAC operation, a station management entity (SME) must be present. The SME is a layer-independent entity that may be viewed as residing in a separate management plane. The exact functions of the SME are not specified in this standard, but in general this entity may be viewed as being responsible for such functions as the gathering of layer-dependent status from the various layer management entities, and similarly setting the value of layer-specific parameters. SME would typically perform such functions on behalf of general system management entities and would implement standard management protocols. Figure 17.1 depicts the relationship among management entities.

The management SAPs within this model are the following:

a) SME-MLME SAP
b) SME-PLME SAP

Editors' Notes: *To be removed prior to final publication.*

The use of the term "SAP" is inconsistent with other 802 standards (e.g., 802.2 LLC). SAP is normally taken to refer to a codepoint corresponding to a service client to which frames must be delivered. Here, SAP is being used to refer to an interface. The term "Service Interface" should be substituted for "SAP".

In this fashion, the model reflects what is anticipated to be a stackable implementation approach in which PLME functions are controlled by SME. In particular, different PHY implementations are required to have separate interfaces with the SME. The interfaces of the SME with the different PHYs are not part of this standard and are specified in the respective standard documents that specify the management primitives and MIBs for the different PHYs.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

=

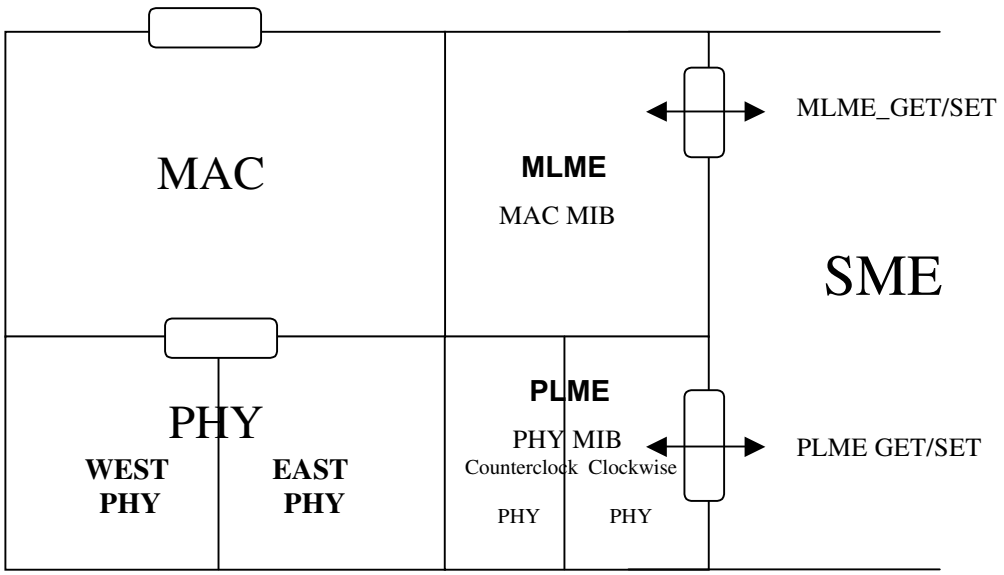


Figure 13–1—RPR MAC Management SAPs

13.2 Generic management primitives

Editors' Notes: *To be removed prior to final publication.*

This is a service interface definition, and hence the format of this subclause should be changed to correspond to that of service interface definitions elsewhere in this standard (see the MAC service interface, for instance).

The management information specific to each layer is represented as a management information base (MIB) for that layer. The MAC and PHY layer management entities are viewed as "containing" the MIB for that layer. The generic model of MIB-related management primitives exchanged across the management SAPs is to allow the SAP user-entity to either GET the value of a MIB attribute, or to SET the value of a MIB attribute. The invocation of a SET.request primitive may require that the layer entity perform certain defined actions.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The GET and SET primitives are represented as REQUESTs with associated CONFIRM primitives. These primitives are prefixed by MLME or PLME depending upon whether the MAC or PHY layer management SAP is involved.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The provision of "CONFIRM" primitives within a service interface is meaningless in the context of the abstract definition being described, and should be eliminated. No other abstract service interface definition within this standard (or other 802 standards) has such provisions.*
>
> *Also, the primitives should be formatted to be consistent with those of other abstract service interfaces within this standard (e.g., the MAC service interface).*

In the following, XX denotes MLME or PLME:

```
XX-GET.request (
   MIBattribute
   )
```

Requests the value of the given MIBattribute.

```
XX-GET.confirm (
   status,
   MIBattribute,
   MIBattributevalue
   )
```

Returns the appropriate MIB attribute value if status = "success," otherwise returns an error indication in the status field. Possible error status values include "invalid MIB attribute".

```
XX-SET.request (
   MIBattribute,
   MIBattributevalue
   )
```

Requests that the indicated MIB attribute be set to the given value. If this MIBattribute implies a specific action, then this requests that the action be performed.

```
XX-SET.confirm (
   status,
   MIBattribute
   )
```

If status = "success," this confirms that the indicated MIB attribute was set to the requested value, otherwise it returns an error condition in status field. If this MIBattribute implies a specific action, then this confirms that the action was performed. Possible error status values include "invalid MIBattribute" and "attempt to set read-only MIB attribute."

Additionally, there are certain requests (with associated confirms) that may be invoked across a given SAP that do not involve the setting or getting of a specific MIB attribute. Each SAP supports one of these, as follows:

```
XX-RESET.request: where XX is MLME or PLME as appropriate
XX-RESET.confirm
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

This service is used to initialize the management entities, the MIBs, and the data path entities. It may include a list of attributes for items to be initialized to non-default values. The corresponding.confirm indicates success or failure of the request.

Other SAP-specific primitives are in the following sections.

13.3 MLME SAP interface

The services provided by the MLME to the SME are specified in this section. These services are described in an abstract way and do not imply any particular implementation or exposed interface. MLME SAP primitives are of the general form ACTION.request followed by ACTION.confirm. The SME uses the services provided by the MLME through the MLME SAP.

According to the IETF layering principles, the RPR interface should be stacked over the two west and east span interfaces.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above gratuitous reference to "IETF layering principles" should be removed.*

13.3.1 RPR interface configuration

The RPR ring interface can be activated/deactivated for administrative purposes. Its activation/deactivation allows/forbids the upper layers to send packets on the ring. It can be activated if and only if at least one of the underlying span interfaces is activated.

The RPR interface has its own operational state that goes down when both the span interfaces go down.

The RPR MAC entity requires the MAC address. This is fixed by the vendor and can be only read for maintenance purposes.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above statements about the MAC address being fixed is implementation-specific (and, further, outside the scope of the LME clause) and should be removed.*

13.3.2 Topology discovery monitoring

The configuration management should allow monitoring, for maintenance purposes, the state of the auto-configuration and topology discovery protocols. It should also allow disabling the support of some features, even if supported by all the stations on the ring.

The detailed configuration requirements for the auto-configuration and topology discovery protocols depend from the actual mechanism that will be used and are now for further study.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *We need contributions that can be used to generate draft text for the above.*

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

If some misconfiguration condition is detected, a notification is sent to the management system for maintenance purposes. For example, the topology discovery mechanism can discover that two stations on the ring have the same MAC address.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above needs to be converted to a Note, as it is an illustrative example and not normative.*

13.3.3 Protection switching

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *This protection switching portion of the LME is inconsistent with the architecture and approach of the remainder of the LME. For example, topology discovery and MAC configuration are performed using standard MIB objects, rather than defining special primitives associated with some service interface. (In addition, defining special service interface primitives for all of these functions would quickly render the service interfaces large, incomprehensible and unwieldy.) The protection switching functions should therefore be controlled through MIB objects rather than through service interface primitives. This entire subclause should be rewritten to be consistent with other functions.*

The protection management mechanisms that will be defined in this standard are based on information known to the MAC entity. PHY layer protection mechanisms are independent and are addressed in the appropriate PHY specifications.

The configuration management should allow monitoring, for maintenance purposes, the state of the protection switching.

It should also allow activating/deactivating the RPR protection as well as activating/deactivating the usage of the Signal Degrade condition as switching criteria.

When the RPR protection is deactivated, the RPR MAC never switches under failure conditions. When the usage of the SD as a switching criteria is disabled, the RPR MAC never switches because of a signal degrade condition detected on the physical interface.

In order to support multi-layer protection mechanisms, a hold-off timer should be configured for each span interface, because it highly depends on the kind of physical network you have between two adjacent RPR nodes, if any.

In order to ensure robustness against unstable alarms, a wait time to restore (WTR) should be configured for each node, because there is no need to have a granular configuration of it.

The usage of the hold-off timer and of the wait to restore is specified in the protection switching section.

It should be possible to force a switching event for operational purposes.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *All of the above appears to mandate protection switching functions that is the province of the protection clause and not the LME clause. It should be deleted or moved to the proper location.*

This mechanism deals with the modes under which an RPR MAC switches to protection.

13.3.3.1 MLME-SWITCHTOPROTECTION.request

This primitive requests a change in the protection mode of an RPR ring.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The primitive parameters are as follows:

```
MLME-SWITCHTOPROTECTION.request (
         ReversionMode,
         Side
         )
```

This primitive is generated by the SME to implement a user request for a node to switch to protection at the east or the west side.

Name	Type	Valid range	Description
ReversionMode	**Enumeration**	**FORCED, AUTOMATIC-REVERTIVE, AUTOMATIC-NONREVERTIVE**	**An enumerated type that describes the desired way to revert from protection**
Side	**Enumeration**	**WEST, EAST**	**An enumerated type that describes the side of the ring relative to the station that will switch**

13.3.3.1.1 Effect of receipt

This request sets the reversion mode of the protection switch that occurs at the specified side. The MLME subsequently issues a MLME- SWITCHTOPROTECTION.confirm that reflects the results of the protection switch request.

13.3.3.2 MLME- SWITCHTOPROTECTION.confirm

This primitive confirms the change in protection switching mode.

The primitive parameters are as follows:

```
MLME- SWITCHTOPROTECTION.confirm (
         ResultCode
         )
```

Name	Type	Valid range	Description
ResultCode	Enumeration	SUCCESS, INVALID_PARAMETERS, NOT_SUPPORTED	Indicates the result of the MLME- SWITCHTOPROTEC-TION.request

This primitive is generated by the MLME as a result of an MLME- SWITCHTOPROTECTION.request by the SME. It is not generated until the switch is completed.

13.3.3.2.1 Effect of receipt

The SME is notified of the protection switch.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

13.3.4 Performance and Accounting Measurements

The following statistics should be kept for the RPR interface.

1) Fairness message performance monitoring.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *What does "Fairness message performance monitoring" mean? We need to be more specific here. Contributions are requested.*

2) How many frames/octets have been inserted on the ring (by the upper layer)
3) How many frames/octets have been delivered to the upper layer
4) How many frames, received from the west interface, have been stripped because originated by the node itself
5) How many frames, received from the east interface, have been stripped because originated by the node itself
6) How many transit frames, received from the west interface, have been discarded because of the TTL expiration
7) How many transit frames, received from the east interface, have been discarded because of the TTL expiration
8) How many frames, received from the west interface, have been discarded because of a bad FCS (this counter is fixed to 0 when the cut-through method is implemented)
9) How many frames, received from the east interface, have been discarded because of a bad FCS (this counter is fixed to 0 when the cut-through method is implemented)
10) How many frames addressed to the node have been discarded because of a bad FCS (this counter is fixed to 0 when the store and forward method is implemented)
11) How many frames, addressed to the node, have been discarded because of an unknown or unsupported protocol.
12) How many frames addressed to the node have been discarded even if no error has been detected (e.g. because of buffer congestion). This counter has an implementation specific meaning and in some implementations it may be always equal to 0.
13) How many frames originated by the node have been discarded (e.g. because of buffer congestion). This counter has an implementation specific meaning and in some implementations it may be always equal to 0.
14) How many frames, received from the west interface, correctly passed-through the MAC entity.
15) How many frames, received from the east interface, correctly passed-through the MAC entity.
16) How many transit frames, received from the west interface, have been discarded even if no error has been detected (e.g. because of buffer congestion). This counter has an implementation specific meaning and in some implementations it may be always equal to 0.
17) How many transit frames, received from the east interface, have been discarded even if no error has been detected (e.g. because of buffer congestion). This counter has an implementation specific meaning and in some implementations it may be always equal to 0.

For the counters defined in points 1, 2, 9, 11, 12, 13, 14, 15 and 16, there is one counter for each QoS class and a global counter.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above definitions of counters and objects are clearly not very precise. We need some more explicit definition of each counter and object, along with the specific conditions that cause these objects to be modified (and a description of how they are modified). A tabular format is preferred.*
>
> *Also, all of the above counters and objects need to be reviewed in conjunction with the functionality that would update them (as defined in the other clauses).*

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

13.3.4.1 Topology Discovery statistics

The exact requirements depend on the mechanism that is used and then are for further study.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *We need text here. Contributions are solicited.*

13.3.5 Notifications and Fault Management

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *We need text here. Contributions are solicited.*

TBD

13.3.6 RPR Ping Management

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *All of the functionality in this portion is part of the OAM clause, so we should delete the primitives and functionality from this clause. However, we need to introduce MIB objects to support this functionality within this clause.*
>
> *Also, we need to check for contradictions between the functionality defined here and that defined in the OAM clause before deleting.*

13.3.6.1 MLME-PING.request

This primitive requests the station to loopback another station on the RPR ring.

The primitive parameters are as follows:

```
MLME-PING.request (
         Addressed Station,
         Ringlet,
         Request Type,
         CoS,
         Timer
         )
```

This primitive is generated by the SME to implement a user request for a node to perform a ping operation.

Name	Type	Valid range	Description
Addressed Station	MAC Address	Any valid unicast MAC address	The MAC address of the RPR station to be ping
Ringlet	Enumeration	SHORTEST, COUNTERCLOCKWISE, CLOCKWISE	The ringlet over which the RPR ping request message should be sent (see section 16.2

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Request Type	Enumeration	SHORTEST, SAME, OPPOSITE, COUNTERCLOCKWISE, CLOCKWISE	The ringlet over which the addressed station should send the RPR ping reply message (see section 16.2)
CoS	Integer	(0Ö7)	The CoS to be used in the RPR OAM frames carrying the RPR ping request and reply messages.
Timer	Integer	(0..65535)	The number of seconds the source station should wait for the reply before declaring the ping failed.

13.3.6.1.1 Effect of receipt

This requests causes the station to send an OAM RPR ping request message.

13.3.6.2 MLME- PING.confirm

This primitive notifies the success or the failure of a ping operation.

The primitive parameters are as follows:

```
MLME- PING.confirm (
         ResultCode
         )
```

Name	Type	Valid range	Description
ResultCode	Enumeration	SUCCESS, INVALID_PARAMETERS, FAILURE	Indicates the result of the MLME- PING.request

This primitive is generated by the MLME as a result of an MLME- PING.request by the SME. It is not generated until the OAM RPR ping reply frame is received or the timer expires.

13.3.6.2.1 Effect of receipt

The SME is notified of the success or of the failure of the ping procedure.

13.4 Ring Aggregation

Editors' Notes: *To be removed prior to final publication.*

All of the statements within this subclause are not acceptable for a normative standard. (The last sentence, regarding forwarding requests to the IANA, is particularly egregious!) This entire section should be deleted. Also, the Ring Aggregation statements should be deleted from the MAC Reference Model, until contributions are received and accepted that make Ring Aggregation part of the RPR standard.

Ring aggregation is outside of the scope of this project. If Ring Aggregation for RPR Links is supported in the future then Ring Aggregation objects and a Ring Aggregation MIB will be defined. For this purpose the IEEE 802.3 link aggregation objects defined in LAG MIB (IEEE 802.3 – Annex 30C) might serve as a guide.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Link aggregation can be managed as an interface stacked over multiple RPR interfaces

A request for a new ifType for the aggregation should be forwarded to the IANA

13.5 PLME SAP interface

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *All of this subclause is outside the scope of the RPR standard. We use standard PHYs without modification. The PHYs that we use already have management objects and interfaces defined for them. Hence it is out of scope for P802.17 to specify new management objects and interfaces to these PHYs. The entire subclause should be deleted.*

Each span interface can be activated/deactivated separately for administrative purposes. Its activation/deactivation allows/forbids the MAC layer to send packets on that span.

In order to avoid inconsistent configurations, a span interface can be deactivated only if the RPR interface has been deactivated or the other span interface is still active.

Each span interface has its own operational state that can be read for maintenance purposes.

The management of Sonet/SDH PHY as well as of Ethernet physical interfaces are already defined in the relevant standard recommendations. The IEEE 802.17 will reuse the already defined PLME SAP primitives to manage the Sonet/SDH and Ethernet PHY layers.

The number frames that are received or transmitted on each span interface are counted as span interface statistics representing the number of frames that the span interface has delivered or has received from the upper layer.

13.5.1 The Ethernet PHY

For the Ethernet PHY we will use the 802.3ae LAN and WAN PHY primitives. Also we will use the MIB objects that are defined in the MAU-MIB (RFC 2668) for the Ethernet PHY objects and the updated MIB in preparation (draft-ietf-hubmib-mau-mib-v3-00) for management of 10 Gb/s LAN PHY. In addition we will use the MIB objects defined in (draft-ietf-hubmib-wis-mib-00) to manage the10 Gb/s WAN PHY – called ETHER-WIS-MIB.

The Ethernet Reconciliation Sublayer management should be defined in the IEEE 802.17 specification.

According to the IETF layering principles, the RPR interface, when working over Ethernet interfaces, should be stacked over the west and east "Ethernet" interfaces.

An Ethernet span interface can be activated at any time, because it is the lowest level interface on the system. Its operational state goes down when the media is unavailable.

13.5.2 The SONET PHY

Sonet/SDH interfaces are layered interfaces and are managed as a set of stacked interfaces. The physical medium, the section and the line layers are managed as a single layered interface. The recommended IANA ifType is sonet (39). The Path layer is managed as a stacked interface over the Medium/Section/Line interface. The recommended IANA ifType is sonetPath (50). For the SONET PHY the SONET/SDH PHY objects are defined in the SONET-MIB (RFC 2558).

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

13.5.2.1 The GFP Adaptation Layer

An additional GFP interface might be stacked over the Path interface to represent the GFP adaptation layer.

The GFP management is outside the scope of IEEE 802.17.

According to the IETF layering principles, the RPR interface, when working over Sonet/SDH interfaces, should be stacked over the west and east GFP interfaces.

A GFP span interface can be activated only if the underlying Sonet/SDH interface is activated. Its operational state goes down when a signal fail condition is detected on the Sonet/SDH path or if there is a payload mismatch (the value in the received C2 byte is different than the GFP code, i.e. [TBD] value) or there is a loss of frame alignment.

[Editor's note – The signal label value to be put in the C2 byte of Sonet/SDH interfaces has not yet been assigned by ITU-T.]

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex A

(informative)

Bibliography

Editor's Note: *To be removed prior to final publication.*
As normative references are supplied in clause 2, it is not clear that this annex is required.

[B1] IEEE 802.3 - 2000 Edition Carrier sense multiple access with collision detection (CSMA/CD) MAC and physical layer specification

[B2] ITU-T G.707 Network Node Interface for the Synchronous Digital Hierarchy (SDH).

[B3] ITU-T G.774 Series SDH Management Information Model for the Network Element View

[B4] ITU-T G.783 Characteristics of Synchronous Digital Hierarchy (SDH) Equipment Functional Blocks

[B5] ITU-T G.7041 Generic Framing Procedure

[B6] RFC 2615 PPP over SONET/SDH

[B7] RFC 1471 The Definitions of Managed Objects for the Link Control Protocol of the Point-to-Point Protocol

[B8] RFC 1661 The Point-to-point Protocol (PPP)

[B9] RFC 1662 PPP in HDLC-like Framing

[B10] RFC 2863 The Interfaces Group MIB

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex B

(normative)

Transmit clock synchronization

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002 Initial draft document for RPR WG review.

Editors' Note: *To be removed prior to final publication.*

Editors suggest that "transmit clock synchronization" is a normative MAC function, and belongs in a MAC clause instead of a separate annex. Comments and suggestions are welcome.

Editors' Note: *To be removed prior to final publication.*

This text does not appear to fully describe the requirements for RPR "Transmit Clock Synchronization". Editors request comments or presentations providing specific, additional details.

Each node operates in "free-run" mode. That is, the receive clock is derived from the incoming receive stream while the transmit clock is derived from a local oscillator. This eliminates the need for expensive clock synchronization as required in existing SONET networks. Differences in clock frequency are accommodated by inserting a small amount of idle bandwidth at each node's output.

The clock source for the transmit clock shall be selected to deviate by no more than 200 ppm from the center frequency. The overall outgoing rate of the node shall be rate shaped to accommodate the worst case difference between receive and transmit clocks of adjacent nodes. This is accomplished by monitoring the input data rate (from the line and the MAC client), and comparing that to the output data rate. If the rates differ, it can be assumed that there are differences between the clocks, and the output data rate can be adjusted appropriately.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex C

(normative)

Ethernet reconciliation sublayers

Editors' Notes: *To be removed prior to final publication.*

References:
IEEE Standard 802.3, IEEE P802.3ae.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002 Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

Text has been modified from the baseline document:
- Add definition for gigabit Ethernet reconciliation sublayer
- Update ten gigabit Ethernet reconciliation sublayer by referencing P802.3ae and removing unnecessary text.
These changes are intended to reflect the intent of the motion that was passed at the 802.17 interim meeting in January. Concerns or comments are welcome.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.1 Overview

C.1.1 Scope

This annex defines two reconciliation sublayers for use with Ethernet physical layer entities (PHYs). The first is a Gigabit Ethernet Reconciliation Sublayer (GERS) that maps the logical primitives at the RPR MAC physical layer service interface to the Gigabit Media Independent Interface (GMII), for use with gigabit Ethernet PHYs. The GMII and the gigabit Ethernet PHYs are defined in IEEE Standard 802.3. The second is a Ten Gigabit Ethernet Reconciliation Sublayer (XGERS) that maps the logical primitives at the RPR MAC physical layer service interface to the Ten Gigabit Media Independent Interface (XGMII), for use with ten gigabit Ethernet PHYs. Optionally, an XGMII Extender Sublayer (XGXS) may be used to provide a Ten Gigabit Attachment Unit Interface (XAUI) instead of the XGMII. The XAUI, XGMII, XGXS, and ten gigabit Ethernet PHYs are defined in IEEE P802.3ae.

C.2 Gigabit Ethernet Reconciliation Sublayer (GERS)

The Gigabit Ethernet Reconciliation Sublayer (GERS) converts the logical physical layer service interface of the RPR MAC to the Gigabit Media Independent Interface (GMII) defined in IEEE Standard 802.3.

C.2.1 General requirements

C.2.1.1 Summary of major concepts

c) The GERS maps the signals provided at the GMII to the logical physical layer service interface primitives provided at the MAC;
d) Each direction of data transfer is independent and serviced by data, delimiter, error, and clock signals;
e) In the transmit direction, the GERS accepts frames from the MAC, inserts preamble and interframe idle, and generates corresponding signals on the GMII;
f) In the receive direction, the GERS receives signals from the GMII, removes preamble and interframe idle, and conveys frames to the MAC.

C.2.1.2 Relationship to other sublayers

The relationship of the GERS to RPR and IEEE Standard 802.3 sublayers is shown in Figure C.1.

C.2.1.3 Rate of operation

The GMII is capable of supporting a data rate of 1000 Mbps. This does not correspond to a fixed MAC data rate, because preamble and interframe idle are generated by the XGERS.

Editors' Notes: *To be removed prior to final publication.*

A requirement for "delay constraints" appears to be needed (similar to IEEE 802.3), because an RPR network has similar requirements to ensure stable and predictable operation.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

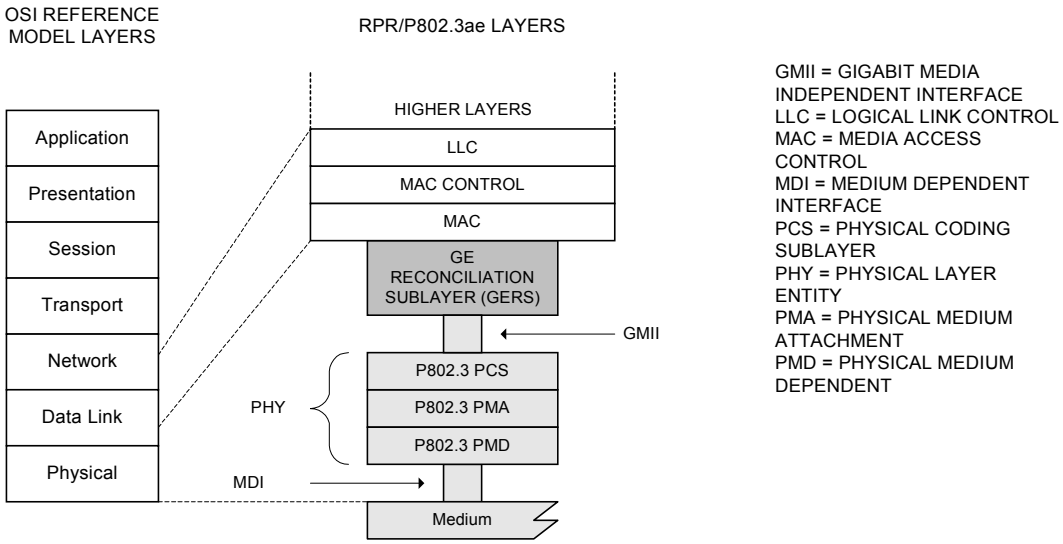


Figure 0.1—GERS inputs and outputs

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.2.1.4 GMII structure

The GMII is composed of independent transmit and receive paths. Each direction uses 8 data signals (TXD<7:0> and RXD<7:0>), enable signals (TX_EN and RX_EN), coding error signals (TX_ER and RX_ER), and a clock (GTX_CLK and RX_CLK). Figure C.2 shows a schematic view of the GERS inputs and outputs. The GMII also includes carrier sense (CRS) and collision detected (COL) signals that are not defined for the GERS since only full duplex PHYs are supported.

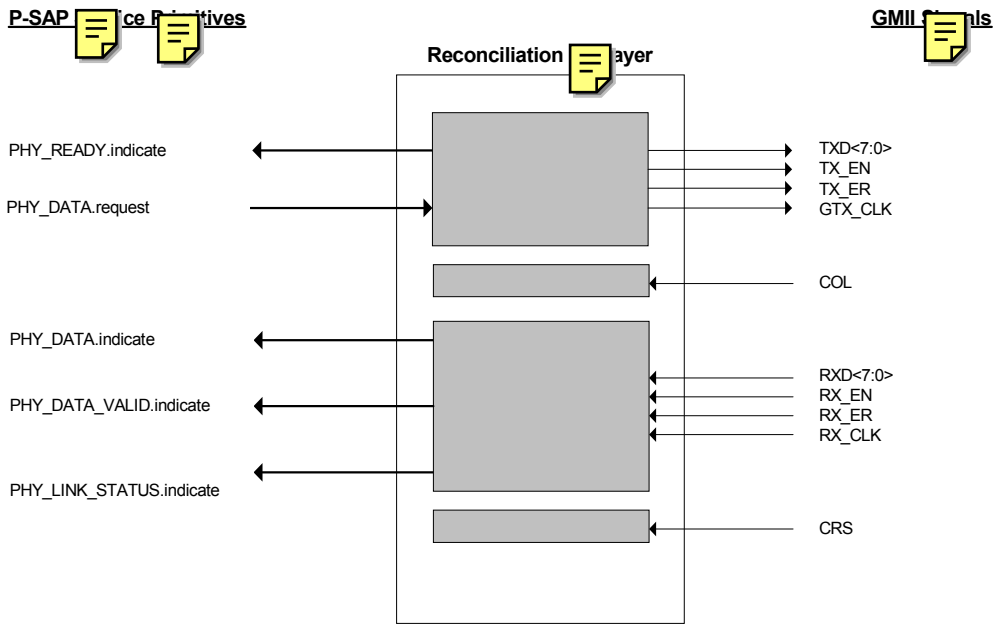


Figure 0.2—GERS inputs and outputs

C.2.1.5 Mapping of GMII signals to service interface primitives

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that the "mapping of signals to service interface primitives" should be moved to Clause 7, since the service interface is common to all reconciliation sublayers. The annexes will include only details specific to that annex.

The GERS shall map the signals provided at the GMII to the MAC physical layer service interface primitives defined in Clause 7. Mappings for the following primitives are defined:

PHY_DATA.request
PHY_DATA.indicate
PHY_DATA_VALID.indicate
PHY_LINK_STATUS.indicate
PHY_READY.indicate.

C.2.1.5.1 Mapping of PHY_DATA.request

C.2.1.5.1.1 Function

Map the primitive PHY_DATA.request to the GMII signals TXD<7:0>, TX_EN, TX_ER, and GTX_CLK.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.2.1.5.1.2 Semantics of the service primitive

PHY_DATA.request (OUTPUT_FRAME, LENGTH)

The OUTPUT_FRAME parameter either takes the value of a complete RPR frame or NO_FRAME, and represents the transfer of a complete RPR frame from the MAC to the GERS. The value NO_FRAME indicates that the MAC has no frame for transfer.

The LENGTH parameter is not used by the GERS. The value of this parameter is undefined.

C.2.1.5.1.3 When generated

This primitive is generated by the MAC sublayer to request the transmission of a frame on the physical medium, or to indicate that no frame is available for transmission.

C.2.1.5.1.4 Effect of receipt

The OUTPUT_FRAME is conveyed to the PHY using the signals TXD<7:0> when TX_EN is asserted. The data signals transition synchronously with GTX_CLK. TX_EN is deasserted to indicate that valid data is not present. EXTEND and EXTEND_ERROR functions are not supported. TX_ER is only asserted to indicate the propagation of a transmit error.

The GERS shall prepend a preamble and generate an interpacket gap for each frame as described in C.2.3.1.

Editors' Notes: *To be removed prior to final publication.*

The 802.3 "EXTEND" functions are currently not defined, but need to be added if copper GE PHYs are to be supported. Comments are welcome.

C.2.1.5.2 Mapping of PHY_DATA.indicate

C.2.1.5.2.1 Function

Map the primitive PHY_DATA.indicate to the GMII signals RXD<7:0>, RX_DV, RX_ER, and RX_CLK.

C.2.1.5.2.2 Semantics of the service primitive

PHY_DATA.indicate (INPUT_FRAME, LENGTH)

The INPUT_FRAME parameter either takes the value of a complete RPR frame or NO_FRAME, and represents the transfer of a complete RPR frame from the XGERS to the MAC. The value NO_FRAME indicates that the RS does not have a frame for transfer. EXTEND is not supported by the GERS.

The LENGTH parameter is not used by the GERS. The value of this parameter is null.

C.2.1.5.2.3 When generated

The INPUT_FRAME is derived from the signals RXD<7:0> while RX_DV is asserted. Each primitive generated to the MAC sublayer entity corresponds to a PHY_DATA.request issued by the MAC at the other end of the link connecting two RPR stations. The INPUT_FRAME consists of the complete RPR frame beginning seven bytes following a valid Start control character, and continuing until a RX_DV is deasserted. Synchronization between the GERS and the PHY is achieved using the RX_CLK signal.

C.2.1.5.2.4 Effect of receipt

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.2.1.5.3 Mapping of PHY_DATA_VALID.indicate

C.2.1.5.3.1 Function

Map the primitive PHY_DATA_VALID.indicate to the XGMII signals RXD<7:0>, RX_DV, RX_ER, and RX_CLK.

C.2.1.5.3.2 Semantics of the service primitive

PHY_DATA_VALID.indicate (DATA_VALID_STATUS)

The DATA_VALID_STATUS parameter can take one of two values: DATA_VALID or DATA_NOT_VALID. The DATA_VALID value indicates that the INPUT_FRAME parameter of the PHY_DATA.indicate primitive contains a valid frame. The DATA_NOT_VALID value indicates that the INPUT_FRAME parameter of the PHY_DATA.indicate primitive does not contain valid data of a received frame.

C.2.1.5.3.3 When generated

The PHY_DATA_VALID.indicate service primitive shall be generated by the GERS whenever the DATA_VALID_STATUS parameter changes from DATA_VALID to DATA_NOT_VALID or vice versa.

DATA_VALID_STATUS shall assume the value DATA_VALID when a PHY_DATA.indicate transaction is generated in response to reception of a complete frame beginning with the assertion of RX_DV and ending with the deassertion of RX_DV. If RX_DV and RX_ER are both asserted during frame reception, the GERS shall ensure that the MAC will detect a FrameCheckError in that frame (see C.2.3.3).

C.2.1.5.3.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.2.1.5.4 Mapping of PHY_LINK_STATUS.indicate

Editors' Notes: *To be removed prior to final publication.*

Mapping of PHY_LINK_STATUS to the GMII is TBD. Comments are welcome.

C.2.1.5.4.1 Function

Map the primitive PHY_LINK_STATUS.indicate to the GMII.

C.2.1.5.4.2 Semantics of the service primitive

PHY_LINK_STATUS.indicate (LINK_STATUS)

The LINK_STATUS parameter takes the value of OK, FAIL or DEGRADE, and signifies the status of the PHY. OK indicates the absense of a link_fault condition. FAIL indicates the presence of a link_fault condition. DEGRADE is not generated by the XGERS.

C.2.1.5.4.3 When generated

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

TBD.

C.2.1.5.4.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.2.1.5.5 Mapping of PHY_READY.indicate

C.2.1.5.5.1 Function

Map the primitive PHY_READY.indicate to the GMII signals TXD<7:0>, TX_EN, TX_ER, and GTX_CLK.

C.2.1.5.5.2 Semantics of the service primitive

PHY_READY.indicate (READY_STATUS)

The READY_STATUS parameter takes the value of READY or NOT_READY, and signifies whether the GERS is able to accept a frame from the MAC for transmission on the GMII.

C.2.1.5.5.3 When generated

PHY_READY.indicate is generated in response to PHY_DATA.request to indicate the beginning of frame transmission by the GMII, and the completion of frame transmission by the GMII. READY_STATUS assumes the value NOT_READY if a frame has been received from the MAC, and a preamble, the frame, and a minimum interpacket gap have not been fully transmitted by the GMII. READY_STATUS assumes the value READY any time a preamble, frame, or minimum interpacket gap are not being transmitted.

C.2.1.5.5.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.2.2 GMII data stream

Data frames transmitted through the GMII are transferred within the GMII data stream. The data stream is a combination of data bytes and control signals.

The GMII datastream shall meet the requirements of IEEE Standard 802.3, Clause 35, section 35.2.3, with the exception of sections 35.2.3.1 and 35.2.3.5. Carrier extension is not defined for the GERS.

C.2.3 Functional specifications

C.2.3.1 Transmit

C.2.3.1.1 General requirements

The GERS shall meet the transmit functional requirements specified in IEEE Standard 802.3, Clause 35, sections 35.2.2.1, 35.2.2.3, 35.2.2.4, and 35.2.2.5.

C.2.3.1.2 Preamble

The MAC RPR frame does not contain a preamble. The GERS shall prepend a preamble to each frame transmitted on the GMII in accordance with IEEE Standard 802.3, Clause 35, section 35.2.3.2.1.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.2.3.1.3 Interpacket gap

The RPR MAC does not transmit interpacket gap or idle characters. The GERs shall insert a minimum interpacket gap including the Terminate control character in accordance with IEEE Standard 802.3, Clause 4 for operation at 1000 Mbps.

C.2.3.2 Receive

The GERS shall meet the receive functional requirements specified in IEEE Standard 802.3, Clause 35, sections 35.2.2.2, 35.2.2.6, 35.2.2.7, and 35.2.2.8.

C.2.3.3 Error and fault handling

The GERS shall meet the error and fault handling functional requirements specified in IEEE Standard 802.3, Clause 35, sections 35.2.1.5 and 35.2.1.6.

C.2.3.4 CRC (carrier sense) and COL (collision) signals

The GERS supports only full duplex PHYs. The behavior of the CRC and COL signals is undefined.

C.2.4 Electrical Characteristics

The GMII provided by the GERS shall meet the electrical requirements of IEEE Standard 802.3, Clause 35, section 35.4.

C.3 Ten Gigabit Ethernet Reconciliation Sublayer (XGERS)

C.3.1 General requirements

The Ten Gigabit Ethernet Reconciliation Sublayer (XGERS) converts the logical physical layer service interface of the RPR MAC to the Ten Gigabit Independent Interface (XGMII). Optionally, an XGMII Extender Sublayer (XGXS) may be used to provide a Ten Gigabit Attachment Unit Interface (XAUI) instead of the XGMII. The XAUI, XGMII, and XGXS are defined in IEEE P802.3ae. Though the XGMII is an optional interface, it is used as a basis for specification of the XGERS, and an implementation using the XGXS to provide a XAUI shall behave functionally as if the XGMII were implemented.

C.3.1.1 Summary of major concepts

a) The XGERS maps the signal set provided at the XGMII to the physical layer service interface primitives provided at the MAC;
b) Each direction of data transfer is independent and serviced by data, control, and clock signals;
c) In the transmit direction, the XGERS accepts frames from the MAC, inserts Start control characters, preamble, Terminate control characters, and interpacket Idle control characters to generate continuous data or control characters on the XGMII;
d) In the receive direction, the XGERS expects continuous data or control characters from the XGMII, removes Start control characters, preamble, Terminate control characters, and interpacket Idle control characters and conveys frames to the MAC;
e) The XGERS participates in link fault detection and reporting;
f) The XGERS may use an XGXS to provide a XAUI instead of the XGMII.

C.3.1.2 Relationship to other sublayers

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The relationship of the XGERS to RPR and IEEE P802.3ae sublayers is shown in Figure C.3..

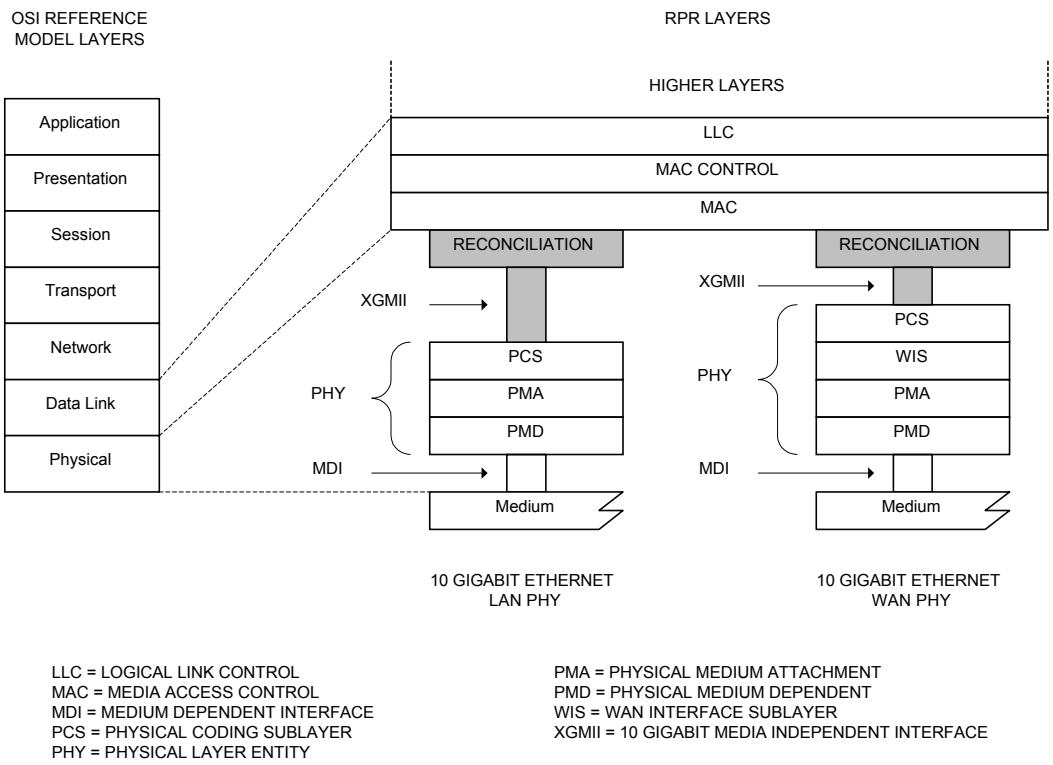


Figure 0.3—XGERS inputs and outputs

C.3.1.3 Rate of operation

The XGMII transfers data or control characters at a fixed rate of 10.0 Gbps. This does not correspond to a fixed MAC data rate, because preamble and interpacket characters are generated by the XGERS.

Editors' Notes: *To be removed prior to final publication.*

A requirement for "delay constraints" appears to be needed (similar to IEEE 802.3), because an RPR network has similar requirements to ensure stable and predictable operation.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.3.1.4 XGMII structure

The XGMII is composed of independent transmit and receive paths. Each direction uses 32 data signals (TXD<31:0> and RXD<31:0>), four control signals (TXC<3:0> and RXC<3:0>) and a clock (TX_CLK and RX_CLK). Figure C.4 shows a schematic view of the XGERS inputs and outputs.

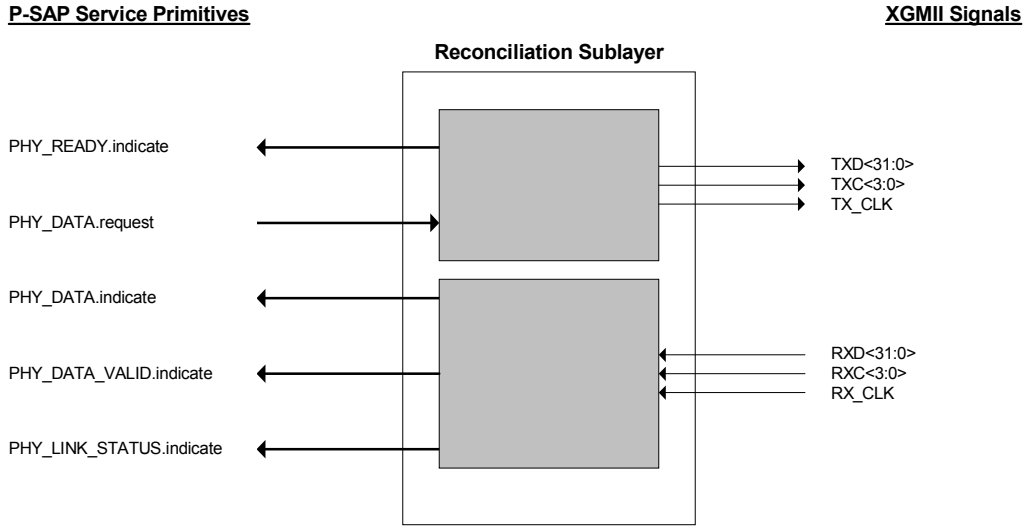


Figure 0.4—XGERS inputs and outputs

The 32 data and four control signals in each direction shall be organized into four lanes, as shown in Table C.3. The four lanes in each direction share a common clock—TX_CLK for transmit and RX_CLK for receive. The four lanes are used in round-robin sequence to carry the datastream. The first octet is aligned to lane 0, the second to lane 1, the third to lane 2, the fourth to lane 3, and then repeating with the fifth to lane 0, etc. Delimiters and interframe idle characters are encoded on the TXD and RXD signals with the control code indicated by assertion of TXC and RXC respectively.

Table 0.1—Transmit and receive lane associations

TXD RXD	TXC RXD	Lane
<7:0>	<0>	0
<15:8>	<1>	1
<23:16>	<2>	2
<32:24>	<3>	3

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.3.1.5 Mapping of XGMII signals to MAC physical layer service interface primitives

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that the "mapping of signals to service interface primitives" should be moved to Clause 7, since the service interface is common to all reconciliation sublayers. The annexes will include only details specific to that annex.

The XGERS shall map the signals provided at the XGMII to the MAC physical layer service interface primitives defined in Clause 7. Mappings for the following primitives are defined:

PHY_DATA.request
PHY_DATA.indicate
PHY_DATA_VALID.indicate
PHY_LINK_STATUS.indicate
PHY_READY.indicate.

C.3.1.5.1 Mapping of PHY_DATA.request

C.3.1.5.1.1 Function

Map the primitive PHY_DATA.request to the XGMII signals TXD<31:0>, TXC<3:0> and TX_CLK.

C.3.1.5.1.2 Semantics of the service primitive

PHY_DATA.request (OUTPUT_FRAME, LENGTH)

The OUTPUT_FRAME parameter either takes the value of a complete RPR frame or NO_FRAME, and represents the transfer of a complete RPR frame from the MAC to the XGERS. The value NO_FRAME indicates that the MAC has no frame for transfer.

The LENGTH parameter is not used by the XGERS. The value of this parameter is undefined.

C.3.1.5.1.3 When generated

This primitive is generated by the MAC sublayer to request the transmission of a frame on the physical medium, or to indicate that no frame is available for transmission.

C.3.1.5.1.4 Effect of receipt

The OUTPUT_FRAME is conveyed to the PHY by the signals TXD<31:0> and TXC<3:0> on each TX_CLK edge. A frame transferred by PHY_DATA.request transaction shall be mapped to TXD signals in sequence (TXD<0:7>,... TXD<24:31>, TXD<0:7>). The XGERS shall prepend a Start control character and preamble to each frame as described in C.3.3.1.2, and append a Terminate control character and Idle control characters to each frame as described in C.3.3.1.3.

C.3.1.5.2 Mapping of PHY_DATA.indicate

C.3.1.5.2.1 Function

Map the primitive PHY_DATA.indicate to the XGMII signals RXD<31:0>, RXC<3:0> and RX_CLK.

C.3.1.5.2.2 Semantics of the service primitive

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

PHY_DATA.indicate (INPUT_FRAME, length)

The INPUT_FRAME parameter either takes the value of a complete RPR frame or NO_FRAME, and represents the transfer of a complete RPR frame from the XGERS to the MAC. The value NO_FRAME indicates that the RS does not have a frame for transfer.

The LENGTH parameter is not used by the XGERS. The value of this parameter is null.

C.3.1.5.2.3 When generated

The INPUT_FRAME is derived from the signals RXD<31:0> and RXC<3:0> received from the PHY on each edge of RX_CLK. Each primitive generated to the MAC sublayer entity corresponds to a PHY_DATA.request issued by the MAC at the other end of the link connecting two RPR stations. The INPUT_FRAME consists of the complete RPR frame beginning seven bytes following a valid Start control character, and continuing until a Terminate control character. The Start control character, preamble, Terminate control character, and Idle control characters are not part of an RPR frame, and are removed prior to transfer of a frame to the MAC.

To assure robust operation, the value of the data transferred to the MAC may be changed by the XGERS as required by XGMII error indications (see C.3.3.3). Sequence ordered_sets are not indicated to the MAC (see C.3.3.3).

C.3.1.5.2.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.3.1.5.3 Mapping of PHY_DATA_VALID.indicate

C.3.1.5.3.1 Function

Map the primitive PHY_DATA_VALID.indicate to the XGMII signals RXC<3:0> and RXD<31:0>.

C.3.1.5.3.2 Semantics of the service primitive

PHY_DATA_VALID.indicate (DATA_VALID_STATUS)

The DATA_VALID_STATUS parameter can take one of two values: DATA_VALID or DATA_NOT_VALID. The DATA_VALID value indicates that the INPUT_FRAME parameter of the PHY_DATA.indicate primitive contains a valid frame. The DATA_NOT_VALID value indicates that the INPUT_FRAME parameter of the PHY_DATA.indicate primitive does not contain valid data of a received frame.

C.3.1.5.3.3 When generated

The PHY_DATA_VALID.indicate service primitive shall be generated by the XGERS whenever the DATA_VALID_STATUS parameter changes from DATA_VALID to DATA_NOT_VALID or vice versa.

DATA_VALID_STATUS shall assume the value DATA_VALID when a PHY_DATA.indicate transaction is generated in response to reception of a complete frame beginning with a Start control character on lane 0 if the prior RXC<3:0> and RXD<31:0> contained four Idle characters or a Sequence ordered set, and ending when RXC of any lane is asserted for anything except an Error control character. In the absence of errors, DATA_NOT_VALID is caused by a Terminate control character. When DATA_VALID_STATUS changes from DATA_VALID to DATA_NOT_VALID because of a control character other than Terminate, the

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

XGERS shall ensure that the MAC will detect a FrameCheckError prior to indicating DATA_NOT_VALID to the MAC (see C.3.3.3).

C.3.1.5.3.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.3.1.5.4 Mapping of PHY_LINK_STATUS.indicate

C.3.1.5.4.1 Function

Map the primitive PHY_LINK_STATUS.indicate to the XGMII signals RXD<31:0>, RXC<3:0> and RX_CLK.

C.3.1.5.4.2 Semantics of the service primitive

PHY_LINK_STATUS.indicate (LINK_STATUS)

The LINK_STATUS parameter takes the value of OK, FAIL or DEGRADE, and signifies the status of the PHY as indicated by Link Fault Signaling (see C.3.3.3). OK indicates the absense of a link_fault condition. FAIL indicates the presence of a link_fault condition. DEGRADE is not generated by the XGERS.

C.3.1.5.4.3 When generated

The PHY_LINK_STATUS.indicate service primitive shall be generated by the XGERS whenever there is a change in the link_fault variable defined in C.3.3.3. LINK_STATUS assumes the value OK when the link_fault variable has the value OK. LINK_STATUS assumes the value FAIL when the link_fault variable has the values Local Fault or Remote Fault.

C.3.1.5.4.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.3.1.5.5 Mapping of PHY_READY.indicate

C.3.1.5.5.1 Function

Map the primitive PHY_READY.indicate to the XGMII signals TXD<31:0>, TXC<3:0> and TX_CLK.

C.3.1.5.5.2 Semantics of the service primitive

PHY_READY.indicate (READY_STATUS)

The READY_STATUS parameter takes the value of READY or NOT_READY, and signifies whether the XGERS is able to accept a frame from the MAC for transmission on the XGMII.

C.3.1.5.5.3 When generated

PHY_READY.indicate is generated in response to PHY_DATA.request to indicate the beginning of frame transmission by the XGMII, and the completion of frame transmission by the XGMII. READY_STATUS assumes the value NOT_READY if a frame has been received from the MAC, and a Start control character, preamble, the frame, a Terminate control character, and the minimum interpacket gap have not been fully transmitted by the XGMII. READY_STATUS assumes the value READY any time a Start control character, preamble, a frame, a Terminate control character, or the minimum interpacket gap are not being transmitted.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

C.3.1.5.5.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.3.2 XGMII data stream

Data frames transmitted through the XGMII are transferred within the XGMII data stream. The data stream is a sequence of bytes, where each byte conveys either a data octet or control character.

The XGMII datastream shall meet the requirements of IEEE P802.3ae, Clause 46, section 46.2, with the exception of references to the MAC interpacket gap in section 46.2.1. The interpacket gap for the RPR XGERS is defined in C3.3.1.3.

C.3.3 Functional specifications

C.3.3.1 Transmit

C.3.3.1.1 General requirements

The XGERS shall meet the transmit functional requirements specified in IEEE P802.3ae, Clause 46, section 46.3.1.1 through 46.3.1.3.

C.3.3.1.2 Preamble

The MAC RPR frame does not contain a preamble. The XGERS shall prepend a preamble including a Start control character to each frame transmitted on the XGMII in accordance with IEEE P802.3ae, Clause 46, section 46.2.2.

C.3.3.1.3 Interpacket gap

The RPR MAC does not transmit interpacket gap or idle characters. The XGERs shall insert a minimum interpacket gap including the Terminate control character in accordance with IEEE Standard 802.3, Clause 4 for operation at 10 Gbps, and as described in IEEE P802.3ae, Clause 46, Section 46.3.1.4.

C.3.3.2 Receive

The XGERS shall meet the receive functional requirements specified in IEEE P802.3ae, Clause 46, section 46.3.2.

C.3.3.3 Error and fault handling

The XGERS shall meet the error and fault handling functional requirements specified in IEEE P802.3ae, Clause 46, section 46.3.3.

C.3.4 Electrical characteristics

The XGMII provided by the XGERS shall meet the electrical requirements specified in IEEE P802.3ae, Clause 46, section 46.4.

C.3.5 XGXS and XAUI

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The XGMII Extender Sublayer (XGXS) may be implemented to provide a Ten Gigabit Attachment Unit Interface (XAUI) instead of the XGMII.

If implemented, the XGXS and XAUI shall meet all requirements specified in Clause 47 of IEEE P802.3ae.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text was updated and has been deleted:

Note: Text is largely borrowed from P802.3ae/D3.1, and needs to be cleaned up. Need to insert figures. XGMII remains unchanged, RS has been modified for RPR.

C.4 Overview

This clause defines the logical and electrical characteristics for the Reconciliation Sublayer (RS) and 10 Gigabit Media Independent Interface (XGMII) between the RPR media access control (MAC) layer and various 10 Gigabit Ethernet PHYs. Figure 2 shows the relationship of the RS and XGMII to the ISO (IEEE) OSI reference model.

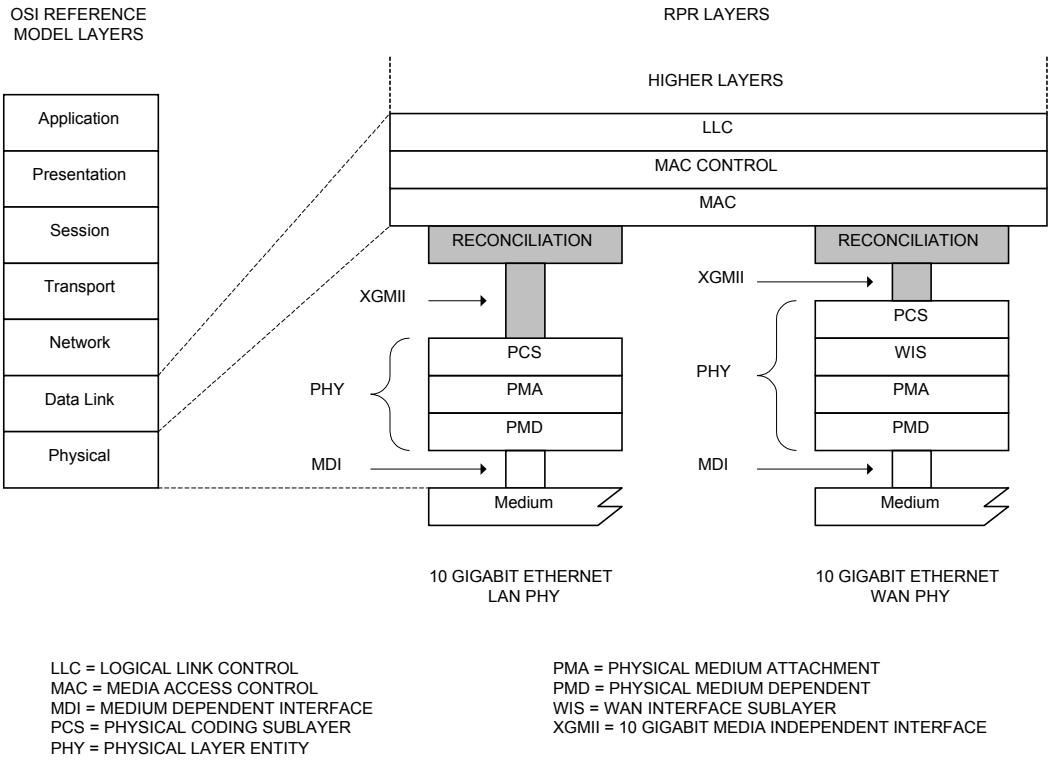


Figure A.1—RS and XGMII location in the OSI protocol stack

The purpose of the RS is to convert the logical P-SAP interface of the MAC to the XGMII interface that is specified by P802.3ae as an interface to 10 Gigabit PHYs. Though the XGMII is an optional interface, it is used extensively in this standard as a basis for specification. The 10 Gb/s Physical Coding Sublayer (PCS) is specified to the XGMII interface, so if not implemented, a conforming implementation shall behave functionally as if the RS and XGMII were implemented.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

The purpose of the 10 Gigabit Media Independent Interface (XGMII) is to provide a simple, inexpensive, and easy-to-implement interconnection between the Media Access Control (MAC) sublayer and the Physical layer (PHY). The 10 Gigabit Attachment Unit Interface (XAUI) may optionally be used to extend the operational distance of the XGMII with reduced pin count.

The RS adapts the logical P-SAP interface of the MAC to the parallel encoding of 10 Gb/s PHYs. Though the XGMII is an optional interface, it is used extensively in this standard as a basis for specification. The 10 Gb/s Physical Coding Sublayer (PCS) is specified to the XGMII interface, so if not implemented, a conforming implementation shall behave functionally as if the RS and XGMII were implemented.

The XGMII has the following characteristics:

a) It is capable of supporting 10 Gb/s operation;
b) Data and delimiter are synchronous to clock reference;
c) It provides independent 32-bit-wide transmit and receive data paths;
d) It uses signal levels compatible with common digital ASIC processes;
e) It provides for full-duplex operation only

C.4.1 Summary of major concepts

a) The RS converts the P-SAP service primitives to the parallel data paths of the XGMII;
b) The RS maps the signal set provided at the XGMII to the P-SAP service primitives provided at the MAC;
c) Each direction of data transfer is independent and serviced by data, control, and clock signals;
d) The RS generates continuous data or control characters on the transmit path and expects continuous data or control characters on the receive path;
e) The RS participates in link fault detection and reporting by monitoring the receive path for status reports that indicate an unreliable link;
f) When the XGMII is optionally extended with XAUI, two XGMII interfaces logically exist (see Figure 46–1). When extended, signals on the transmit path are from the RS to an XGXS of the XAUI interface on one XGMII and from the XGXS to the PHY on the other XGMII. When extended, signals on the receive path are from the PHY to an XGXS of the XAUI interface on one XGMII and from the XGXS to the RS on the other XGMII. The descriptions of the XGMII as between the RS and the PHY are therefore equally applicable between the RS and the XGXS or the XGXS and the PHY.

C.4.2 Summary of major concepts

a) The RS converts the P-SAP service primitives to the parallel data paths of the XGMII;
b) The RS maps the signal set provided at the XGMII to the P-SAP service primitives provided at the MAC;
c) Each direction of data transfer is independent and serviced by data, control, and clock signals;
d) The RS generates continuous data or control characters on the transmit path and expects continuous data or control characters on the receive path;

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

e) The RS participates in link fault detection and reporting by monitoring the receive path for status reports that indicate an unreliable link;

f) When the XGMII is optionally extended with XAUI, two XGMII interfaces logically exist (see Figure 46–1). When extended, signals on the transmit path are from the RS to an XGXS of the XAUI interface on one XGMII and from the XGXS to the PHY on the other XGMII. When extended, signals on the receive path are from the PHY to an XGXS of the XAUI interface on one XGMII and from the XGXS to the RS on the other XGMII. The descriptions of the XGMII as between the RS and the PHY are therefore equally applicable between the RS and the XGXS or the XGXS and the PHY.

C.4.3 Application

This clause applies to the interface between the MAC and PHY. The physical implementation of the interface is primarily intended as a chip-to-chip (integrated circuit to integrated circuit) interface implemented with traces on a printed circuit board. The XGMII may also be used in other ways, for example, as a logical interface between ASIC logic modules within an integrated circuit.

This interface is used to provide media independence so that an identical media access controller may be used with all 10GBASE PHY types using either serial or wavelength division multiplexed optics.

C.4.4 Rate of operation

The XGMII supports only the 10 Gb/s data rate as defined within this clause.

C.4.5 Delay Constraints

Predictable operation of the MAC Control PAUSE operation requires that there be an upper bound on the propagation delays through the network. This implies that MAC, MAC Control sub-layer, and PHY implementors must conform to certain delay maxima, and that network planners and administrators conform to constraints regarding the cable topology and concatenation of devices. Table 46–1 contains the values of maximum cumulative MAC Control, MAC and RS round-trip (sum of transmit and receive) delay in bit time as specified in 1.4 and pause quanta as specified in 31B.2.

C.4.6 Allocation of functions

The allocation of functions at the XGMII balances the need for media independence with the need for a simple and cost-effective interface. The bus width and signaling rate are applicable to short distance chip-to-chip interconnect with printed circuit board trace lengths electrically limited to approximately 7 cm. The XGMII maximizes media independence by cleanly separating the Data Link and Physical Layers of the ISO (IEEE) seven-layer reference model.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text was cophas been updated and deleted (continued):

C.4.7 XGMII structure

The XGMII is composed of independent transmit and receive paths. Each direction uses 32 data signals (TXD<31:0> and RXD<31:0>), four control signals (TXC<3:0> and RXC<3:0>) and a clock (TX_CLK and RX_CLK). Figure 46–2 depicts a schematic view of the RS inputs and outputs.

The 32 TXD and four TXC signals shall be organized into four data lanes, as are the 32 RXD and four RXC signals (see Table 46–2). The four lanes in each direction share a common clock — TX_CLK for transmit and RX_CLK for receive. The four lanes are used in round-robin sequence to carry an octet stream. On transmit, each eight PHY_DATA.request transactions represent an octet transmitted by the MAC. The first octet is aligned to lane 0, the second to lane 1, the third to lane 2 the fourth to lane 3, then repeating with the fifth to lane 0, etc. Delimiter and interframe idle characters are encoded on the TXD and RXD signals with the control code indicated by assertion of TXC and RXC respectively.

C.4.8 Mapping of XGMII signals to P-SAP service primitives

The Reconciliation Sublayer (RS) shall map the signals provided at the XGMII to the P-SAP service primitives as shown in Figure 2. The following P-SAP service primitives are defined:

— PHY_DATA.request
— PHY_DATA.indicate
— PHY_DATA_VALID.indicate
— PHY_LINK_STATUS.indicate
— PHY_READY.indicate

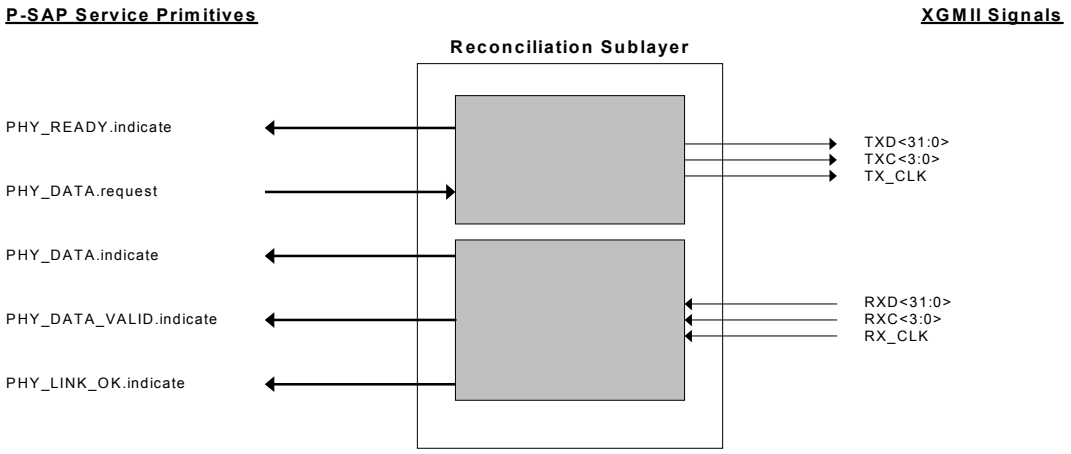


Figure A.2—Reconciliation Sublayer (RS) inputs and outputs

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

C.4.8.1 Mapping of PHY_DATA.request

C.4.8.1.1 Function

Map the primitive PHY_DATA.request to the XGMII signals TXD<31:0>, TXC<3:0> and TX_CLK.

C.4.8.1.2 Semantics of the service primitive

```
PHY_DATA.request (OUTPUT_UNIT)
```

The OUTPUT_UNIT parameter either takes the value of an octet of data or DATA_COMPLETE, and represents the transfer of an octet of data from the MAC to the RS. The value DATA_COMPLETE indicates that the MAC has no more data to transfer.

C.4.8.1.3 When generated

This primitive is generated by the MAC sublayer to request the transmission of a data octet on the physical medium, or to indicate that no more data is available for transmission.

C.4.8.1.4 Effect of receipt

The OUTPUT_UNIT values are conveyed to the PHY by the signals TXD<31:0> and TXC<3:0> on each TX_CLK edge. Each PHY_DATA.request transaction shall be mapped to a TXD signal in sequence (TXD<0:7>,... TXD<24:31>, TXD<0:7>) as described in xxx. After four PHY_DATA.request transactions from the MAC sublayer, the RS requests transmission of 32 data bits by the PHY. The first eight octets of the frame shall be converted as follows:

a) The first octet is converted to a Start control character and aligned to lane 0;
b) The following six octets are converted to Preamble characters;
c) The eighth character is converted to a SFD character.

The MAC does not generate interframe spacing—the RS generates IPG between MAC frames. The DATA_COMPLETE value shall initiate IPG generation as follows:

a) The DATA_COMPLETE value is mapped to a Terminate control character;
b) Interframe spacing is generated in accordance with IEEE Std 802.3 Clause 4, and mapped to IPG.

Figure A.3—Transmit and receive lane associations

C.4.8.2 Mapping of PHY_DATA.indicate

C.4.8.2.1 Function

Map the primitive PHY_DATA.indicate to the XGMII signals RXD<31:0>, RXC<3:0> and RX_CLK.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

C.4.8.2.2 Semantics of the service primitive

`PHY_DATA.indicate (INPUT_UNIT)`

The INPUT_UNIT parameter takes the value of an octet of data, and defines the transfer of an octet of data from the RS to the MAC.

C.4.8.2.3 When generated

The INPUT_UNIT values are derived from the signals RXC<3:0> and RXD<31:0> received from the PHY on each edge of RX_CLK. Each primitive generated to the MAC sublayer entity corresponds to a PHY_DATA.request issued by the MAC at the remote end of the link connecting two DTEs. For each RXD<31:0> during frame reception, the RS shall generate four PHY_DATA.indicate transactions until the end of frame (Terminate control character), where one, two, three, or four PHY_DATA.indicate transactions will be generated from the RXD<31:0> containing the Terminate. During frame reception, each RXD signal shall be mapped in sequence into a PHY_DATA.indicate transaction (RXD<0:7>, ... RXD<24:31>, RXD<0:7>) as described in xxx.

The RS shall convert a valid Start control character and the subsequent eight octets of data to null data prior to generation of the associated PHY_DATA.indicate transactions. The RS shall not generate any PHY_DATA.indicate primitives for a Terminate control character. To assure robust operation, the value of the data transferred to the MAC may be changed by the RS as required by XGMII error indications (see 46.3.3). Sequence ordered_sets are not indicated to the MAC (see 46.3.4).

C.4.8.2.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.4.8.3 Mapping of PHY_DATA_VALID.indicate

C.4.8.3.1 Function

Map the primitive PHY_DATA_VALID.indicate to the XGMII signals RXC<3:0> and RXD<31:0>.

C.4.8.3.2 Semantics of the service primitive

`PHY_DATA_VALID.indicate (DATA_VALID_STATUS)`

The DATA_VALID_STATUS parameter can take one of two values: DATA_VALID or DATA_NOT_VALID. The DATA_VALID value indicates that the INPUT_UNIT parameter of the PHY_DATA.indicate primitive contains a valid data of an incoming frame. The DATA_NOT_VALID value indicates that the INPUT_UNIT parameter of the PHY_DATA.indicate primitive does not contain valid data of an incoming frame.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

C.4.8.3.3 When generated

The PHY_DATA_VALID.indicate service primitive shall be generated by the RS whenever the DATA_VALID_STATUS parameter changes from DATA_VALID to DATA_NOT_VALID or vice versa. DATA_VALID_STATUS shall assume the value DATA_VALID when a PHY_DATA.indicate transaction is generated in response to reception of a Start control character on lane 0 if the prior RXC<3:0> and RXD<31:0> contained four Idle characters or a Sequence ordered set. DATA_VALID_STATUS shall assume the value DATA_NOT_VALID when RXC of the current lane in sequence is asserted for anything except an Error control character. In the absence of errors, DATA_NOT_VALID is caused by a Terminate control character. When DATA_VALID_STATUS changes from DATA_VALID to DATA_NOT_VALID because of a control character other than Terminate, the RS shall ensure that the MAC will detect a FrameCheckError prior to indicating DATA_NOT_VALID to the MAC (see xxx).

C.4.8.3.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.4.8.4 Mapping of PHY_LINK_STATUS.indicate

C.4.8.4.1 Function

Map the primitive PHY_LINK_STATUS.indicate to the XGMII signals RXD<31:0>, RXC<3:0> and RX_CLK.

C.4.8.4.2 Semantics of the service primitive

```
PHY_LINK_STATUS.indicate (LINK_STATUS)
```

The LINK_STATUS parameter takes the value of OK, FAIL, or DEGRADE, and signifies the status of the PHY as indicated by [illegible] signaling. OK indicates that no Local Fault signal has been received. FAIL indicates that a [illegible] has been received. DEGRADE is not generated by the RS.

C.4.8.4.3 When generated

The LINK_STATUS values are derived from the signals RXC<3:0> and RXD<31:0> received from the PHY on each edge of RX_CLK. Details TBD.

C.4.8.4.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

C.4.8.5 Mapping of PHY_READY.indicate

C.4.8.5.1 Function

Map the primitive PHY_READY.indicate to the XGMII signals RXD<31:0>, RXC<3:0> and RX_CLK.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text has been updated and deleted (continued):

C.4.8.5.2 Semantics of the service primitive

`PHY_READY.indicate (READY_STATUS)`

The READY_STATUS parameter takes the value of READY or NOT_READY, and signifies the status of a transmitted frame. If a frame is being received from the MAC and has not been transmitted, READY_STATUS indicates NOT_READY. If a frame has been received and fully transmitted, READY_STATUS indicates READY signifying that a new frame can be received.

C.4.8.5.3 When generated

The LINK_STATUS values is derived from the RS transmit state machine. Details TBD.

C.4.8.5.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.5 XGMII data stream

Data frames transmitted through the XGMII shall be transferred within the XGMII data stream. The data stream is a sequence of bytes, where each byte conveys either a data octet or control character. The parts of the data stream are shown in Figure A.4—.

<inter-frame><preamble><sfd><data><efd>

Figure A.4—XGMII data stream

For the XGMII, transmission and reception of each bit and mapping of data octets to lanes shall be as shown in Figure A.5—.

C.5.1 Inter-frame <inter-frame>

The inter-frame <inter-frame> period on an XGMII transmit or receive path is an interval during which no frame data activity occurs. The interpacket gap is generated by the RS. The <inter-frame> begins with the Terminate control character, continues with Idle control characters and ends with the Idle control character prior to a Start control character. The length of the interpacket gap may be changed between the transmitting RS and receiving RS by one or more functions (e.g., RS lane alignment, PHY clock rate compensation or 10GBASE-W data rate adaptation functions). The minimum IPG at the XGMII of the receiving RS is five octets.

The signaling of link status information logically occurs in the <inter-frame> period (see xxx).

C.5.2 Preamble <preamble> and start of frame delimiter <sfd>

The preamble <preamble> begins a frame transmission by a MAC as specified in 4.2.5 and when generated by a MAC consists of 7 octets with the following bit values:

```
10101010 10101010 10101010 10101010 10101010 10101010 10101010
```

The Start control character indicates the beginning of MAC data on the XGMII. On transmit, the RS converts the first data octet of preamble transferred from the MAC into a Start control character. On receive, the RS will convert the Start control character into a preamble data octet. The start control character is aligned to lane 0 of the XGMII by the RS on transmit and by the PHY on receive.

The start of frame delimiter <sfd> indicates the start of a frame and immediately follows the preamble. The bit value of <sfd> at the XGMII is unchanged from the Start Frame Delimiter (SFD) specified in 4.2.6 and is the bit sequence:

```
10101011
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

Figure A.5—Relationship of data lanes to MAC serial bit stream

The preamble and SFD are shown above with their bits ordered for serial transmission from left to right. As shown, the left-most bit of each octet is the LSB of the octet and the right-most bit of each octet is the MSB of the octet.

The preamble and SFD are transmitted through the XGMII as octets sequentially ordered on the lanes of the XGMII. The first preamble octet is replaced with a Start control character and it is aligned to lane 0, the second octet on lane 1, the third on lane 2 and the fourth on lane 3, and the four octets are transferred on the next edge of TX_CLK. The fifth octet is assigned to lane 0 with subsequent octets sequentially assigned to the lanes with the SFD assigned to lane 3. The XGMII <preamble> and <sfd> are:

Lane 0	**Lane 1**	**Lane 2**	**Lane 3**
Start	10101010	10101010	10101010
10101010	10101010	10101010	10101011

C.5.3 Data <data>

The data <data> in a well-formed frame shall consist of a set of data octets.

C.5.4 End of frame delimiter <efd>

Assertion of TXC with the appropriate Terminate control character encoding of TXD on a lane constitutes an end of frame delimiter <efd> for the transmit data stream. Similarly, assertion of RXC with the appropriate Terminate control character encoding of RXD constitutes an end of frame delimiter for the receive data stream. The XGMII shall recognize the end of frame delimiter on any of the four lanes of the XGMII.

C.5.5 Definition of Start of Packet and End of Packet Delimiter

For the purposes of Clause 30 layer management the start of packet delimiter is defined as the Start control character; and the end of packet delimiter is defined as the end of the last sequential data octet preceding the Terminate control character or other control character causing a change from DATA_VALID to DATA_NOT_VALID. (See 46.1.7.5.2 and 30.3.2.1.5.)

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6 XGMII functional specifications

The XGMII is designed to make the differences among the various media and transceiver combinations transparent to the MAC sublayer. The selection of logical control signals and the functional procedures are all designed to this end.

C.6.1 Transmit

C.6.1.1 TX_CLK (10 Gb/s transmit clock)

TX_CLK is a continuous clock used for operation at 10 Gb/s. TX_CLK provides the timing reference for the transfer of the TXC<3:0> and TXD<31:0> signals from the RS to the PHY. The values of TXC<3:0> and TXD<31:0> shall be sampled by the PHY on both the rising edge and falling edge of TX_CLK. TX_CLK is sourced by the RS.

The TX_CLK frequency shall be 156.25MHz ±0.01%, one-sixty-fourth of the MAC transmit data rate.

C.6.1.2 TXC<3:0> (transmit control)

TXC<3:0> indicate that the RS is presenting either data or control characters on the XGMII for transmission. The TXC signal for a lane shall be de-asserted when a data octet is being sent on the corresponding lane and asserted when a control character is being sent. In the absence of errors, the TXC signals are deserted by the RS for each octet of the preamble (except the first octet that is replaced with a Start control character) and remain de-asserted while all octets to be transmitted are presented on the lanes of the XGMII. TXC<3:0> are driven by the RS and shall transition synchronously with respect to both the rising and falling edges of TX_CLK. Table 46–3 specifies the permissible encoding of TXD and TXC for a XGMII transmit lane. Additional requirements apply for proper code sequences and in which lanes particular codes are valid (e.g., S control character is to be aligned to lane 0).

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

Table A.1—Permissible encoding of TXC, and TXD

C.6.1.3 TXD<31:0> (transmit data)

TXD is a bundle of 32 data signals organized into four lanes of eight signals each (TXD<7:0>, TXD<15:8>, TXD<23:16>, and TXD<31:24>) that are driven by the RS. Each lane is associated with a TXC signal as shown in Table 46–2 and shall be encoded as shown in Table 46–3. TXD<31:0> shall transition synchronously with respect to both the rising and falling edges of TX_CLK. For each high or low TX_CLK transition, data and/or control are presented on TXD<31:0> to the PHY for transmission. TXD<0> is the least significant bit of lane 0, TXD<8> the least significant bit of lane 1, TXD<16> the least significant bit of lane 2 and TXD<24> the least significant bit of lane 3.

Assertion on a lane of appropriate TXD values when TXC is asserted will cause the PHY to generate code-groups associated with either Start, Terminate, Sequence or Error control characters. While the TXC of a lane is de-asserted, TXD values are used to request the PHY to generate code-groups corresponding to the data octet value of TXD. An example of normal frame transmission is illustrated in Figure A.6—.

Figure A.7— shows the behavior of TXD and TXC during an example transmission of a frame propagating an error.

Figure A.6—Normal frame transmission

Figure A.7—Transmit Error Propagation

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6.1.4 Start control character alignment

On transmit, it may be necessary for the RS to modify the length of the <inter-frame> in order to align the Start control character (first octet of preamble) on lane 0. This shall be accomplished in one of two ways.

— A MAC implementation may incorporate this RS function into its design and always insert additional idle characters to align the start of preamble on a four byte boundary. Note that this will reduce the effective data rate for certain packet sizes separated with minimum inter-frame spacing.

— Alternatively, the RS may maintain the effective data rate by sometimes inserting and sometimes deleting idle characters to align the Start control character. When using this method the RS must maintain a Deficit Idle Count (DIC) that represents the cumulative count of idle characters deleted or inserted. The DIC is increments for each idle character deleted, decremented for each idle character inserted, and the decision of whether to insert or delete idle characters is constrained by bounding the DIC to a minimum value of zero and maximum value of three. Note that this may result in inter-frame spacing observed on the transmit XGMII that is up to three octets shorter than the minimum transmitted preamble specified in Clause 4, however the frequency of shortened inter-frame spacing is constrained by the DIC rules. The Deficit Idle Count is only reset at initialization and is applied regardless of the size of the IPG transmitted by the MAC sublayer. An equivalent technique may be employed to control RS alignment of the Start control character provided that the result is the same as if the RS implemented DIC as described.

C.6.2 Receive

C.6.2.1 RX_CLK (receive clock)

RX_CLK is a continuous clock which provides the timing reference for the transfer of the RXC<3:0> and RXD<31:0> signals from the PHY to the RS. RXC<3:0> and RXD<31:0> shall be sampled by the RS on both the rising and falling edge of RX_CLK. RX_CLK is sourced by the PHY.

The frequency of RX_CLK may be derived from the received data or it may be that of a nominal clock (e.g., TX_CLK). When the received data rate at the PHY is within tolerance, the RX_CLK frequency shall be 156.25MHz ±0.01%, one-sixty-fourth of the MAC receive data rate.

There is no need to transition between the recovered clock reference and a nominal clock reference on a frame-by-frame basis. If loss of received signal from the medium causes a PHY to lose the recovered RX_CLK reference, the PHY shall source the RX_CLK from a nominal clock reference. Transitions from nominal clock to recovered clock or from recovered clock to nominal clock shall not decrease the time between adjacent edges of RX_CLK.

NOTE—This standard neither requires nor assumes a guaranteed phase relationship between the RX_CLK and TX_CLK signals.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6.2.2 RXC<3:0> (receive control)

RXC<3:0> indicate that the PHY is presenting either recovered and decoded data or control characters on the XGMII. The RXC signal for a lane shall be de-asserted when a data octet is being received on the corresponding lane and asserted when a control character is being received. In the absence of errors, the RXC signals are de-asserted by the PHY for each octet of the preamble (except the first octet that is replaced with a Start control character) and remain de-asserted while all octets to be received are presented on the lanes of the XGMII. RXC<3:0> are driven by the PHY and shall transition synchronously with respect to both the rising and falling edges of RX_CLK. Table A.2— specifies the permissible encoding of RXD and RXC for a XGMII receive lane. Additional requirements apply for proper code sequences and in which lanes particular codes are valid (e.g., Start control character is to be aligned to lane 0).

Figure 46–7 shows the behavior of RXC<3:0> during an example frame reception with no errors.

Table A.2—Permissible encoding of RXC, and RXD

Figure A.8—Basic Frame Reception

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6.2.3 RXD (receive data)

RXD is a bundle of 32 data signals (RXD<31:0>) organized into four lanes of eight signals each (RXD<7:0>, RXD<15:8>, RXD<23:16>, and RXD<31:24>) that are driven by the PHY. Each lane is associated with a RXC signal as shown in Table 46–2 and shall be encoded as shown in Table 46–4. RXD<31:0> shall transition synchronously with respect to both the rising and falling edges of RX_CLK. For each high or low RX_CLK transition, received data and/or control are presented on RXD<31:0> for mapping by the RS. RXD<0> is the least significant bit of lane 0, RXD<8> the least significant bit of lane 1, RXD<16> the least significant bit of lane 2 and RXD<24> the least significant bit of lane 3. Figure 46–7 shows the behavior of RXD<31:0> during frame reception.

While the RXC of a lane is de-asserted, RXD of the lane is used by the RS to generate PHY_DATA.indicate transactions. Assertion on a lane of appropriate RXD values when RXC is asserted indicates to the RS the Start control character, Terminate control character, Sequence control character or Error control character that drive its mapping functions.

RXC of a lane is asserted with the appropriate Error control character encoding on RXD of the lane to indicate an error was detected somewhere in the frame presently being transferred from the PHY to the RS (e.g., a coding error, or any error that the PHY is capable of detecting, and that may otherwise be undetectable at the MAC sublayer).

The effect of an error control character on the RS is defined in 46.3.3.1.Figure A.9—shows the behavior of RXC and RXD during the reception of an example frame with an error.

Figure A.9—Reception with error

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6.3 Error and fault handling

C.6.3.1 Response to error indications by the XGMII

If, during frame reception (i.e., when DATA_VALID_STATUS = DATA_VALID), a control character other than a Terminate control character is signaled on a received lane, the RS shall ensure that the MAC will detect a FrameCheckError in that frame. This requirement may be met by incorporating a function in the RS that produces a received frame data sequence delivered to the MAC sublayer that is guaranteed to not yield a valid CRC result, as specified by the frame check sequence algorithm (see 3.2.8). This data sequence may be produced by substituting data delivered to the MAC. The RS generates eight PHY_DATA.indicate primitives for each Error control character received within a frame, and may generate eight PHY_DATA.indicate primitives to ensure FrameCheckError when a control character other than Terminate causes the end of the frame.

Other techniques may be employed to respond to a received Error control character provided that the result is that the MAC sublayer behaves as though a FrameCheckError occurred in the received frame.

C.6.3.2 Conditions for generation of transmit Error control characters

If, during the process of transmitting a frame, it is necessary to request that the PHY deliberately corrupt the contents of the frame in such a manner that a receiver will detect the corruption with the highest degree of probability, then an Error control character may be asserted on a transmit lane by the appropriate encoding of the lane's TXD and TXC signals.

C.6.3.3 Response to indication of invalid frame sequences

The 10 Gb/s PCS is required to align the Start control character to lane 0. The RS shall not indicate DATA_VALID to the MAC for a Start control character received on any other lane. Error free 10 Gb/s operation will not change the SFD alignment in lane 3. A 10 Gb/s MAC/RS implementation is not required to process a packet that has an SFD in a position other than lane 3 of the column following the column containing the Start control character.

C.6.4 Link fault signaling

Sublayers within the PHY are capable of detecting faults that render a link unreliable for communication. Upon recognition of a fault condition a PHY sublayer indicates Local Fault status on the data path. When this local fault status reaches an RS, the RS indicates the fault status to the MAC through the PHY_LINK_STATUS.indicate service primitive. When the RS no longer receives fault status messages, it returns to normal operation, it signals normal status to the MAC.

Status is signaled in a four byte Sequence ordered_set as shown in Table A.3—. The PHY indicates Local Fault with a sequence control character in lane 0 and data characters of 0x00 in lanes 1 and 2 plus a data character of 0x01 in lane 3. Remote Fault is not used. Though most fault detection is on the receive data path of a PHY, in some specific sublayers, faults can be detected on the transmit side of the PHY. This is also indicated by the PHY with a Local Fault status.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

Table A.3—Sequence ordered_sets

The RS reports the fault status of the link. Local Fault indicates a fault detected on the receive data path between the remote RS and the local RS. The RS shall implement the link fault signaling state machine (see Figure 46– 9).

C.6.4.1 Conventions

The notation used in the state diagram follows the conventions of 21.5. The notation ++ after a counter indicates it is to be increments.

C.6.4.2 Variables and counters

The link fault signaling state machine uses the following variables and counters:

col_cnt
: A count of the number of columns received not containing a fault_sequence. This counter increments at RX_CLK rate (on both the rising and falling clock transitions) unless reset.

fault_sequence
: A new column received on RXC<3:0> and RXD<31:0> comprising a Sequence ordered_set of four bytes and consisting of a Sequence control character in lane 0 and a seq_type in lanes 1, 2, and 3 indicating either Local Fault or Remote Fault.

last_seq_type
: The seq_type of the previous Sequence ordered_set received
: Values:Local Fault; 0x00 in lane 1, 0x00 in lane 2, 0x01 in lane 3.
: Remote Fault; 0x00 in lane 1, 0x00 in lane 2, 0x02 in lane 3.

link_fault
: An indicator of the fault status.
: Values:OK; No fault.
: Local Fault; fault detected by the PHY.
: Remote Fault; fault detection signaled by the remote RS.

reset
: Condition that is true until such time as the power supply for the device that contains the RS has reached the operating region.
: Values: FALSE: The device is completely powered and has not been reset (default).
: TRUE: The device has not been completely powered or has been reset.

seq_cnt
: A count of the number of received Sequence ordered_sets of the same type.

seq_type
: The value received in the current Sequence ordered_set
: Values: Local Fault; 0x00 in lane 1, 0x00 in lane 2, 0x01 in lane 3.
: Remote Fault; 0x00 in lane 1, 0x00 in lane 2, 0x02 in lane 3.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Editors' Notes: *To be removed prior to final publication.*

The following text is similar or identical to that in IEEE P802.3ae and has been deleted from this annex. It is referenced instead:

C.6.4.3 State Diagram

Figure A.10—Link Fault Signaling State Machine

The link fault signaling state machine specifies the RS monitoring of RXC<3:0> and RXD<31:0> for Sequence ordered_sets. The variable link_fault is set to indicate the value of a received Sequence ordered_set when the following conditions have been met:

a) Four fault_sequences containing the same fault value have been received
b) Without receiving any fault_sequence within a period of 128 columns

The variable link_fault is set to OK following any interval of 128 columns not containing a Remote Fault or Local Fault Sequence ordered_set.

The RS output onto TXC<3:0> and TXD<31:0> is controlled by the variable link_fault.

a) link_fault = OK
b) The RS shall send MAC frames as requested through the PHY service interface. In the absence of MAC frames, the RS shall generate Idle control characters.
c) link_fault = Local Fault
d) The RS shall continuously generate Remote Fault Sequence ordered_sets.
e) ink_fault = Remote Fault
f) The RS shall continuously generate Idle control characters.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex D

(normative)

SONET/SDH reconciliation sublayers

Editors' Notes: *To be removed prior to final publication.*

References:
TBD.

Definitions:
None.

Abbreviations:
None.

Revision History:

Draft 0.1, February 2002	Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

Editors suggest substantial revisions to this annex addressing the following issues:

1. Remove service interface references (expected to be included in Clause7)
2. Revise "SPI" interface requirements based on OIF specifications
3. Define text for sections of the text currently identified as "TBD"
4. Clarify all references that are used in this annex
5. Move all references to GFP or byte-synchronous HDLC-like framing to Clause 7, and condense with references to requirements
6. Change all references to PoS to byte-synchronous HDLC-like framing

"GRS" (GFP) and "SRS" (SONET/SDH) reconciliation sublayers will be identical, except that GRS will pass a "GFP header" to PHY (format is TBD). Comments with specific suggestions and/or concerns are welcome.

Editors' Notes: *To be removed prior to final publication.*

References to the "P-SAP" interface have been changed to the "MAC physical layer service interface" for consistency with other clauses.

Editors' Notes: *To be removed prior to final publication.*

References to the PoS reconciliation sublayer (PRS) have been changed to the SONET/SDH reconciliation sublayer (SRS).

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

D.1 Introduction to SONET/SDH PHY Interface

D.1.1 Overview

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> ***Editors suggest that most of the text in the overview be moved to Clause 7 or deleted, since this annex specifies only a reconciliation sublayer.***

This annex couples the IEEE 802.17 (RPR MAC) to a family of SONET/SDH Physical Layers. The relationships among SONET/SDH, the IEEE 802.17 (RPR MAC) and the ISO/IEC Open System Interconnection (OSI) reference model are shown in Figure B.1

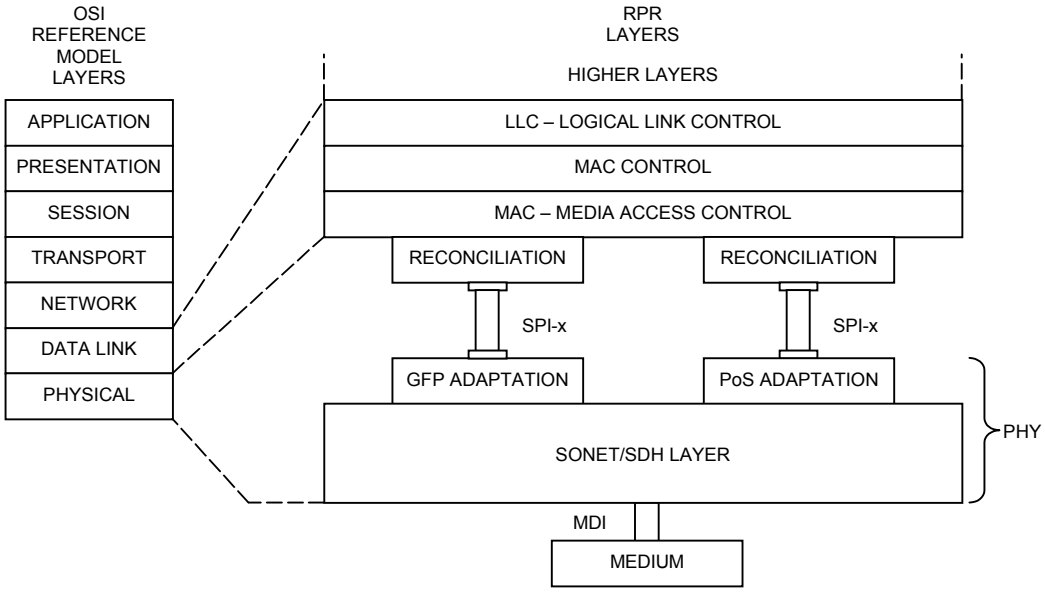


Figure B.1—Architectural positioning of SONET/SDH Physical Interfaces

The SONET/SDH PHY interfaces supports full-duplex operations at all the possible speeds.

The interface between the SONET/SDH layer and the adaptation layer (either the GFP or the SONET/SDH) is the standard interface between the SONET/SDH path sublayer and any upper layer as defined in the ITU-T Recommendations (?) and (?).

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> ***Need exact references for ITU-T recommendations above. Also see succeeding paragraph.***

The GFP and PoS adaptation layers are built over the SONET/SDH Path layer and not directly over a SONET/SDH Medium. Because of the multiplexing capacity of SONET/SDH and of the virtual concatenation feature, there is not a one-to-one relationship between the PHY's bit rate seen by the RPR MAC sublayer and the speed of the physical media.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The functionality performed in the SONET/SDH layer are outside the scope of this specification. The implementations must be compliant with the already existing specifications (?) and (?)**.** The higher-layer paths (VC4), all the possible contiguous concatenated paths (VC4-4Nc) and all the possible virtual concatenated paths (VC4-Nv) are supported by the standard IEEE 802.17 RPR MAC.

All the SONET/SDH speeds, ranging from 150 Mb/s (VC-4) up to 10 Gb/s (VC4-64c), are supported by the IEEE 802.17 Standard. All the intermediate speeds that are a 4x upgrades (VC4-4Nc) are supported. In addition, using virtual concatenation all the intermediate speeds at step of 150 Mb/s (VC4-Nv) can also be supported.

[Editor's note – Check if there is the need to mention other specifications, e.g. for timing purposes.]

The IEEE 802.17 RPR MAC supports two kinds of adaptation layers for SONET/SDH interfaces, that differ each other in the way RPR frames are mapped into the SONET/SDH path payload. These two kinds of adaptation layers do not interwork each other so there are two different SONET/SDH PHYs supported by the IEEE 802.17 RPR MAC.

The first PHY type supports the GFP framing (see section D.1.3), as defined in (?). In this case a family of Generic Reconciliation Sublayer (GRS) is defined to adapt the MAC physical layer service interface to the SONET/SDH signal.

The second PHY type supports the PoS framing (see section D.1.4), as defined in (?). In this case a family of SONET/SDH Reconciliation Sublayer (SRS) is defined to adapt the MAC physical layer service interface to the SONET/SDH signal.

There are two families of Reconciliation Sublayers that differ on the encapsulation mechanism used to map RPR frames into SONET/SDH paths.

The different versions of the Reconciliation Sublayer in the same family (GRSx or SRSx) differ between each other on the version of the SPI-x interface used.

D.1.1.1 System Packet Interface (SPI)

Editors' Notes: *To be removed prior to final publication.*

Editors suggest that references to "SPI" and "system packet interface" be changed, since the interface defined in this annex may be identical or substantially identical to the SPI bus, but is not the SPI bus defined by OIF specifications. Further, the interface defined in this annex is likely to be a subset of the full OIF requirements.

The System Packet Interface (SPI) is a family of interfaces providing an interconnection between the Media Access Control (MAC) sublayer and the Physical Layer entities (PHY).

Two SPI versions are currently defined and supported by IEEE 802.17 Standard.

1) SPI Level 3 supports operations between 155 Mb/s and 622 Mb/s through eight bits wide transmit and receive data paths. It also supports operations between 155 Mb/s and 2,5 Gb/s through 32-bits wide transmit and receive data paths.
2) SPI Level 4 Phase 1 support operations between 622Mb/s and 10Gb/s through 64-bits wide transmit and receive data paths
3) SPI Level 4 Phase 2 supports operations between 622 Mb/s and 10 Gb/s through 16-bits wide transmit and receive data paths.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Because of the SONET/SDH standard, there is not a one-to-one correspondence between the rate at the SPI interface and the line rate on the physical medium. The former is the link rate seen by the RPR MAC sublayer.

D.1.1.2 Supported speeds

The standard IEEE 802.17 RPR MAC supports all the allowed speeds defined in (?) and (?) for SONET/SDH paths and concatenated (either contiguous or virtual) paths starting from 150 Mb/s up to 9,6 Gb/s.

Table 17.4 summarizes all the supported RPR PHY speeds.

Note that because of the standard SONET/SDH multiplexing and virtual concatenation features, there is not a one-to-one correspondence between the RPR PHY speed and the speed of the SONET/SDH physical media over which the signal is transmitted. The possible mappings are outside the scope of this specification and already defined in the existing ITU-T Recommendations (?) and (?).

Sonet Path sublayer	SDH Path sublayer	Bit-rate
STS-3c-SPE	VC4	150 Mb/s
STS-12c-SPE	VC4-4c	600 Mb/s
STS-48c-SPE	VC4-16c	2,4 Gb/s
STS-192c-SPE	VC4-64c	9,6 Gb/s
STS-3c-Nv SPE (Note 1)	VC4-Nv (Note 1)	Nx150 Nb/s (Note 1)

Table B.1—Sonet/SDH Supported Path layers and speeds

Note 1 – All the integer values of X between 2 and 192 are supported. The virtual concatenated paths can support all the speeds from 300 Mb/s up to 9,6 Gb/s with a granularity of 150 Mb/s.

D.1.1.3 Management

Editors' Notes: *To be removed prior to final publication.*

Editors suggest deleting sub-clause D.1.1.3 in it's entirety, since it doesn't appear that the Management objects, attributes, and actions are not appropriate for this sublayer annex.

Managed objects, attributes and actions are defined for all the SONET/SDH components in the relevant standard recommendations (?).

Managed objects, attributes and actions are defined for all the PoS components in the relevant standard specifications (?),(?) and [TBD].

[Editor's note – If we need to define an RPRCP we need a MIB module to manage it. IETF has not yet defined it. If we do not use the LCP/NCP negotiations, we need a MIB to manage this "new" standard PoS interface.]

Managed objects, attributes and actions are defined for all the GFP components in the relevant standard recommendations [TBD].

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

[Editor's note – Neither ITU-T neither IETF has yet addressed the issue of managing the GFP. By the completion of the IEEE 802.17 standard some specification will be available and referred here.]

D.1.2 GFP Framing

The first method to map RPR frames over SONET/SDH path payload is using the GFP mechanisms as defined in (?).

The standard IEEE 802.17 recommends the usage of the GFP with the null extension header and no GFP FCS. All the required functionality for the GFP framing are divided into two sublayers:

1) The Generic Reconciliation sublayer (GRS) that provides a mapping between the signals provided at the SPI-x and the MAC physical layer service interface definition using GFP.
2) The GFP Adaptation Sublayer that provides a mapping between the signals provided at the SPI-x and the service interface between the GFP adaptation sublayer and the SONET/SDH path sublayer.

D.1.2.1 Generic Reconciliation Sublayer (GRS)

The GRS provides a mapping between the signals provided at the SPI-x and the MAC physical layer service interface definition when GFP (refer to (?)) is used.

There are three GRS versions defined by the IEEE 802.17 Standard.

a) The GRS1 (clause D.2) is used to map the signals provided at the 8-bit SPI-3 and the MAC physical layer service interface.
b) The GRS2 (clause D.4) is used to map the signals provided at the 32-bit SPI-3 and the MAC physical layer service interface.
c) The GRS3 (clause D.6) is used to map the signals provided at the SPI-4 Phase 2 and the MAC physical layer service interface.

Editors' Notes: *To be removed prior to final publication.*

Editors suggest adding a SPI-3/32-bit interface for consistency with the four versions specified for the SRS.

The GRS receives an indication of the Signal Fail and/or of the Signal Degrade condition in the SONET/SDH PHY from the Layer Management Entity (LME) and passes this information to the MAC sublayer through the MAC physical layer service interface.

D.1.2.2 GFP Adaptation Sublayer

Editors' Notes: *To be removed prior to final publication.*

Editors suggest moving the GFP framing functions described below to Clause 7 since these functions are performed in a separate adaptation sublayer.

All the functions of this sublayer must be compliant to the standard ITU-T Recommendations (?), (?) and (?)**.** The specification of them is outside the scope of this specification. This section only defines which are the GFP functions that are placed in this sublayer.

The GFP Adaptation Sublayer in the transmit direction performs the scrambling of the GFP payload area (refer to (?)) and inserts the GFP idle frames for rate adaptation purposes. In the receive direction it performs

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

frame delineation, the discarding of the GFP idle frames and the de scrambling of the GFP payload area (refer to (?)).

This sublayer receives the Trail Signal Fail (TSF) and Trail Signal Degrade (TSD) indications from the SONET/SDH Path service interface that represents respectively a failure condition and a degrade condition in the SONET/SDH layer. More details on how this information is generated and passed to the GFP adaptation sublayer are provided in (?).

The TSD indication is passed to the Layer Management Entity (LME) that is responsible to rely it to the GRS.

The TSF indication is correlated with the dPLM and dLFD defects, as defined in (?) and in XXX, to detect a layer 1 failure. This Signal Fail condition is then passed to the Layer Management Entity (LME) that is responsible to rely it to the GRS.

[Editor's note – There is not yet an ITU-T recommendation dealing with the defect correlation in the GFP sublayer. This recommendation should be referred here.]

D.1.3 PoS Framing

Editors' Notes: *To be removed prior to final publication.*

Editors suggest moving the PoS (byte-synchronous HDLC-like) framing functions described below to Clause 7 since these functions are performed in a separate adaptation sublayer.

The second method to map RPR frames over SONET/SDH path payload is using the PoS mechanisms as defined in **, .**. The PoS entity is fully terminated at each station within the 802.17 network.

The standard IEEE 802.17 recommends the usage of a fixed provisioning of the POS link, with no required link negotiation. Only certain features defined in , are defined for use in 802.17 links. These features will be described in [Clause #ref].

All the required functionality for the PoS framing are divided into two sublayers:

a) The SONET/SDH Reconciliation Sublayer (SRS) that provides a mapping between the signals provided at the SPI-x and the MAC physical layer service interface definition using PoS.
b) The PoS Adaptation Sublayer that provides a mapping between the signals provided at the SPI-x and the service interface between the PoS adaptation sublayer and the SONET/SDH path sublayer.

D.1.3.1 SONET/SDH Reconciliation Sublayer (SRS)

The SRS provides a mapping between the signals provided at the SPI-x and the MAC physical layer service interface definition when PoS (refer to (?)) is used.

There are four SRS versions defined by the IEEE 802.17 Standard.

1) The SRS1 (clause D.3) is used to map the signals provided at the 8-bit SPI-3 and the MAC physical layer service interface.
2) The SRS2 (clause D.5) is used to map the signals provided at the 32-bit SPI-3 and the MAC physical layer service interface.
3) The SRS3 (clause D.7) is used to map the signals provided at the SPI-4 Phase 1 and the MAC physical layer service interface.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

4) The SRS3 (clause D.7) is used to map the signals provided at the SPI-4 Phase 2 and the MAC physical layer service interface.

The SRS receives an indication of the Signal Fail and/or of the Signal Degrade condition in the SONET/SDH PHY from the Layer Management Entity (LME) and passes this information to the MAC sublayer through the physical layer service interface.

D.1.3.2 POS Adaptation Sublayer

The POS Adaptation sublayer defines the mapping of 802.17 frames into the SONET/SDH path sublayer. The method functionally the same as that defined in PPP over SONET/SDH and PPP in HDLC-like framing , except that:

— No LCP will be necessary for link establishment in 802.17, rather, a set of recommended LCP configuration options are defined which all compliant POS adaptation sublayers must support by default.
— The mapping does not carry PPP frames, it carries 802.17 frames
— PPP specific and legacy options are not required or defined for use.

The following sub clauses will define in more detail the POS Adaptation Sublayer Requirements.

D.1.3.3 RPR in HDLC-like Framing

FLAG 0x7E	802.17 FRAME (TRANSPARENCY STUFFED)	FLAG 0x7E	ADDITIONAL INTERFRAME FILL OR NEXT FRAME

Figure B.2—RPR in HDLC Frame Structure

A summary of RPR HDLC-like framing is shown in Figure B.2. The Figure does not include any octets inserted for transparency. Note that the frame structure differs from that in in that certain LCP options are define such that fields from the basic frame are not required. These will be discussed in the clauses that follow.

The POS adaption sublayer defines only the use of octet-stuffed framing on 8-bit asynchronous or octet-synchronous links.

D.1.3.3.1 Flag sequence

The Flag Sequence indicates the beginning or end of a frame. The octet stream is examined on an octet-by-octet basis for the value 01111110 (hexadecimal 0x7e).

D.1.3.3.2 Payload transparency

An octet stuffing procedure is used. The Control Escape octet is defined as binary 01111101 (hexadecimal 0x7d), most significant bit first.

As a minimum, sending implementations MUST escape the Flag Sequence and Control Escape octets.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The transmitter examines the entire frame between the two Flag Sequences. Each Flag Sequence, Control Escape octet is replaced by a two octet sequence consisting of the Control Escape octet followed by the original octet exclusive-or'd with hexadecimal 0x20.

This is bit 5 complemented, where the bit positions are numbered 76543210 (the 6th bit as used in ISO numbered 87654321 -- BEWARE when comparing documents).

Receiving implementations MUST correctly process all Control Escape sequences.

On reception, each Control Escape octet is removed, and the following octet is exclusive-or'd with hexadecimal 0x20, unless it is the Flag Sequence (which aborts a frame).

D.1.3.3.3 Invalid Frames

Frames which are too short (less than 2 octets), or which end with a Control Escape octet followed immediately by a closing Flag Sequence, or in which octet-framing is violated, are silently discarded.

D.1.3.4 Time

There is no provision for inter-octet time fill.

The flag sequence MUST be transmitted during interframe time fill.

D.1.3.5 Transmission Considerations

All octets are transmitted Least Significant Bit first.

D.1.3.6 Defined LCP Options

The POS Adaptation sublayer will not required an LCP negotiation to set link parameters. Instead a set of LCP options statically defined for POS links in an 802.17 network station. These options are as follows:

1) Address and Control Field compression is always used
2) Protocol field is not used
3) FCS is never computed nor appended to the Frame
4) The Asynchronous Control Character Map (ACCM) is not used
5) $X^{43}+1$ Payload Scrambling is used

D.1.3.7 Physical Layer Requirements

PPP treats SONET/SDH transport as octet oriented synchronous links.

SONET/SDH links are full-duplex by definition.

D.1.3.7.1 Interface Format

PPP in HDLC-like framing presents an octet interface to the physical layer. There is no provision for sub-octets to be supplied or accepted (?)(?).

The octet stream is mapped into the SONET STS-SPE/SDH Higher Order VC, with the octet boundaries aligned with the SONET STS-SPE/SDH Higher Order VC octet boundaries.

Scrambling is performed during insertion into the SONET STS- SPE/SDH Higher Order VC to provide adequate transparency and protect against potential security threats (see Section 6). For backwards compatibil-

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

ity with RFC 1619 (STS-3c-SPE/VC-4 only), the scrambler MAY have an on/off capability where the scrambler is bypassed entirely when it is in the off mode. If this capability is provided, the default MUST be set to scrambling enabled.

For RPR over SONET/SDH, the entire SONET/SDH payload (SONET STS- SPE/SDH Higher Order VC minus the path overhead and any fixed stuff) is scrambled using a self-synchronous scrambler of polynomial X**43 + 1. [Clause #REF] for the description of the scrambler.

The proper order of operation is:

When transmitting:

CPDU -> RPR -> Byte stuffing -> Scrambling -> SONET/SDH framing

When receiving:

SONET/SDH framing -> Descrambling -> Byte destuffing -> FCS detection -> RPR -> CPDU

Where CPDU is Client Protocol Data Unit.

The Path Signal Label (C2) indicates the contents of the SONET STS-SPE/SDH Higher Order VC. The value CF (CFhex) is used to indicate PPP with X^43 + 1 scrambling.

For compatibility with RFC 1619 (STS-3c-SPE/VC-4 only), if scrambling has been configured to be off, then the value XX (XX hex) is used for the Path Signal Label to indicate PPP without scrambling.

The Multiframe Indicator (H4) is unused, and MUST be zero.

[ED: need to get appropriate code-points defined in for RPR over POS - TBD].

D.1.3.7.2 Control Signals

The POS Adaptation layer does not require the use of control signals as discussed in .

D.1.3.8 Link Performance Monitoring

This sublayer receives the Trail Signal Fail (TSF) and Trail Signal Degrade (TSD) indications from the SONET/SDH path service interface that represents respectively a failure condition and a degrade condition in the SONET/SDH layer. More details on how this information is generated and passed to the POS adaptation sublayer are provided in (?).

The TSD indication is passed to the Layer Management Entity (LME) that is responsible to relay it to the SRS.

The TSF indication is correlated with the dPLM defect, as defined in (?), to detect a layer 1 failure. This Signal Fail condition is then passed to the Layer Management Entity (LME) that is responsible to rely it to the SRS.

D.2 Generic Reconciliation Sublayer version 1 (GRS1) and the 8-bit System Packet

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Interface Level 3 (SPI-3)

D.2.1 Overview

This clause defines the logical and electrical characteristics for the Generic Reconciliation Sublayer version 1 (GRS2) and System Packet Interface Level 3 (SPI-3) using the 8-bit data path, between the RPR media access control and various SONET/SDH PHYs.

Figure B.3 shows the relationship of the Reconciliation sublayer and SPI-3 to the ISO/IEC OSI reference model

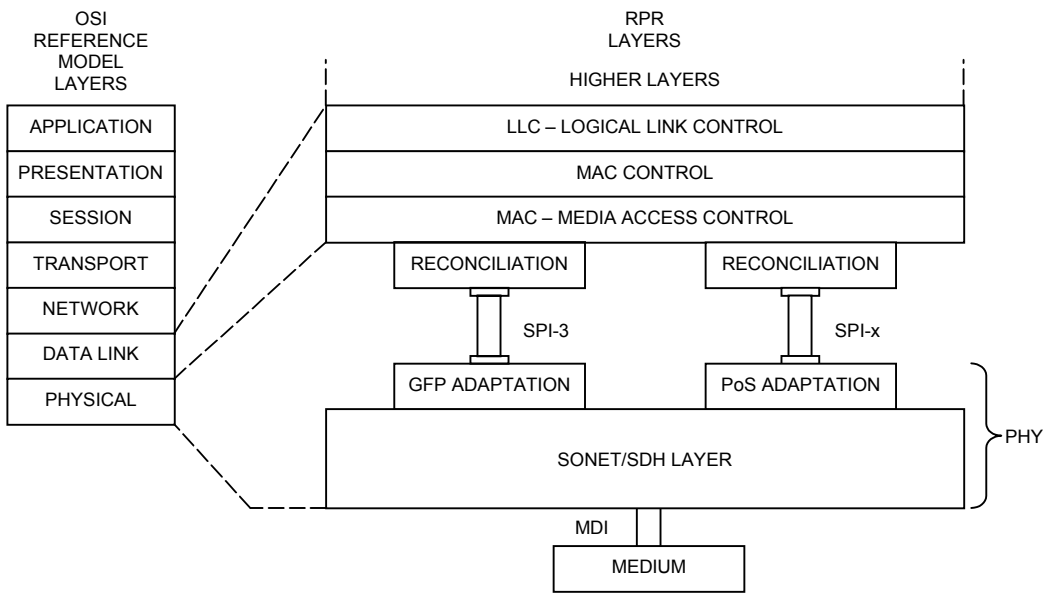


Figure B.3—GRS1 and SPI-3 location in the OSI protocol stack

The purpose of this interface is to provide a simple, inexpensive and easy-to-implement interconnection between Media Access Control (MAC) sublayer and PHYs, and between PHYs and Layer Management Entity (LME).

The interface has the following characteristics:

a) It is capable of supporting operations between 155 Mb/s and 622 Mb/s (with a 155 Mb/s granularity).
b) Data and delimiter are synchronous to clock references.
c) It provides independent 8-bit wide transmit and receive data paths.
d) It provides a simple management interface.
e) It provides for full duplex operations.

D.2.1.1 Summary of major concepts

a) Each direction of data transfer is serviced by Data (a 8-bit bundle), Delimiter, Parity and Clock signals.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

b) The SPI-3 supports logical channels with individual word-level or packet level flow control.
c) The flow control is done out-of-band.
d) The Reconciliation sublayer maps the signal set provided at the SPI-3 to the physical layer service interface primitives provided to the MAC.

D.2.1.2 Application

This clause applies to the interface between the MAC and PHYs and between PHYs and Layer Management Entity (LME). The implementation of the interface is primarily intended as a chip-to-chip (integrated circuits to integrated circuit) interface implemented with traces on a printed circuit board. SA motherboard-to-daughter board interface between two or more printed circuit boards is not precluded.

This interface is used to provide media independence so that an identical media access controller may be used with any of the supported SONET/SDH PHY types.

D.2.1.3 Rate of operation

The SPI-3 with 8-bit data path supports operation at all the speeds between 155 Mb/s and 622 Mb/s at the granularity of 155 Mb/s. It is defined in **Error! Reference source not found.**

SONET/SDH PHY that provide an SPI-3 with 8-bit data path shall support operations, at the SONET/SDH Path level, at the selected rate on the SPI-3. PHYs must report the rates at which they are operating via the management interface.

D.2.1.4 Allocation of functions

The allocation of functions in the SPI interfaces is such that it readily lends itself to implementation in both PHY and MAC sublayer entities.

D.2.1.5 Mapping of SPI-3 signals to physical layer service interface primitives and Station Management

Editors' Notes: *To be removed prior to final publication.*

Editors suggest moving the physical layer service interface description to Clause 7 since this interface is common to all reconciliation sublayers. This annex should describe only requirements specific to the reconciliation sublayer being described.

The Generic Reconciliation Sublayer (GRS) shall map the signals provided at the SPI-3 to physical layer service interface primitives as shown in Figure 17.6. The following service primitives are defined:

— PHY_DATA.request
— PHY_DATA.indicate
— PHY_DATA_VALID.indicate
— PHY_LINK_OK.indicate
— PHY_READY.indicate

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

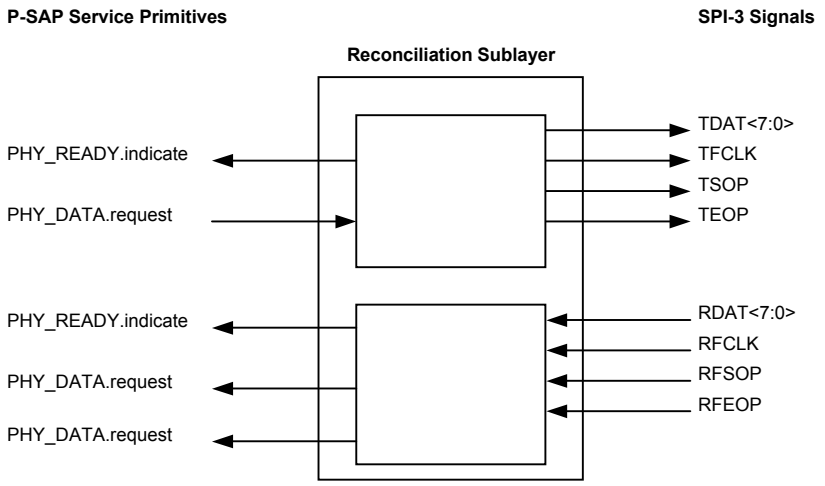


Figure B.4—Generic Reconciliation Sublayer Version 1 inputs and outputs

[Editor's note – The same principles described in section D.4.2.1 applies also here.]

D.2.1.6 SPI-3 functional specifications

The SPI-3 is designed to make the differences among the various SONET/SDH Path sublayers and media combinations transparent to the RPR MAC sublayer.

D.2.1.7 Transmit

The SPI-3 packet interface supports transmit and receive data transfers at clock rates independent of the line bit rate. As a result, PHY layer devices must support packet rate decoupling using FIFOs.

To ease the interface between the Link Layer and PHY layer devices and to support multiple PHY layer interfaces, FIFOs are used. Control signals are provided to both the Link Layer and PHY layer devices to allow either one to exercise flow control. Since the bus interface is point-to-point, the receive interface of the PHY device pushes data to the Link Layer device. For the transmit interface, the packet available status granularity is byte-based.

In the receive direction, when the PHY layer device has stored an end-of packet (a complete small packet or the end of a larger packet) or some predefined number of bytes in its receive FIFO, it sends the in-band address followed by FIFO data to the Link Layer device. The data on the interface bus is marked with the valid signal (RVAL) asserted. A multi-port Link Layer device with multiple FIFOs would service each port in a round-robin fashion when sufficient data is available in its FIFO. The Link Layer device can pause the data flow by deasserting the enable signal (RENB).

In the transmit direction, when the PHY layer device has space for some predefined number of bytes in its transmit FIFO, it informs the Link Layer device by asserting a transmit packet available (TPA). The Link Layer device can then write the 4-byte address followed by packet data to the PHY layer device using the enable signal (TENB). The Link Layer device shall monitor TPA for a high to low transition, which would indicate that the transmit FIFO is full (the number of bytes left in the FIFO can be user selectable, but

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

must be predefined), and suspend data transfer to avoid an overflow. The Layer device can pause the data flow by deasserting the enable signal (TENB).

SPI-3 defines both byte-level and packet-level transfer control in the transmit direction. In byte-level transfer, FIFO status information is presented on a cycle-by-cycle basis. With packet-level transfer, the FIFO status information applies to segments of data. When using byte level transfer, direct status indication must be used. In this case, the PHY layer device provides the transmit packet available status of the selected port (STPA) in the PHY device. As well, the PHY layer device may provide direct access to the transmit packet available status of all ports (DTPA[]) in the PHY device if the number of ports is small. With packet level transfer, the Layer device is able to do status polling on the transmit direction. The Layer device can use the transmit port address TADR[] to poll individual ports of the PHY device, which all respond on a common polled (PTPA) signal.

Since the variable size nature of packets does not allow any guarantee as to the number of bytes available, in both transmit and receive directions, a selected PHY transmit packet available is provided on signal STPA and a receive data valid on signal RVAL. STPA and RVAL always reflect the status of the selected PHY to or from which data is being transferred. RVAL indicates if valid data is available on the receive data bus and is defined such that data transfers can be aligned with packet boundaries.

Physical layer port selection is performed using in-band addressing. In the transmit direction, the Layer device selects a PHY port by sending the address on the TDAT[] bus marked with the TSX signal active and TENB signal inactive. All subsequent TDAT[] bus operations marked with the TSX signal inactive and the TENB active will be packet data for the specified port.

In the receive direction, the PHY device will specify the selected port by sending the address on the RDAT[] bus marked with the RSX signal active and RVAL signal inactive. All subsequent RDAT[] bus operations marked with RSX inactive and RVAL active will be packet data from the specified port.

Both byte-level and packet-level modes are specified in this standard in order to support the current low density multi-port physical layer devices and future higher density multi-port devices. When the number of ports in the physical layer device is limited, byte-level transfer using DTPA[] signals provides a simpler implementation and reduces the need for addressing pins. In this case, direct access will start to become unreasonable as the number of ports increase. Packet-level transfer provides a lower pin count solution using the TADR[] bus when the number of ports is large. In-band addressing ensures the protocol remains consistent between the two approaches. However, the final choice left to the system designers and physical layer device manufacturers to select which approach best suits their desired applications.

Packets shall be written into the transmit FIFO and read from the receive FIFO using a defined data structure. Octets are written in the same order they are to be transmitted or they were received on the SONET line. Within an octet, the MSB (bit 7) is the first bit to be transmitted. The SPI-3 specification does not preclude the transfer of 1-byte packets. In this case, both start of packet and end of packet signals shall be asserted simultaneously.

For packets longer than the PHY device FIFO, the packet must be transferred over the bus interface in sections. The number of bytes of packet data in each section may be fixed or variable depending on the application. In general, the Receive Interface will round-robin between receive FIFOs with fill levels exceeding a programmable high water mark or with at least one end of packet stored in the FIFO. The Receive Interface would end the transfer of data when an end of a packet is transferred or when a programmable number of bytes have been transferred. The Link Layer device may send fixed size sections of packets on the Transmit Interface or use the TPA signal to determine when the FIFO reaches a full level.

The in-band address is specified in a single clock cycle operation marked with the RSX/TSX signals. The port address is specified by the TDAT[7:0]/RDAT[7:0] signals. The address is the numeric value of the TDAT[7:0]/RDAT[7:0] signals where bit 0 is the least significant bit and bit 7 is the most significant bit.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Thus, up to 256 ports may be supported by a single interface. With a 32-bit interface, the upper 24 bits shall be ignored.

D.2.1.8 Receive

The receive FIFO shall have a programmable threshold defined in terms of the number of bytes of packet data stored in the FIFO. A multi-port PHY device must service each receive FIFO with sufficient packet data to exceed the threshold or with an end of packet. The PHY should service the required FIFOs in a round-robin fashion. The type of round-robin algorithm will depend on the various data rates supported by the PHY device and is outside this specification.

The amount of packet data transferred, when servicing the receive FIFO, is bounded by the FIFO's programmable threshold. Thus, a transfer is limited to a maximum of 256 bytes of data (64 cycles for a 32-bit interface or 256 cycles for an 8-bit interface) or until an end of packet is transferred to the Layer device. At the end of a transfer, the PHY device will round-robin to the next receive FIFO.

The PHY device should support a programmable minimum pause of 0 or 2 clock cycles between transfers. A pause of 0 clock cycles maximizes the throughput of the interface. A pause of 2 clock cycles allows the Layer device to pause between transfers.

D.2.1.9 SPI-3 Signal timing characteristic

TFCLK Frequency is specified to be 104 MHz. TFCLK duty cycle is min 40% and max 60%.

Setup time relative to TFCLK for TENB, TDAT, TPRTY, TSOP, TEOP, TMOD, TERR, TSX and TADR is 2 ns minimum. Hold time relative to TFCLK for the same set of signals is 0.5 ns minimum.

Validity of DTPA, STPA, PTPA relative to the rising edge of TFCLK is 1.5ns minimum and 6.0 ns maximum.

RFCLK Frequency is specified to be 104 MHz. RFCLK duty cycle is min 40% and max 60%.

Setup time relative to RFCLK for RENB is 2 ns minimum. Hold time relative to RFCLK for RENB is 0.5 ns minimum.

Validity of RDAT, RPRTY, RSOP, REOP, RMOD, RERR, RVAL, and RSX relative to the rising edge of RFCLK is 1.5 ns minimum and 6.0 ns maximum.

D.2.1.10 SPI-3 electrical characteristics

TBD

D.3 POS Reconciliation Sublayer version 1 (SRS1) and the 8-bit System Packet Interface Level 3 (SPI-3)

[Editor's note – The same principles described in section D.5 applies also here.]

TBD

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

D.4 Generic Reconciliation Sublayer version 2 (GRS2) and the 32-bit System Packet Interface Level 3 (SPI-3)

D.4.1 Overview

This clause defines the logical and electrical characteristics for the Generic Reconciliation Sublayer version 2 (GRS2) and System Packet Interface Level 3 (SPI-3) using the 32-bit data path, between the RPR media access control and various SONET/SDH PHYs.

Figure 17.7 shows the relationship of the Reconciliation sublayer and SPI-3 to the ISO/IEC OSI reference model

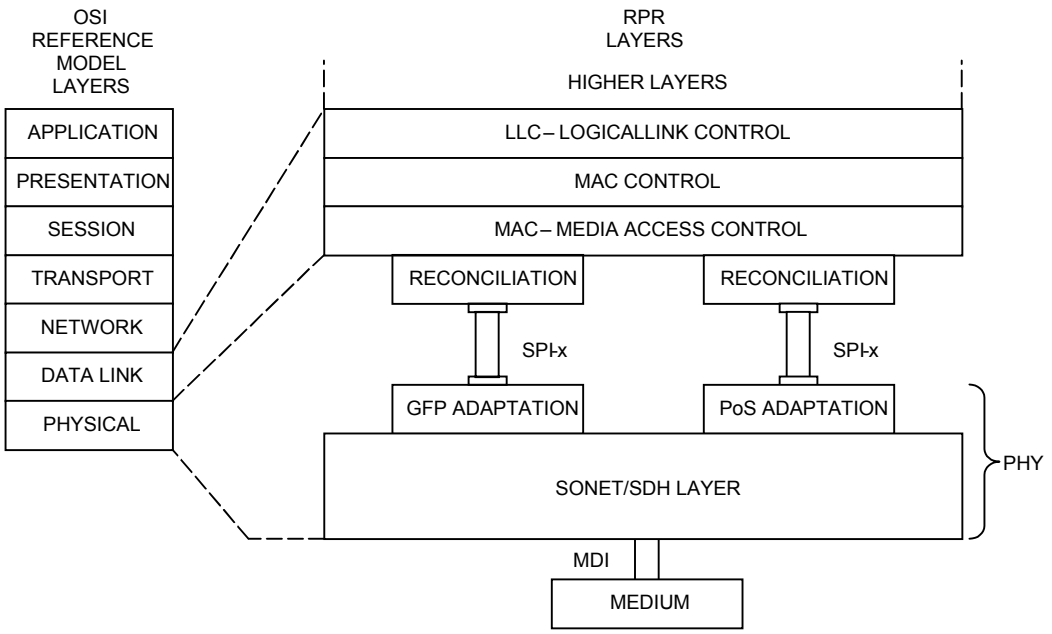


Figure B.5—GRS-3 and SPI-3 location in the OSI protocol stack

The purpose of this interface is to provide a simple, inexpensive and easy-to-implement interconnection between Media Access Control (MAC) sublayer and PHYs, and between PHYs and Layer Management Entity (LME).

The interface has the following characteristics:

a) It is capable of supporting operations between 155 Mb/s and 622 Mb/s (with a 155 Mb/s granularity).
b) Data and delimiter are synchronous to clock references.
c) It provides independent 32-bit wide transmit and receive data paths.
d) It provides a simple management interface.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

e) It provides for full duplex operations.

D.4.1.1 Summary of major concepts

a) Each direction of data transfer is serviced by Data (a 32-bit bundle), Delimiter, Parity and Clock signals.
b) The SPI-3 supports logical channels with individual word-level or packet level flow control.
a) The flow control is done out-of-band.
a) The Reconciliation Sublayer maps the signal set provided at the SPI-3 to the physical layer service interface primitives provided to the MAC.

D.4.1.2 Application

This clause applies to the interface between the MAC and PHYs and between PHYs and Layer Management Entity (LME). The implementation of the interface is primarily intended as a chip-to-chip (integrated circuits to integrated circuit) interface implemented with traces on a printed circuit board. SA motherboard-to-daughterboard interface between two or more printed circuit boards is not precluded.

This interface is used to provide media independence so that an identical media access controller may be used with any of the supported SONET/SDH PHY types.

D.4.1.3 Rate of operation

The SPI-3 with 32-bit data path supports operation at all the speeds between 155 Mb/s and 2.5 Gb/s at the granularity of 155 Mb/s. It is defined in D.8

SONET/SDH PHY that provide an SPI-3 with 32-bit data path shall support operations, at the SONET/SDH Path level, at the selected rate on the SPI-3. PHYs must report the rates at which they are operating via the management interface.

D.4.1.4 Allocation of functions

The allocation of functions in the SPI interfaces is such that it readily lends itself to implementation in both PHY and MAC sublayer entities.

D.4.1.5 Mapping of SPI-3 signals to physical layer service primitives and Station Management

The Generic Reconciliation Sublayer (GRS) shall map the signals provided at the SPI-3 to the physical layer service primitives as shown in Figure 17.8. The following service primitives are defined:

— PHY_DATA.request
— PHY_DATA.indicate
— PHY_DATA_VALID.indicate
— PHY_LINK_OK.indicate
— PHY_READY.indicate

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

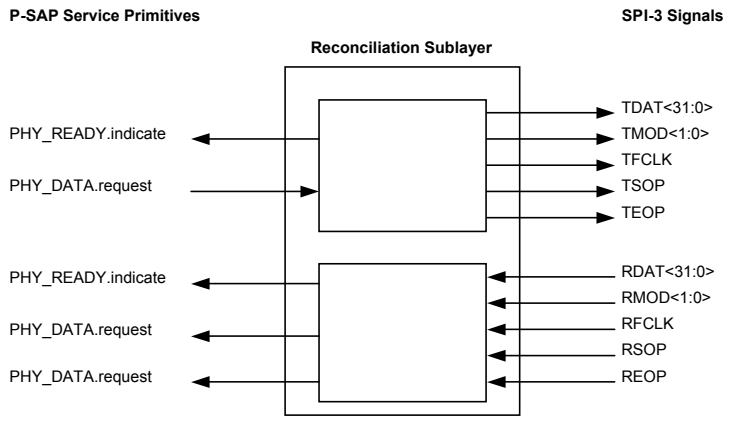


Figure B.6—Generic Reconciliation Sublayer Version 2 inputs and outputs

D.4.1.5.1 Mapping of PHY_DATA.request

D.4.1.5.1.1 Function

Map the primitive PHY_DATA.request to the SPI-3 signals TFCLK, TERR, TENB, TDAT<31:0>,TPRTY, TMOD<1:0>, TSX, TSOP, TEOP.

When transmitting RPR MAC frames the TENB signal is low, so the TSX signal is ignored (see D.8). The usage of the TSX and TENB signals for in-band addressing with multi-port PHY is an implementation choice and out of scope of this specification. If used, this feature should be compliant with the requirements in D.8.

D.4.1.5.1.2 Semantics of the service primitive

```
PHY_DATA.request (OUTPUT_UNIT)
```

The OUTPUT_UNIT parameter either takes the value of an octet of data or the DATA_COMPLETE, and it represents the transfer of an octet of data from the MAC to the RS. The value DATA_COMPLETE indicates that the MAC has no more data to transfer.

D.4.1.5.1.3 When generated

This primitive is generated by the MAC sublayer to request the transmission of a data octet on the physical medium, or to indicate that no more data is available for transmission.

D.4.1.5.1.4 Effect of receipt

The OUTPUT_UNIT values are conveyed to the PHY by the signals TDAT<31:0> and TMOD<1:0> on each TFCLK rising edge.

Each PHY_DATA.request transaction shall be mapped to a TDAT signal in sequence (TDAT<0:7>, …, TDAT<24:31>, TDAT<0:7>), using the big endian coding, as described in D.2.2.6. After the first five PHY_DATA.request and after each four PHY_DATA.request transactions from the MAC sublayer, the RS requests transmission of 32 data bits by the PHY, together with the proper parity information in the TPRTY signal (see D.8), containing the values of the previous four PHY_DATA.request transactions except the last one. The TMOD<1:0> is always fixed to the “00” value except when the end-of-packet is transmitted.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

[Editor's note – The mapping of a MAC data byte on the SPI interface is left to the open discussion. Do we have to invert the bit sequence, due to the big endian order required at the SPI?]

The first eight octets of the frame shall be converted as follows, in order to build a GFP frame:

a) The first two octets are not changed. They correspond to the PLI field of the GFP core header.
b) The following two octets are filled with the CRC-16 calculated, as described in (?), on the first two octets. They correspond to the cHEC field of the GFP core header.
c) The following two octets are filled with the value 0x0009, representing the PTI=î000î (user data frame), the PFI=0 (no payload FCS), the EXI=î0000î (null extension header) and the UPI=î0000 1001î (RPR frame). They correspond to the Type field of the GFP payload header.
d) The last two octets are filled with the CRC-16 calculated, as described in (?) on the previous two octets. They correspond to the tHEC field of the GFP payload header.

[Editor's note – The UPI value "0000 1001" is assigned to RPR by ITU-T as a provisionally agreed item in the living list.]

When transmitting the first four octets, representing the PLI and cHEC fields of the GFP core header, on TDAT<31:0>, the TSOP signal is also high.

The RPR MAC does not generate inter-frame spacing and the GRS does not generate it because there is no need of IPG between RPR MAC frames when using SONET/SDH PHY layers.

The DATA_COMPLETE value shall trigger the indication of the end-of-packet on the SPI-3 as follows:

a) The DATA_COMPLETE value is not mapped to any character on the TDAT signal.
b) The RS requests transmission of 32 data bits by the PHY, together with the proper parity information in the TPRTY signal, containing the values of the previous PHY_DATA.request transactions not yet transmitted. When transmitting this TDAT<31:0> signal, the TEOP is also high and the TMOD<1:0> represents the number of valid data bytes on the TDAT<31:0> as defined in D.8.

D.4.1.5.2 Mapping of PHY_DATA.indicate

D.4.1.5.2.1 Function

Map the primitive PHY_DATA.indicate to the SPI-3 signals RFCLK, RVAL, RENB, RDAT<31:0>, RPRTY, RMOD<1:0>, RSOP, REOP, RERR, RSX.

When receiving RPR MAC frames the RENB signal is low and the RVAL is high, so the RSX signal is ignored (see D.8). The usage of the RSX and RENB signals for in-band addressing with multi-port PHY is an implementation choice and out of scope of this specification. If used, this feature should be compliant with the requirements in D.8.

D.4.1.5.2.2 Semantics of the service primitive

```
PHY_DATA.indicate (INPUT_UNIT)
```

The INPUT_UNIT parameter takes the value of an octet of data and defines the transfer of an octet of data from the RS to the MAC.

D.4.1.5.2.3 When generated

The INPUT_UNIT values are derived from the signals RMOD<1:0> and RDAT<31:0> received from the PHY on each rising edge of the RFCLK. Each primitive generated to the MAC sublayer entity corresponds

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

to a PHY_DATA.request issued by the MAC at the upstream end of the ringlet connecting two RPR adjacent nodes.

For each RDAT<31:0> during frame reception, the RS shall generate four PHY_DATA.indicate transactions until the end of frame (when the REOP is high), where one, tow, three or four PHY_DATA.indicate transactions will be generated from the RDAT<31:0> according to the value coded in the RMOD<1:0> as defined in D.8.

During frame reception each RDAT signal shall be mapped in sequence into a PHY_DATA.indicate transaction (RDAT<0:7>, …, RDAT<24:31>, RDAT<0:7>) as described in D.2.2.6.

Before starting the frame reception, the first eight octets should be checked to be consistent. The following checks are performed:

a) The GFP tHEC field is checked in accordance with (?).
b) The GFP Type field, eventually corrected by the tHEC, is checked to be equal to 0x0009, representing the PTI=î000î (user data frame), the PFI=0 (no payload FCS), the EXI=î0000î (null extension header) and the UPI=î0000 1001î (RPR frame). If the Type field is different from the expected one, the GFP frame must be discarded.

If the previous checks have detected that the GFP frame has to be discarded, no PHY_DATA.indicate primitive is generated during the reception of this GFP frame.

If the previous checks succeed, the RS shall convert the first eight octets of data (delineated by the RSOP signal) as follows, prior to generation of the associated PHY_DATA.indicate transactions.

a) The first two octets are not changed, in order to pass to the RPR MAC layer the length information.
b) The following six octets are converted to null data.

[Editor's note – Which is the null data to be placed in these bytes? We need an input from the frame format group.]

D.4.1.5.2.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

D.4.1.5.3 Mapping of PHY_DATA_VALID.indicate

D.4.1.5.3.1 Function

Map the primitive PHY_DATA_VALID.indicate to the SPI-3 signals RFCLK, RVAL, RENB, RDAT<31:0>, RPRTY, RMOD<1:0>, RSOP, REOP, RERR, RSX.

D.4.1.5.3.2 Semantics of the service primitive

```
PHY_DATA_VALID.indicate (DATA_VALID_STATUS)
```

The DATA_VALID_STATUS parameter can take one of two values: DATA_VALID or DATA_NOT_VALID. The DATA_VALID value indicates that the INPUT_UNIT parameter of the PHY_DATA.indicate primitive contains a valid data of an incoming frame. The DATA_NON_VALID value indicates that the INPUT_UNIT parameter of the PHY_DATA.indicate primitive des not contain valid data of an incoming frame.

D.4.1.5.3.3 When generated

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The RS shall generate the PHYD_DATA_VALID.indicate service primitive whenever the DATA_VALID_STATUS parameter changes from DATA_VALID to DATA_NOT_VALID or vice versa.

The DATA_VALID_STATUS shall assume the value DATA_VALID when a PHY_DATA.indicate transaction is generated in response to reception of a RSOP signal.

The value DATA_NOT_VALID is assumed after the REOP signal to indicate the end of frame to the RPR MAC.

D.4.1.5.3.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

D.4.1.5.4 Mapping of PHY_LINK_OK.indicate

D.4.1.5.4.1 Function

Map the primitive PHY_LINK_OK.indicate to the STA signals MDSF and MDSD.

D.4.1.5.4.2 Semantics of the service primitive

```
PHY_LINK_OK.indicate (LINK_STATUS)
```

The LINK_STATUS parameter can take the value of OK, FAIL or DEGRADE, and signifies the status of the PHY as indicated by the MDSF (Management Data Signal Fail) and MDSD (Management Data Signal Degrade).

The value OK indicates that there are neither failure nor degrade conditions in the physical layer. The value FAIL indicates that a failure has been detected in the physical layer. The value DEGRADE indicates that a signal degrade condition has been detected in the physical layer.

D.4.1.5.4.3 When generated

The RS shall generate the PHY_LINK_OK.indicate primitives whenever the LINK_STATUS parameter changes from one possible value (OK, FAIL, DEGRADE) to a different possible value.

The LINK_STATUS values are derived from the MDSF and MDSD signals received from the Layer Management asynchronously from the data transfer.

The LINK_STATUS parameter shall assume the value OK when neither MDSF nor MDSD signal has been received.

It shall assume the value FAIL when the MDSF signal is asserted.

The value DEGRADE is assumed when the MDSD signal is asserted and the MDSF has not been received.

D.4.1.5.4.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

D.4.1.5.5 Mapping of PHY_READY.indicate

D.4.1.5.5.1 Function

Map the primitive PHY_READY.indicate to the GRS state machine.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

[Editors' note – We have to discuss more in detail this primitive and how it should work.]

D.4.1.5.5.2 Semantics of the service primitive

```
PHY_READY.indicate (READY_STATUS)
```

The READY_STATUS parameter takes the value of READY or NOT_READY, and signifies the status of a transmitted frame. The READY value indicates that a new frame can be received from the RPR MAC sublayer. The NOT_READY value indicates that TBD.

[Editor's note – It is not completely clear to me how the PHY_READY.indicate primitive is used.]

D.4.1.5.5.3 When generated

The RS shall generate the PHY_READY.indicate primitive whenever the READY_STATUS parameter changes from one the READY value to the NOT_READY value and vice versa.

If a frame is being received from the MAC and has not been transmitted, READY_STATUS indicates NOT_READY. If a frame has been received and fully transmitted, READY_STATUS indicates READY, signifying that a new frame can be received.

[Editor's note – It is not completely clear to me how the PHY_READY.indicate primitive is used.]

D.4.1.5.5.4 Effect of receipt

The effect of receipt of this primitive by the MAC sublayer is unspecified.

D.4.2 SPI-3 functional specifications

Refer to section D.2.1.6.

D.5 POS Reconciliation Sublayer version 2 (SRS2) and the 32-bit System Packet Interface Level 3 (SPI-3)

[Editor's note – The same principles of section D.4 applies also here, with some updates due to the usage of POS instead of GFP.]

TBD

D.6 Generic Reconciliation Sublayer version 3 (GRS3) and System Packet Interface Level 4 Phase 2 (SPI-4)

[Editor's note – The same principles of section D.4 applies also here.]

TBD

D.7 POS Reconciliation Sublayer version 3 (SRS3) and System Packet Interface Level 4 Phase 2 (SPI-4)

[Editor's note – The same principles of section D.5 applies also here.]

TBD

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

D.8 SPI-3 Signaling Function Specifications

All signals are expected to be updated and sampled using the rising edge of the transmit FIFO clock TFCLK.

D.8.1 TFCLK (transmit clock)

TFCLK is a continuous clock used to synchronize data transfer transactions between the LINK Layer device and the PHY layer device. TFCLK may cycle at a rate up to 104 MHz. TFCLK is sourced by the MAC to the PHY.

D.8.2 ERR (transmit error Indicator)

TERR is used to indicate that there is an error in the current packet. TERR should only be asserted when TEOP is asserted; it is considered valid only when TENB is asserted.

D.8.3 TENB (transmit write enable)

TENB used to control the flow of data to the transmit FIFOs. When TENB is high the TDAT, TMOD, TSOP, TEOP and TERR signals are invalid and should be ignored by the PHY. The TSX signal is valid and is processed by the PHY when TENB is high. When TENB is low the TDAT, TMOD, TSOP, TEOP and TERR signals are valid and are processed by the PHY. The TSX signal is ignored by the PHY when TENB is low.

D.8.4 TDAT[31:0] (transmit packet data bus)

TDAT is an 8-bit or 32-bit bus which carries the packet octets that are written to the selected transmit FIFO and the in-band port address to select the desired transmit FIFO. The TDAT bus is considered valid only when TENB is asserted. Data is transmitted in big endian order on TDAT.

D.8.5 TMOD[1:0] (transmit word modulo)

TMOD is required only when TDAT is 32-bits. TMOD indicates the number of valid bytes of data in TDAT. The TMOD bus should always be zero except during the last word transfer of a packet on TDAT. When TEOP is asserted the number of valid packet data bytes on TDAT is decoded from TMOD as:

TMOD[1:0] = ì00îTDAT[31:0] is valid
TMOD[1:0] = ì01îTDAT[31:8] is valid
TMOD[1:0] = ì10îTDAT[31:16] is valid
TMOD[1:0] = ì11îTDAT[31:24] is valid

TMOD is considered valid only when TENB is asserted.

D.8.6 TPRTY (transmit bus parity)

TPRTY indicates the parity calculated over the TDAT bus. TPRTY is considered valid only when TENB or TSX is asserted. The parity calculation is such that odd parity is indicated on this signal.

D.8.7 TSX (transmit start of transfer)

TSX indicates when the in-band port address is present on the TDAT bus. When TSX is high and TENB is high, the value of TDAT is the address of the transmit FIFO to be selected. Subsequent data transfers on the

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

TDAT bus will fill the FIFO specified by this in-band address. For single channel PHY devices the TSX signal is optional. TSX is considered valid only when TENB is not asserted.

D.8.8 TSOP (transmit start of packet)

TSOP is used to delineate the packet boundaries on the TDAT bus. When TSOP is high the start of the packet is present on the TDAT bus. TSOP is required to be present at the beginning of every packet and is considered valid only when TENB is asserted.

D.8.9 TEOP (transmit end of packet)

TEOP is used to delineate the packet boundaries on the TDAT bus. When TEOP is high, the end of the packet is present on the TDAT bus. TEOP is required to be present at the end of every packets and is considered valid only when TENB is asserted.

D.8.10 TADR (transmit PHY Address)

TADR bus is used with the PTPA signal to poll the transmit FIFO's packet available status. When TADR is sampled on the rising edge of TFCLK by the PHY the polled packet available indication PTPA is updated with the status of the channel specified by the TADR address on the following rising edge of TFCLK.

D.8.11 DTPA (direct transmit packet available)

DTPA bus provides direct status indication for the corresponding ports in the PHY device. DTPA transitions high when a predefined minimum number of bytes is available in its transmit FIFO. Once high, the DTPA signal indicates that its corresponding transmit FIFO is not full. When DTPA transitions low it indicates that its transmit FIFO is full or near full. DTPA is updated on the rising edge of TFCLK.

D.8.12 RFCLK (receive FIFO write clock)

RFCLK is a continuous clock used to synchronize data transfer transactions between the MAC and the PHY. RFCLK may cycle at a rate up to 104 MHz.

D.8.13 RVAL (receive data valid)

RVAL indicates the validity of the receive data. RVAL is low between transfers and when RSX is asserted. It is also low when the PHY pauses a transfer due to an empty receive FIFO. When a transfer is paused by holding RENB high RVAL will hold its value unchanged although no new data will be present on RDAT until the transfer resumes. When RVAL is high the RDAT, RMOD, RSOP, REOP and RERR signals are valid. When RVAL is low, the RDAT, RMOD, RSOP, REOP and RERR signals are invalid and must be disregarded. The RSX signal is valid when RVAL is low.

D.8.14 RENB (receive read enable)

The RENB signal is used to control the flow of data from the receive FIFOs. During data transfer, RVAL must be monitored as it will indicate if the RDAT, RPRTY, RMOD, RSOP, REOP, RERR and RSX are valid. The system may deassert RENB at anytime if it is unable to accept data from the PHY device. When RENB is sampled low by the PHY device a read is performed from the receive FIFO and the RDAT, RPRTY, RMOD, RSOP, REOP, RERR, RSX and RVAL signals are updated on the following rising edge of RFCLK.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

When RENB is sampled high by the PHY device a read is not performed and the RDAT, RPRTY, RMOD, RSOP, REOP, RERR, RSX and RVAL remain unchanged on the following rising edge of RFCLK.

D.8.15 RDAT[31:0] (receive packet data bus)

RDAT is an 8-bit or 32-bit bus which carries the packet octets that are read from the receive FIFO and the in-band port address of the selected receive FIFO. RDAT is considered valid only when RVAL is asserted. Data is received in big endian order.

D.8.16 RPRTY (receive parity)

RPRTY signal indicates the odd parity calculated over the RDAT bus.

D.8.17 RMOD[1:0] (receive word modulo)

RMOD is required only when RDAT is a 32-bit bus. RMOD indicates the number of valid bytes of data in RDAT. The RMOD bus should always be zero except during the last word transfer of a packet on TDAT. When REOP is asserted the number of valid packet data bytes on RDAT is decoded from RMOD as:

RMOD[1:0] = ì00îRDAT[31:0] is valid

RMOD[1:0] = ì01îRDAT[31:8] is valid

RMOD[1:0] = ì10îRDAT[31:16] is valid

RMOD[1:0] = ì11îRDAT[31:24] is valid

RMOD is considered valid only when RVAL is asserted

D.8.18 REOP (receive end of packet)

REOP is used to delineate the packet boundaries on the RDAT bus. When REOP is high, the end of the packet is present on the RDAT bus. REOP is required to be present at the end of every packets and is considered valid only when RVAL is asserted.

D.8.19 RERR (receive error indicator)

RERR is used to indicate that the current packet is in error. RERR shall only be asserted when REOP is asserted. Conditions that can cause RERR to be set may be, but are not limit to, FIFO overflow, abort sequence detection and FCS error.

D.8.20 RSX (receive start of transfer)

RSX indicates when the in-band port address is present on the RDAT bus. When RSX is high, the value of RADAT[7:0] is the address of the receive FIFO to be selected by the PHY. Subsequent data transfers on the RDAT bus will be from the FIFO specified by this in-band address. For single channel PHY devices the RSX signal is optional. For multi-port PHY devices, RSX must be asserted at the beginning of each transfer. When RSX is high RVAL must be low.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

D.9 SPI-3 management functions

Management functions for this interface involves setting of FIFO thresholds used as triggers for flow control indicators.

D.10 SPI-4 signaling function specifications

SPI-4 is an interface for packet and cell transfer between a physical layer (PHY) device and a link layer device, for aggregate bandwidths of OC-192 ATM and Packet over SONET/SDH (POS), as well as 10 Gb/s Ethernet applications.

On both the transmit and receive interfaces, FIFO status information is sent separately from the corresponding data path. By taking FIFO status information out-of-band, it is possible to decouple the transmit and receive interfaces so that each operates independently of the other. Such an arrangement makes POS-PHY L4 suitable not only for bidirectional but also for unidirectional link layer devices.

In both the transmit and receive interfaces, the packet's address, delineation information and error control coding is sent in-band with the data.

SPI-4 has the following general characteristics:

Editors' Notes: *To be removed prior to final publication.*

The modifications suggested at the beginning of this annex will include removing the multiple-level lettered and numbered list below which does not meet IEEE formatting guidelines.

a) Point-to-point connection (i.e., between single PHY and single Link Layer device).
   1) Support for 256 ports (suitable for STS-1 granularity in SONET/SDH applications (192 ports) and Fast
   2) Ethernet granularity in Ethernet applications (100 ports)).
   3) Transmit / Receive Data Path:
      i) 16 bits wide.
      ii) In-band port address, start/end-of-packet indication, error-control code.
      iii) LVDS I/O (IEEE 1596.3 – 1996 [1], ANSI/TIA/EIA-644-1995 [2]).
      iv) 622 Mb/s minimum data rate per line.
      v) Source-synchronous double-edge clocking, 311 MHz minimum.

b) Transmit / Receive FIFO Status Interface:
   1) - LVTTL I/O or optional LVDS I/O (IEEE 1596.3 – 1996 [1], ANSI/TIA/EIA-644-1995 [2]).
   2) - Maximum 1/4 data path clock rate for LVTTL I/O, data path clock rate (double-edge
   3) clocking) for LVDS I/O.
   4) - 2-bit parallel FIFO status indication.
   5) - In-band Start-of-FIFO Status signal.
   6) - Source-synchronous clocking.

Data and control lines are driven off the rising and falling edges of the clock [TDCLK].

D.10.1 TDCLK (Transmit Data Clock)

TDCLK is a clock associated with TDAT and TCTL. TDCLK provides the datapath source-synchronous double-edge clocking with a minimum frequency of 311 MHz. Data and control lines are driven off the rising and falling edges of the clock. TDCLK is sourced by the MAC to the PHY.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

D.10.2 TDAT[15:0] (Transmit Data)

TDAT is a 16-bit bus used to carry payload data and in-band control words from the Link Layer to the PHY device. A control word is present on TDAT when TCTL is high. The minimum data rate for TDAT is 622 Mb/s.

D.10.3 TCTL (Transmit Control)

TCTL is high when a control word is present on TDAT, otherwise it is low. TCTL is sourced by the MAC to the PHY.

D.10.4 TSCLK (Transmit Status Clock)

TSCLK is a clock associated with TSTAT providing source-synchronous clocking. For LVTTL I/O a maximum clockrate restraint is ¼ that of the data path clock rate. LVDS I/O allows a maximum of that equal to the data path clock (double-edge clocking).

D.10.5 TSTAT[1:0] (Transmit FIFO Status)

TSTAT is a 2-bit bus used to carry round-robin FIFO status information, along with associated error detection and framing. The maximum data rate for TSTAT is dependent on the I/O type, either LVDS or LVTTL, and is limited to its respective TSCLK restraints. TSTAT is sourced by the PHY to the MAC. The FIFO status formats are:

TSSTAT[1:0] = ì11î	Reserved for framing or to indicate a disabled status link.
TSSTAT[1:0] = ì10î	SATISFIED
TSSTAT[1:0] = ì01î	HUNGRY
TSSTAT[1:0] = ì00î	STARVING

D.10.6 RDCLK (Receive Data Clock)

RDCLK is a clock associated with RDAT and RCTL. RDCLK provides the datapath source-synchronous double-edge clocking with a minimum frequency of 311 MHz. Data and control lines are driven off the rising and falling edges of the clock. RDCLK is sourced by the PHY to the MAC.

D.10.7 RDAT[15:0] (Receive Data)

RDAT is a 16-bit bus which carries payload data and in-band control from the PHY to the Link Layer device. A control word is present on RDAT when RCTL is high. The minimum data rate for RDAT is 622 Mb/s.

D.10.8 RCTL (Receive Control)

RCTL is high when a control word is present on RDAT, otherwise it is low. RCTL is sourced by the PHY to the MAC.

D.10.9 RSCLK (Receive Status Clock)

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

RSCLK is a clock associated with RSTAT providing source-synchronous clocking. RSCLK is sourced by the Mac to the PHY. LVDS I/O allows a maximum of that equal to the data path clock (double-edge clocking).

D.10.10 RSTAT[1:0] (Receive FIFO Status)

RSTAT is a 2-bit bus used to carry round-robin FIFO status information, along with associated error detection and framing. The maximum data rate for RSTAT is dependent on the I/O type, either LVDS or LVTTL, and is limited to its respective RSCLK restraints. RSTAT is sourced by the Mac to the PHY. The FIFO status formats are:

TSSTAT[1:0] = ì11î Reserved for framing or to indicate a disabled status link.

TSSTAT[1:0] = ì10î SATISFIED

TSSTAT[1:0] = ì01î HUNGRY

TSSTAT[1:0] = ì00î STARVING

D.11 SPI-4 management functions

Editors' Notes: *To be removed prior to final publication.*

Editors suggest D.13 be deleted in its entirety, since these management functions don't appear to be applicable to an annex describing a reconciliation sublayer.

Parameter	Definition	P	CH	Units
CALENDAR[i]	Port address at calendar location i.	Yes	I	(N/A)
CALENDAR_LEN	Length of the calendar sequence.	Yes	I	(N/A)
CALENDAR_M	Number of times calendar sequence is repeated between insertions of framing pattern.	Yes	I	(N/A)
MAX_CALENDAR_LEN	Maximum supported value of MA_CALENDAR_LEN	No	I	(N/A)
MaxBurst1	Maximum number of 16 byte blocks that the FIFO can accept when FIFO Status channel indicates Starving.	Yes	C/I	16 byte blocks
MaxBurst2	Maximum number of 16 byte blocks that the FIFO can accept when FIFO Status channel indicates Hungry. MaxBurst2 <= MaxBurst1	Yes	C/I	16 byte blocks

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

a	Number of repetitions of the data training sequence that must be scheduled every DATA_MAX_T cycles.	Yes	I	(N/A)
DATA_MAX_T	Maximum interval between scheduling of training sequences on Data Path interface.	Yes	I	Cycles
FIFO_MAX_T	Maximum interval between scheduling of training sequences on FIFO Status Path interface.	Yes	I	Cycles

P = Provisionable, CH = Per channel (C) pr per interface (I)

Upon reset, the FIFOs in the datapath receiver are emptied, and any outstanding credits are cleared in the data path transmitter. After reset, but before active traffic is generated, the data transmitter shall send continuous training patterns. Transmission of training patterns shall continue until valid information is received on the FIFO Status Channel. The receiver shall ignore all incoming data until it has observed the training pattern and acquired synchronization with the data. Synchronization may be declared after a provisionable number of consecutive correct DIP-4 codewords are seen. Loss of synchronization may be reported after a provisionable number of consecutive incorrect DIP- 4 codewords is detected.

After reset but before active traffic is generated, the FIFO Status Channel transmitter shall send a continuous "1 1" framing pattern for LVTTL implementations, or continuous training patterns for optional LVDS implementations. Once the corresponding data channel has achieved synchronization, and a calendar has been provisioned, it may begin transmission of FIFO Status information. Once the data transmitter has received valid FIFO Status information (as indicated, for example, by a sufficient number of consecutively correct DIP- 2 codewords), it may begin transmission of data bursts to channels that have been provisioned and have space available.

In the event that the data path receiver is reset but the transmitter is still active, events at the receiver follow the same behavior as above. It shall ignore all incoming data until it has observed the training pattern and acquired synchronization with the data. It shall also send a continuous "1 1" framing pattern for LVTTL implementations (or continuous training patterns for optional LVDS implementations) on its FIFO Status Channel, cancelling previously granted credits and setting them to zero. In this case the transmitter should send continuous training patterns to facilitate reacquistion by the receiver.

In the event that the data path transmitter is reset but the receiver is still active, events at the transmitter follow the same behavior as above. The transmitter shall send continuous training patterns until a calendar is configured and valid status information is received on the FIFO Status Channel. At the same time, the receiver may have lost synchronization with the data, and begun sending continuous framing patterns (or continuous training patterns for optional LVDS implementations) on the FIFO Status Channel. Once the data transmitter has received valid FIFO Status information (as indicated, for example, by a sufficient number of consecutively correct DIP- 2 codewords), it may begin transmission of data bursts to channels that have been provisioned and have space available.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex E

(normative)

SNMP MIB definitions for 802.17

Editors' Notes: *To be removed prior to final publication.*

References:
IEEE Standard 802.3, IEEE P802.3ae.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002 Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

This clause really should not have all these references to RFC2233. We are defining the RPR MIB here and are not bound by RFC2233. If points of clarification relating to RFC2233 are to be brought out, this should be done via Notes or other non-normative ways.

E.1 Introduction

Editors' Notes: *To be removed prior to final publication.*

We need contributions to fill in text here. Failing the availability of contributions, the editor will undertake the task for D1.0.

E.2 The SNMP management framework

Editors' Notes: *To be removed prior to final publication.*

We need contributions to fill in text here. Failing the availability of contributions, the editor will undertake the task for D1.0.

E.3 Security considerations

Editors' Notes: *To be removed prior to final publication.*

We need contributions to fill in text here. Failing the availability of contributions, the editor will undertake the task for D1.0.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

E.4 Structure of the MIB

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *We need contributions to fill in text here. Failing the availability of contributions, the editor will undertake the task for D1.0.*

E.5 Relationship to other MIBs

It is assumed that a system implementing this MIB will also implement (at least) the "interfaces" group defined in RFC 2233.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *The above is more of a Note than a normative statement.*

E.5.1 Relationship to the Interfaces MIB

RFC 2233, the Interface MIB Evolution, requires that any MIB that is an adjunct of the Interface MIB, clarify specific areas within the Interface MIB. These areas were intentionally left vague in RFC 2233 to avoid over constraining the MIB, thereby precluding management of certain media types.

Implicit in this MIB is the notion of RPR MAC interface. Each of these RPR MAC interface is associated with one interface of the "interfaces" group (one row in the ifTable).

Each RPR MAC interface is uniquely identified by an interface number (ifIndex). The ifIndex value assigned to a given RPR MAC interface is the same for both RPR sides, East and West.

E.5.2 Layering model

This annex assumes the interpretation of the Interfaces Group to be in accordance with RFC 2233, which states that the ifTable contains information on the managed resource's interfaces and that each sublayer below the internetwork layer of a network interface is considered an interface.

E.5.3 IfStackTable

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *We need contributions to fill in text here. Failing the availability of contributions, the editor will undertake the task for D1.0.*

E.6 Definitions for the RPR MIB

RPR-MIB DEFINITIONS ::= BEGIN

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
IMPORTS

   MODULE-IDENTITY, OBJECT-TYPE, experimental, Integer32, Counter64, Counter32, Unsigned32
      FROM SNMPv2-SMI
   MacAddress, TimeStamp,TEXTUAL-CONVENTION, TruthValue
      FROM SNMPv2-TC
   ifIndex
      FROM IF-MIB
   SnmpAdminString
      FROM SNMP-FRAMEWORK-MIB
;

   rprMib  MODULE-IDENTITY
   LAST-UPDATED "200202041200Z"
   ORGANIZATION "IEEE 802.17 Working Group"
   CONTACT-INFO
         "stds-802-17@ieee.org"
   DESCRIPTION
      "The resilient packet ring MIBs for IEEE 802.17"
   ::= { experimental 899 }

--
--         Textual Conventions used in this MIB
--

RprInterfaceSide ::= TEXTUAL-CONVENTION
   STATUS  current
   DESCRIPTION
      "Indicates the side of the RPR interface. Each RPR side is connected to
       both clockwise and counter-clockwise rings.
       EAST side is identified as the side of the node that receives from the clockwise ring.
       WEST side is identified as the side of the node that receives from the counter-clockwise ring."
   SYNTAX INTEGER {
      east          (1),
      west          (2)
      }


RprProtectionCommand ::= TEXTUAL-CONVENTION
   STATUS  current
   DESCRIPTION
      "An RPR ring provides redundancy and protection from a failed node or
       a link/fiber cut through the use of protection modes that are
       automatic or user initiated. Automatic protection modes take effect
       when the ring detects an event, a fiber cut, or a node failure and
       remain in effect until the wait-to-restore value expires. As the
       protection requests travel around the ring, the protection hierarchy
       is applied. All nodes are signalled on the protection messaging
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        channel. If a node or fiber facility failure is detected, traffic
        going toward or coming from the failure direction is wrapped (looped)
        back to go in opposite direction on the other ring or dropped by the
        protecting interface (depend on the protection method). The wrap/drop
        takes place on the nodes adjacent to the failure.

        The protection commands (arranged in ascending priority order) are:

        noCommand
        This value should be returned by a read request when no protection
        command has been written to the object in question since
        initialization. This value may not be used in a write operation.

        manualProtection
        A command which causes a low-priority protection on each end of
        a specified span.

        forcedProtection
        A command which causes a high-priority protection on each end of
        a specified span.

        clear
        Clears protection commands for the specified interface side."
    SYNTAX INTEGER {
      noCommand          (1),
      manualProtection   (2),
      forcedProtection   (3),
      clear              (4)
            }



RprProtectionStatus ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
      "Indicates the current protection status of the interface side.
        The status values are (arranged in ascending priority order) :

        waitToRestore
        The condition for an automatic protection was cleared and the interface
        side is engaged in a wait to restore period.

        manualProtection
        A user initiated manual switch (via the rprMacProtectionCommand
        or TL1 interface) on the interface. Both sides of the
        specified span performs protection.

        signalDegraded
        An automatically-detected status which causes protection on a
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        span when a media signal degrade is detected due to excessive BER.

        signalFailed
        An automatically-detected status which causes protection on a
        span when a media signal failure is detected.

        forcedProtection
        A user initiated forced switch (via the rprMacProtectionCommand
        or TL1 interface) on the interface. Both sides of the
        specified span performs protection."
    SYNTAX BITS {
        waitToRestore        (0),
        manualProtection     (1),
        signalDegraded       (2),
        signalFailed         (3),
        forcedProtection     (4)
        }



--
-- Object groups
--

rprObjects       OBJECT IDENTIFIER ::= { rprMib 1 }

rprGeneral       OBJECT IDENTIFIER ::= { rprObjects 1 }
rprRingCounters   OBJECT IDENTIFIER ::= { rprObjects 2 }
rprLocalCounters  OBJECT IDENTIFIER ::= { rprObjects 3 }
rprErrorCounters  OBJECT IDENTIFIER ::= { rprObjects 4 }



--
-- RPR interface table
--

rprIfTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprIfEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The RPR interface table, extension to the ifTable."
    ::= { rprGeneral 1 }

rprIfEntry OBJECT-TYPE
    SYNTAX RprIfEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        "One such entry for every interface in the ifTable which has an ifType of RPR interface."
    INDEX { ifIndex }
    ::= { rprIfTable 1 }

RprIfEntry ::= SEQUENCE {
    rprPriorityThreshold            Integer32,
    rprIfNodesOnRing                Integer32,
    rprIfTopologyTimer              Integer32,
    rprIfProtectionWTR              Integer32,
    rprIfTimeElapsed                Integer32,
    rprIfValidIntervals             Integer32
            }

rprPriorityThreshold OBJECT-TYPE

    SYNTAX  Integer32 (0..7)

    MAX-ACCESS  read-write
    STATUS  current

    DESCRIPTION
        "In a 2 transit queues RPR MAC, the priority threshold value is
         compared against the 3 priority bits in each packet's RPR header.
         Packets with priority greater or equal to this threshold should be mapped
         to high priory transit queue. Packet with priority less than this threshold
         should be mapped to low priory transit queue.
         Each node determines the threshold value for classifying what is considered
         a high priority packet and what is considered a low priority packet."
        ::= { rprIfEntry 1 }



rprIfNodesOnRing OBJECT-TYPE
    SYNTAX Integer32 (0..256)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of nodes in the RPR ring. When the interface is down the value is 0."
            ::= { rprIfEntry 2 }

rprIfTopologyTimer OBJECT-TYPE
    SYNTAX Integer32 (1..600)
    UNITS "Seconds"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Indicates how frequent topology-discovery messages are sent around the ring."
    DEFVAL { 10 }
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        ::= { rprIfEntry 3 }

rprIfProtectionWTR OBJECT-TYPE
   SYNTAX Integer32 (10..600)
   UNITS "Seconds"
   MAX-ACCESS read-write
   STATUS current
   DESCRIPTION
      "Indicates the length of time in seconds, to remain in the protection state,
       after the cause of an automatic protection is removed. This mechanism
       prevents protection switch oscillations."
   DEFVAL { 60 }
        ::= { rprIfEntry 4 }

rprIfTimeElapsed OBJECT-TYPE
   SYNTAX Integer32 (1..900)
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The number of seconds, including partial seconds, that have elapsed
       since the beginning of the current measurement interval. If, for some
       reason, such as an adjustment in the system's time-of-day clock, the
       current interval exceeds the maximum value, the agent will return the
       maximum value.

       This object is associated with the rprRingCountersCurrentTable,
       rprLocalCountersCurrentTable and rprErrorCountersCurrentTable."
      ::= { rprIfEntry 5 }

rprIfValidIntervals OBJECT-TYPE
   SYNTAX Integer32 (0..96)
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The number of previous 15-minute measurement intervals for which
       data was collected.
       An RPR interface must be capable of supporting up to 96 intervals.
       The value will be 96 unless the measurement was restarted within the
       last (96*15) minutes, in which case the value will be the number of
       complete 15 minute intervals for which the agent has at least some
       data.

       This object is associated with the rprRingCountersIntervalTable,
       rprLocalCountersIntervalTable and rprErrorCountersIntervalTable."
        ::= { rprIfEntry 6 }


--
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
-- RPR topology table
--

rprTopologyTable OBJECT-TYPE
   SYNTAX SEQUENCE OF RprTopologyEntry
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "A topology map that details the list of nodes on the RPR ring."
   ::= { rprGeneral 2 }

rprTopologyEntry OBJECT-TYPE
   SYNTAX RprTopologyEntry
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "Each entry contains information specific to a particular node on
       the ring."
   INDEX { ifIndex, rprTopologyHopsNumber }
   ::= { rprTopologyTable 1 }

RprTopologyEntry ::= SEQUENCE {
   rprTopologyHopsNumber                Unsigned32,
   rprTopologyMacAddress                MacAddress,
   rprTopologyNodeId                    SnmpAdminString,
   rprTopologyWrapSupport               TruthValue,
   rprTopologyProtectionStateEast       INTEGER,
   rprTopologyProtectionStateWest       INTEGER
   }

rprTopologyHopsNumber OBJECT-TYPE
   SYNTAX Unsigned32 (0..256)
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "The value between 0 and (rprIfNodesOnRing - 1) identifies the particular node,
       which is the number of hops around the ring in clockwise direction away from the local node."
   ::= { rprTopologyEntry 1 }

rprTopologyMacAddress OBJECT-TYPE
   SYNTAX MacAddress
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The 48-bit unique MAC address of the node."
   ::= { rprTopologyEntry 2 }

rprTopologyNodeId OBJECT-TYPE
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   SYNTAX SnmpAdminString (SIZE(1..32))
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "This variable contains the node ID, which can be used for any purpose."
   ::= { rprTopologyEntry 3 }

rprTopologyWrapSupport OBJECT-TYPE
   SYNTAX TruthValue
   MAX-ACCESS  read-only
   STATUS  current
   DESCRIPTION
      "A boolean flag to indicate if this node supports wrap protection."
   ::= { rprTopologyEntry 4 }

rprTopologyProtectionStateEast OBJECT-TYPE
   SYNTAX INTEGER {
      normal     (1),
      protected  (2)
      }
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Indicates if protection is currently active on the East side of the
       RPR interface."
   ::= { rprTopologyEntry 5 }

rprTopologyProtectionStateWest OBJECT-TYPE
   SYNTAX INTEGER {
      normal     (1),
      protected  (2)
      }
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Indicates if protection is currently active on the West side of the
       RPR interface."
   ::= { rprTopologyEntry 6 }



--
-- The RPR MAC side table
--

rprMacSideTable OBJECT-TYPE
   SYNTAX SEQUENCE OF RprMacSideEntry
   MAX-ACCESS not-accessible
   STATUS current
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    DESCRIPTION
        "A table of information per RPR side."
    ::= { rprGeneral 3 }

rprMacSideEntry OBJECT-TYPE
    SYNTAX RprMacSideEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of information specific to a particular side of a particular
         RPR interface."
    INDEX { ifIndex, rprMacInterfaceSide }
    ::= { rprMacSideTable 1 }

RprMacSideEntry ::= SEQUENCE {
    rprMacInterfaceSide                     RprInterfaceSide,
    rprMacProtectionActive                  TruthValue,
    rprMacProtectionCommand                 RprProtectionCommand,
    rprMacProtectionStatus                  RprProtectionStatus,
    rprMacProtectionCount                   Counter32,
    rprMacProtectionDuration                Counter32,
    rprMacProtectionLastActivationTime      TimeStamp
    }

rprMacInterfaceSide OBJECT-TYPE
    SYNTAX RprInterfaceSide
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An indication of the side for which this row contains information."
    ::= { rprMacSideEntry 1 }

rprMacProtectionActive OBJECT-TYPE
    SYNTAX TruthValue
    MAX-ACCESS  read-only
    STATUS  current
    DESCRIPTION
        "A boolean flag to indicate if this side of the RPR interface is
         currently engaged in active protection."
    ::= { rprMacSideEntry 2 }

rprMacProtectionCommand OBJECT-TYPE
    SYNTAX RprProtectionCommand
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "The protection mode requested by management for the local node
         that can affect the rprMacProtectionStatus of the RPR node,
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        according to the set of rules describing the RPR protection.

        When read this object returns the last command written or noCommand
        if no command has been written to this interface side since
        initialization.

        The return of the last command written does not imply that this
        command is currently in effect. This request may have been preempted
        by a higher priority event, for example a manual protection is
        preempted by a failure. In order to determine the current state of the
        interface side and the node it is necessary to read the objects
        rprMacProtectionStatus and rprMacProtectionActive.

        There is no pending of commands, that is if a command has been
        preempted by a failure, when the failure clears the command is not
        executed.

        If the command cannot be executed because an equal or higher priority
        request is in effect, an error is returned.

        A clear command on an interface that has no pending protection
        command, has no affect. A clear command clears an active WTR state."
    DEFVAL { noCommand }
    ::= { rprMacSideEntry 3 }

rprMacProtectionStatus OBJECT-TYPE
    SYNTAX RprProtectionStatus
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The current protection mode for the interface side.

        There is no pending of the waitToRestore status, that is if the
        restore period has been preempted by a higher priority event, when
        the event clears the restore does not continue."
    ::= { rprMacSideEntry 4 }

rprMacProtectionCount OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of transitions from idle state to active protection state."
    ::= { rprMacSideEntry 5 }

rprMacProtectionDuration OBJECT-TYPE
    SYNTAX Counter32
            UNITS  "seconds"
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "A count of the accumulative time that the specified side of the RPR
         interface was in active protection state."
    ::= { rprMacSideEntry 6 }

rprMacProtectionLastActivationTime OBJECT-TYPE
    SYNTAX TimeStamp
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of sysUpTime at the time of the last protection activation."
    ::= { rprMacSideEntry 7 }



--
-- RPR ring current counters table
--

rprRingCountersCurrentTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprRingCountersCurrentEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The physical medium dependent layer current counters table."
    ::= { rprRingCounters 1 }

rprRingCountersCurrentEntry OBJECT-TYPE
    SYNTAX RprRingCountersCurrentEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Packets and octets statistics for the current interval for the
         physical medium dependent layer of a particular side of a particular
         RPR interface.
         The corresponding instance of rprIfTimeElapsed indicates the
         number of seconds which have elapsed so far in the current interval."
    INDEX { ifIndex, rprRingCurInterfaceSide }
    ::= { rprRingCountersCurrentTable 1 }

RprRingCountersCurrentEntry ::= SEQUENCE {
    rprRingCurInterfaceSide            RprInterfaceSide,
    rprRingCurUcastLPriOutProPktsIn          Counter64,
    rprRingCurUcastLPriOutProOctetsIn        Counter64,
    rprRingCurUcastLPriInProPktsInCounter64,
    rprRingCurUcastLPriInProOctetsIn         Counter64,
    rprRingCurUcastHPriPktsIn                Counter64,
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   rprRingCurUcastHPriOctetsIn          Counter64,
   rprRingCurMcastLPriPktsIn            Counter64,
   rprRingCurMcastLPriOctetsIn          Counter64,
   rprRingCurMcastHPriPktsIn            Counter64,
   rprRingCurMcastHPriOctetsIn          Counter64,
   rprRingCurUcastLPriOutProPktsOut     Counter64,
   rprRingCurUcastLPriOutProOctetsOut   Counter64,
   rprRingCurUcastLPriInProPktsOut      Counter64,
   rprRingCurUcastLPriInProOctetsOut    Counter64,
   rprRingCurUcastHPriPktsOut           Counter64,
   rprRingCurUcastHPriOctetsOut         Counter64,
   rprRingCurMcastLPriPktsOut           Counter64,
   rprRingCurMcastLPriOctetsOut         Counter64,
   rprRingCurMcastHPriPktsOut           Counter64,
   rprRingCurMcastHPriOctetsOut         Counter64
   }

rprRingCurInterfaceSide OBJECT-TYPE
   SYNTAX RprInterfaceSide
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "An indication of the side of the interface for which this row
       contains information."
   ::= { rprRingCountersCurrentEntry 1 }

rprRingCurUcastLPriOutProPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast
       packets received from the Physical medium dependent layer in the
       current 15 minute interval."
   ::= { rprRingCountersCurrentEntry 2 }

rprRingCurUcastLPriOutProOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast octets
       received from the Physical medium dependent layer in the current
       15 minute interval."
   ::= { rprRingCountersCurrentEntry 3 }

rprRingCurUcastLPriInProPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    STATUS current
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
         packets received from the Physical medium dependent layer in the
         current 15 minute interval."
    ::= { rprRingCountersCurrentEntry 4 }

rprRingCurUcastLPriInProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
         octets received from the Physical medium dependent layer in the
         current 15 minute interval."
    ::= { rprRingCountersCurrentEntry 5 }

rprRingCurUcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         packets received from the Physical medium dependent layer in the
         current 15 minute interval."
    ::= { rprRingCountersCurrentEntry 6 }

rprRingCurUcastHPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         octets received from the Physical medium dependent layer in the
         current 15 minute interval."
    ::= { rprRingCountersCurrentEntry 7 }

rprRingCurMcastLPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast packets received
         from the Physical medium dependent layer in the current 15 minute
         interval."
    ::= { rprRingCountersCurrentEntry 8 }

rprRingCurMcastLPriOctetsIn OBJECT-TYPE
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets received
         from the Physical medium dependent layer in the current 15 minute
         interval."
    ::= { rprRingCountersCurrentEntry 9 }

rprRingCurMcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast packets received
         from the Physical medium dependent layer in the current 15 minute
         interval."
    ::= { rprRingCountersCurrentEntry 10 }

rprRingCurMcastHPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets received
         from the Physical medium dependent layer in the current 15 minute
         interval."
    ::= { rprRingCountersCurrentEntry 11 }

rprRingCurUcastLPriOutProPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast
         packets going to the Physical medium dependent layer in the current
         15 minute interval."
    ::= { rprRingCountersCurrentEntry 12 }

rprRingCurUcastLPriOutProOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast octets
         going to the Physical medium dependent layer in the current 15 minute
         interval."
    ::= { rprRingCountersCurrentEntry 13 }
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
rprRingCurUcastLPriInProPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       packets going to the Physical medium dependent layer in the current
       15 minute interval."
   ::= { rprRingCountersCurrentEntry 14 }

rprRingCurUcastLPriInProOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       octets going to the Physical medium dependent layer in the current
       15 minute interval."
   ::= { rprRingCountersCurrentEntry 15 }

rprRingCurUcastHPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       packets going to the Physical medium dependent layer in the current
       15 minute interval."
   ::= { rprRingCountersCurrentEntry 16 }

rprRingCurUcastHPriOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       octets going to the Physical medium dependent layer in the current 15
       minute interval."
   ::= { rprRingCountersCurrentEntry 17 }

rprRingCurMcastLPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast packets going to
       the Physical medium dependent layer in the current 15 minute
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        interval."
    ::= { rprRingCountersCurrentEntry 18 }

rprRingCurMcastLPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets going to
        the Physical medium dependent layer in the current 15 minute
        interval."
    ::= { rprRingCountersCurrentEntry 19 }

rprRingCurMcastHPriPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast packets going to
        the Physical medium dependent layer in the current 15 minute
        interval."
    ::= { rprRingCountersCurrentEntry 20 }

rprRingCurMcastHPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets going to
        the Physical medium dependent layer in the current 15 minute
        interval."
    ::= { rprRingCountersCurrentEntry 21 }



--
-- RPR ring interval counters
--

rprRingCountersIntervalTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprRingCountersIntervalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The physical medium dependent layer interval counters table."
    ::= { rprRingCounters 2 }

rprRingCountersIntervalEntry OBJECT-TYPE
    SYNTAX RprRingCountersIntervalEntry
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Packets and octets statistics collected for a particular interval
         for the physical medium dependent layer of a particular side of a
         particular RPR interface.
         The corresponding instance of rprIfValidIntervals indicates the
         number of intervals for which the set of statistics is available."
    INDEX { ifIndex,
            rprRingIntInterfaceSide,
            rprRingIntNumber }
    ::= { rprRingCountersIntervalTable 1 }

RprRingCountersIntervalEntry ::= SEQUENCE {
    rprRingIntInterfaceSide                 RprInterfaceSide,
    rprRingIntNumber                        Integer32,
    rprRingIntUcastLPriOutProPktsInCounter64,
    rprRingIntUcastLPriOutProOctetsIn       Counter64,
    rprRingIntUcastLPriInProPktsIn          Counter64,
    rprRingIntUcastLPriInProOctetsIn        Counter64,
    rprRingIntUcastHPriPktsIn               Counter64,
    rprRingIntUcastHPriOctetsIn             Counter64,
    rprRingIntMcastLPriPktsIn               Counter64,
    rprRingIntMcastLPriOctetsIn             Counter64,
    rprRingIntMcastHPriPktsIn               Counter64,
    rprRingIntMcastHPriOctetsIn             Counter64,
    rprRingIntUcastLPriOutProPktsOut        Counter64,
    rprRingIntUcastLPriOutProOctetsOut      Counter64,
    rprRingIntUcastLPriInProPktsOutCounter64,
    rprRingIntUcastLPriInProOctetsOut       Counter64,
    rprRingIntUcastHPriPktsOut              Counter64,
    rprRingIntUcastHPriOctetsOut            Counter64,
    rprRingIntMcastLPriPktsOut              Counter64,
    rprRingIntMcastLPriOctetsOut            Counter64,
    rprRingIntMcastHPriPktsOut              Counter64,
    rprRingIntMcastHPriOctetsOut            Counter64,
    rprRingIntValidData                     TruthValue
    }

rprRingIntInterfaceSide OBJECT-TYPE
    SYNTAX RprInterfaceSide
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An indication of the side of the interface for which this row
         contains information."
    ::= { rprRingCountersIntervalEntry 1 }

rprRingIntNumber OBJECT-TYPE
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    SYNTAX Integer32 (1..96)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
      "A number between 1 and 96, which identifies the intervals for which
       the set of statistics is available. The interval identified by 1 is
       the most recently completed 15 minute interval, and interval
       identified by N is the interval immediately preceding the one
       identified by N-1."
    ::= { rprRingCountersIntervalEntry 2 }

rprRingIntUcastLPriOutProPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast
       packets received from the Physical medium dependent layer in a
       particular 15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 3 }

rprRingIntUcastLPriOutProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast octets
       received from the Physical medium dependent layer in a particular
       15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 4 }

rprRingIntUcastLPriInProPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       packets received from the Physical medium dependent layer in a
       particular 15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 5 }

rprRingIntUcastLPriInProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       octets received from the Physical medium dependent layer in a
       particular 15-minute interval in the past 24 hours."
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    ::= { rprRingCountersIntervalEntry 6 }

rprRingIntUcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         packets received from the Physical medium dependent layer in a
         particular 15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 7 }

rprRingIntUcastHPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         octets received from the Physical medium dependent layer in a
         particular 15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 8 }

rprRingIntMcastLPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priroity multicast packets received
         from the Physical medium dependent layer in a particular 15-minute
         interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 9 }

rprRingIntMcastLPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets received
         from the Physical medium dependent layer in a particular 15-minute
         interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 10 }

rprRingIntMcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priroity multicast packets received
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        from the Physical medium dependent layer in a particular 15-minute
        interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 11 }

rprRingIntMcastHPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets received
        from the Physical medium dependent layer in a particular 15-minute
        interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 12 }

rprRingIntUcastLPriOutProPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast
        packets going to the Physical medium dependent layer in a particular
        15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 13 }

rprRingIntUcastLPriOutProOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast octets
        going to the Physical medium dependent layer in a particular
        15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 14 }

rprRingIntUcastLPriInProPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
        packets going to the Physical medium dependent layer in a particular
        15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 15 }

rprRingIntUcastLPriInProOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
         octets going to the Physical medium dependent layer in a particular
         15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 16 }

rprRingIntUcastHPriPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         packets going to the Physical medium dependent layer in a particular
         15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 17 }

rprRingIntUcastHPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority unicast
         octets going to the Physical medium dependent layer in a particular
         15-minute interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 18 }

rprRingIntMcastLPriPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast packets going to
         the Physical medium dependent layer in a particular 15-minute
         interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 19 }

rprRingIntMcastLPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets going to
         the Physical medium dependent layer in a particular 15-minute
         interval in the past 24 hours."
    ::= { rprRingCountersIntervalEntry 20 }

rprRingIntMcastHPriPktsOut OBJECT-TYPE
    SYNTAX Counter64
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of highpriority multicast packets going to
       the Physical medium dependent layer in a particular 15-minute
       interval in the past 24 hours."
   ::= { rprRingCountersIntervalEntry 21 }

rprRingIntMcastHPriOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority multicast octets going to
       the Physical medium dependent layer in a particular 15-minute
       interval in the past 24 hours."
   ::= { rprRingCountersIntervalEntry 22 }

rprRingIntValidData OBJECT-TYPE
   SYNTAX TruthValue
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "This variable indicates if the data for this interval is valid."
   ::= { rprRingCountersIntervalEntry 23 }


--
-- RPR local current counters table
--

rprLocalCountersCurrentTable OBJECT-TYPE
   SYNTAX SEQUENCE OF RprLocalCountersCurrentEntry
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "The local node traffic current counters table."
   ::= { rprLocalCounters 1 }

rprLocalCountersCurrentEntry OBJECT-TYPE
   SYNTAX RprLocalCountersCurrentEntry
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "Packets and octets statistics for the current interval for the local
       node traffic of a particular side of a particular RPR interface.
       The corresponding instance of rprIfTimeElapsed indicates the
       number of seconds which have elapsed so far in the current interval."
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    INDEX { ifIndex, rprLocalCurInterfaceSide }
    ::= { rprLocalCountersCurrentTable 1 }

RprLocalCountersCurrentEntry ::= SEQUENCE {
    rprLocalCurInterfaceSide                RprInterfaceSide,
    rprLocalCurUcastLPriOutProPktsIn        Counter64,
    rprLocalCurUcastLPriOutProOctetsIn      Counter64,
    rprLocalCurUcastLPriInProPktsInCounter64,
    rprLocalCurUcastLPriInProOctetsIn       Counter64,
    rprLocalCurUcastHPriPktsIn              Counter64,
    rprLocalCurUcastHPriOctetsIn            Counter64,
    rprLocalCurMcastLPriPktsIn              Counter64,
    rprLocalCurMcastLPriOctetsIn            Counter64,
    rprLocalCurMcastHPriPktsIn              Counter64,
    rprLocalCurMcastHPriOctetsIn            Counter64,
    rprLocalCurUcastLPriOutProPktsOut       Counter64,
    rprLocalCurUcastLPriOutProOctetsOut     Counter64,
    rprLocalCurUcastLPriInProPktsOut        Counter64,
    rprLocalCurUcastLPriInProOctetsOut      Counter64,
    rprLocalCurUcastHPriPktsOut             Counter64,
    rprLocalCurUcastHPriOctetsOut           Counter64,
    rprLocalCurMcastLPriPktsOut             Counter64,
    rprLocalCurMcastLPriOctetsOut           Counter64,
    rprLocalCurMcastHPriPktsOut             Counter64,
    rprLocalCurMcastHPriOctetsOut           Counter64
    }

rprLocalCurInterfaceSide OBJECT-TYPE
    SYNTAX RprInterfaceSide
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An indication of the side of the interface for which this row
         contains information."
    ::= { rprLocalCountersCurrentEntry 1 }

rprLocalCurUcastLPriOutProPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast
         packets received from the local node in the current 15 minute
         interval."
    ::= { rprLocalCountersCurrentEntry 2 }

rprLocalCurUcastLPriOutProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast octets
       received from the local node in the current 15 minute interval."
   ::= { rprLocalCountersCurrentEntry 3 }

rprLocalCurUcastLPriInProPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       packets received from the local node in the current 15 minute
       interval."
   ::= { rprLocalCountersCurrentEntry 4 }

rprLocalCurUcastLPriInProOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       octets received from the local node in the current 15 minute
       interval."
   ::= { rprLocalCountersCurrentEntry 5 }

rprLocalCurUcastHPriPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       packets received from the local node in the current 15 minute
       interval."
   ::= { rprLocalCountersCurrentEntry 6 }

rprLocalCurUcastHPriOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       octets received from the local node in the current 15 minute
       interval."
   ::= { rprLocalCountersCurrentEntry 7 }

rprLocalCurMcastLPriPktsIn OBJECT-TYPE
   SYNTAX Counter64
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast packets received
         from the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 8 }

rprLocalCurMcastLPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets received
         from the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 9 }

rprLocalCurMcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast packets received
         from the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 10 }

rprLocalCurMcastHPriOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets received
         from the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 11 }

rprLocalCurUcastLPriOutProPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast
         packets going to the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 12 }

rprLocalCurUcastLPriOutProOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
      "The counter associated with the number of out of profile low priority unicast octets
      going to the local node in the current 15 minute interval."
   ::= { rprLocalCountersCurrentEntry 13 }

rprLocalCurUcastLPriInProPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
      packets going to the local node in the current 15 minute interval."
   ::= { rprLocalCountersCurrentEntry 14 }

rprLocalCurUcastLPriInProOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
      octets going to the local node in the current 15 minute interval."
   ::= { rprLocalCountersCurrentEntry 15 }

rprLocalCurUcastHPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
      packets going to the local node in the current 15 minute interval."
   ::= { rprLocalCountersCurrentEntry 16 }

rprLocalCurUcastHPriOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
      octets going to the local node in the current 15 minute
      interval."
   ::= { rprLocalCountersCurrentEntry 17 }

rprLocalCurMcastLPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast packets going to
      the local node in the current 15 minute interval."
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    ::= { rprLocalCountersCurrentEntry 18 }

rprLocalCurMcastLPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of low priority multicast octets going to
         the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 19 }

rprLocalCurMcastHPriPktsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast packets going to
         the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 20 }

rprLocalCurMcastHPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets going to
         the local node in the current 15 minute interval."
    ::= { rprLocalCountersCurrentEntry 21 }



--
-- RPR local interval counters table
--

rprLocalCountersIntervalTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprLocalCountersIntervalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The local node traffic interval counters table."
    ::= { rprLocalCounters 2 }

rprLocalCountersIntervalEntry OBJECT-TYPE
    SYNTAX RprLocalCountersIntervalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Packets and octets statistics collected for a particular interval for
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        local node traffic of a particular side of a particular RPR interface.
        The corresponding instance of rprIfValidIntervals indicates the
        number of intervals for which the set of statistics is available."
    INDEX { ifIndex,
            rprLocalIntInterfaceSide,
            rprLocalIntNumber }
    ::= { rprLocalCountersIntervalTable 1 }

RprLocalCountersIntervalEntry ::= SEQUENCE {
    rprLocalIntInterfaceSide          RprInterfaceSide,
    rprLocalIntNumber                 Integer32,
    rprLocalIntUcastLPriOutProPktsIn  Counter64,
    rprLocalIntUcastLPriOutProOctetsIn Counter64,
    rprLocalIntUcastLPriInProPktsIn   Counter64,
    rprLocalIntUcastLPriInProOctetsIn Counter64,
    rprLocalIntUcastHPriPktsIn        Counter64,
    rprLocalIntUcastHPriOctetsIn      Counter64,
    rprLocalIntMcastLPriPktsIn        Counter64,
    rprLocalIntMcastLPriOctetsIn      Counter64,
    rprLocalIntMcastHPriPktsIn        Counter64,
    rprLocalIntMcastHPriOctetsIn      Counter64,
    rprLocalIntUcastLPriOutProPktsOut Counter64,
    rprLocalIntUcastLPriOutProOctetsOut Counter64,
    rprLocalIntUcastLPriInProPktsOut  Counter64,
    rprLocalIntUcastLPriInProOctetsOut Counter64,
    rprLocalIntUcastHPriPktsOut       Counter64,
    rprLocalIntUcastHPriOctetsOut     Counter64,
    rprLocalIntMcastLPriPktsOut       Counter64,
    rprLocalIntMcastLPriOctetsOut     Counter64,
    rprLocalIntMcastHPriPktsOut       Counter64,
    rprLocalIntMcastHPriOctetsOut     Counter64,
    rprLocalIntValidData              TruthValue
    }

rprLocalIntInterfaceSide OBJECT-TYPE
    SYNTAX RprInterfaceSide
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "An indication of the side of the interface for which this row
        contains information."
    ::= { rprLocalCountersIntervalEntry 1 }

rprLocalIntNumber OBJECT-TYPE
    SYNTAX Integer32 (1..96)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A number between 1 and 96, which identifies the interval for which
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        the set of statistics is available. The interval identified by 1 is
        the most recently completed 15 minute interval, and interval
        identified by N is the interval immediately preceding the one
        identified by N-1."
    ::= { rprLocalCountersIntervalEntry 2 }

rprLocalIntUcastLPriOutProPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast
        packets received from the local node in a particular 15-minute
        interval in the past 24 hours."
    ::= { rprLocalCountersIntervalEntry 3 }

rprLocalIntUcastLPriOutProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of out of profile low priority unicast octets
        received from the local node in a particular 15-minute interval in
        the past 24 hours."
    ::= { rprLocalCountersIntervalEntry 4 }

rprLocalIntUcastLPriInProPktsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
        packets received from the local node in a particular 15-minute
        interval in the past 24 hours."
    ::= { rprLocalCountersIntervalEntry 5 }

rprLocalIntUcastLPriInProOctetsIn OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of in profile low priority unicast
        octets received from the local node in a particular 15-minute
        interval in the past 24 hours."
    ::= { rprLocalCountersIntervalEntry 6 }

rprLocalIntUcastHPriPktsIn OBJECT-TYPE
    SYNTAX Counter64
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       packets received from the local node in a particular 15-minute
       interval in the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 7 }

rprLocalIntUcastHPriOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       octets received from the local node in a particular 15-minute
       interval in the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 8 }

rprLocalIntMcastLPriPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast packets received
       from the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 9 }

rprLocalIntMcastLPriOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast octets received
       from the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 10 }

rprLocalIntMcastHPriPktsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority multicast packets received
       from the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 11 }
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
rprLocalIntMcastHPriOctetsIn OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority multicast octets received
       from the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 12 }

rprLocalIntUcastLPriOutProPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast
       packets going to the local node in a particular 15-minute interval in
       the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 13 }

rprLocalIntUcastLPriOutProOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority unicast octets
       going to the local node in a particular 15-minute interval in the
       past 24 hours."
   ::= { rprLocalCountersIntervalEntry 14 }

rprLocalIntUcastLPriInProPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       packets going to the local node in a particular 15-minute interval in
       the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 15 }

rprLocalIntUcastLPriInProOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority unicast
       octets going to the local node in a particular 15-minute interval in
       the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 16 }
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
rprLocalIntUcastHPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       packets going to the local node in a particular 15-minute interval
       in the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 17 }

rprLocalIntUcastHPriOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority unicast
       octets going to the local node in a particular 15-minute interval
       in the past 24 hours."
   ::= { rprLocalCountersIntervalEntry 18 }

rprLocalIntMcastLPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast packets going to
       the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 19 }

rprLocalIntMcastLPriOctetsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of low priority multicast octets going to
       the local node in a particular 15-minute interval in the past 24
       hours."
   ::= { rprLocalCountersIntervalEntry 20 }

rprLocalIntMcastHPriPktsOut OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority multicast packets going to
       the local node in a particular 15-minute interval in the past 24
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        hours."
    ::= { rprLocalCountersIntervalEntry 21 }

rprLocalIntMcastHPriOctetsOut OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of high priority multicast octets going to
        the local node in a particular 15-minute interval in the past 24
        hours."
    ::= { rprLocalCountersIntervalEntry 22 }

rprLocalIntValidData OBJECT-TYPE
    SYNTAX TruthValue
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This variable indicates if the data for this interval is valid."
    ::= { rprLocalCountersIntervalEntry 23 }



--
-- RPR error current counters
--

rprErrorCountersCurrentTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprErrorCountersCurrentEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The RPR Errors Current counters table."
    ::= { rprErrorCounters 1 }

rprErrorCountersCurrentEntry OBJECT-TYPE
    SYNTAX RprErrorCountersCurrentEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Errors statistics for the current interval of a particular side of a
        particular RPR interface.
        The corresponding instance of rprIfTimeElapsed indicates the
        number of seconds which have elapsed so far in the current interval."
    INDEX { ifIndex, rprErrorCurInterfaceSide }
    ::= { rprErrorCountersCurrentTable 1 }

RprErrorCountersCurrentEntry ::= SEQUENCE {
    rprErrorCurInterfaceSide            RprInterfaceSide,
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
   rprErrorCurRingLPriOutProPktsTtlExp    Counter64,
   rprErrorCurRingLPriInProPktsTtlExp     Counter64,
   rprErrorCurRingHPriPktsTtlExp          Counter64,
   rprErrorCurRingGiantPkts               Counter64,
   rprErrorCurRingHecPkts         Counter64,
   rprErrorCurRingFcsPkts         Counter64,
   rprErrorCurRingSelfSrcPkts             Counter64,
   rprErrorCurRingUnknownProtPkts         Counter64,
   rprErrorCurRingPmdAbortPkts            Counter64,
   rprErrorCurLocalGiantPkts              Counter64
   }

rprErrorCurInterfaceSide OBJECT-TYPE
   SYNTAX RprInterfaceSide
   MAX-ACCESS not-accessible
   STATUS current
   DESCRIPTION
      "An indication of the side of the interface for which this row
       contains information."
   ::= { rprErrorCountersCurrentEntry 1 }

rprErrorCurRingLPriOutProPktsTtlExp OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of out of profile low priority packets
       received in the current 15 minute interval from the Physical medium
       dependent layer and were dropped due to zero Time To Live (TTL)."
   ::= { rprErrorCountersCurrentEntry 2 }

rprErrorCurRingLPriInProPktsTtlExp OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority packets
       received in the current 15 minute interval from the Physical medium
       dependent layer and were dropped due to zero Time To Live (TTL)."
   ::= { rprErrorCountersCurrentEntry 3 }

rprErrorCurRingHPriPktsTtlExp OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority packets
       received in the current 15 minute interval from the Physical medium
       dependent layer and were dropped due to zero Time To Live (TTL)."
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    ::= { rprErrorCountersCurrentEntry 4 }

rprErrorCurRingGiantPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
         current 15 minute interval from the Physical medium dependent layer
         that exceed the maximum permitted packet size."
    ::= { rprErrorCountersCurrentEntry 5 }

rprErrorCurRingHecPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
         current 15 minute interval from the Physical medium dependent layer
         with HEC error."
    ::= { rprErrorCountersCurrentEntry 6 }

rprErrorCurRingFcsPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
         current 15 minute interval from the Physical medium dependent layer
         with FCS error."
    ::= { rprErrorCountersCurrentEntry 7 }

rprErrorCurRingSelfSrcPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
         current 15 minute interval from the Physical medium dependent layer
         that were transmitted by the node itself i.e. the source MAC is equal to the interface MAC."
    ::= { rprErrorCountersCurrentEntry 8 }

rprErrorCurRingUnknownProtPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        current 15 minute interval from the Physical medium dependent layer
        with unknown or unsupported type value."
    ::= { rprErrorCountersCurrentEntry 9 }

rprErrorCurRingPmdAbortPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets that were aborted by the
        Physical medium dependent layer in the current 15 minute interval."
    ::= { rprErrorCountersCurrentEntry 10 }

rprErrorCurLocalGiantPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in the
        current 15 minute interval from the local node that exceed the maximum permitted packet size."
    ::= { rprErrorCountersCurrentEntry 11 }



--
-- RPR error interval counters table
--

rprErrorCountersIntervalTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RprErrorCountersIntervalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The RPR Errors Interval counters table."
    ::= { rprErrorCounters 2 }

rprErrorCountersIntervalEntry OBJECT-TYPE
    SYNTAX RprErrorCountersIntervalEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Error statistics collected for a particular interval of a particular
        side of a particular RPR interface.
        The corresponding instance of rprIfValidIntervals indicates the
        number of intervals for which the set of statistics is available."
    INDEX { ifIndex,
            rprErrorIntInterfaceSide,
            rprErrorIntNumber }
    ::= { rprErrorCountersIntervalTable 1 }
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
RprErrorCountersIntervalEntry ::= SEQUENCE
    {
  rprErrorIntInterfaceSide                  RprInterfaceSide,
  rprErrorIntNumber                         Integer32,
  rprErrorIntRingLPriOutProPktsTtlExp       Counter64,
  rprErrorIntRingLPriInProPktsTtlExp        Counter64,
  rprErrorIntRingHPriPktsTtlExp             Counter64,
  rprErrorIntRingGiantPkts                  Counter64,
  rprErrorIntRingHecPkts                    Counter64,
  rprErrorIntRingFcsPkts                    Counter64,
  rprErrorIntRingSelfSrcPkts                Counter64,
  rprErrorIntRingUnknownProtPktsCounter64,
  rprErrorIntRingPmdAbortPkts               Counter64,
  rprErrorIntLocalGiantPkts                 Counter64,
  rprErrorIntValidData                      TruthValue
        }

rprErrorIntInterfaceSide OBJECT-TYPE
  SYNTAX RprInterfaceSide
  MAX-ACCESS not-accessible
  STATUS current
  DESCRIPTION
    "An indication of the side of the interface for which this row
     contains information."
  ::= { rprErrorCountersIntervalEntry 1 }

rprErrorIntNumber OBJECT-TYPE
  SYNTAX Integer32 (1..96)
  MAX-ACCESS not-accessible
  STATUS current
  DESCRIPTION
    "A number between 1 and 96, which identifies the interval for which
     the set od statistics is available. The interval identified by 1 is
     the most recently completed 15 minute interval, and interval
     identified by N is the interval immediately preceding the one
     identified by N-1."
  ::= { rprErrorCountersIntervalEntry 2 }

rprErrorIntRingLPriOutProPktsTtlExp OBJECT-TYPE
  SYNTAX Counter64
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
    "The counter associated with the number of out of profile low priority packets
     received in a particular 15-minute interval in the past 24 hours
     from the Physical medium dependent layer and were dropped due to
     zero Time To Live (TTL)."
  ::= { rprErrorCountersIntervalEntry 3 }
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
rprErrorIntRingLPriInProPktsTtlExp OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of in profile low priority packets
       received in a particular 15-minute interval in the past 24 hours
       from the Physical medium dependent layer and were dropped due to
       zero Time To Live (TTL)."
   ::= { rprErrorCountersIntervalEntry 4 }

rprErrorIntRingHPriPktsTtlExp OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of high priority packets
       received in a particular 15-minute interval in the past 24 hours
       from the Physical medium dependent layer and were dropped due to
       zero Time To Live (TTL)."
   ::= { rprErrorCountersIntervalEntry 5 }

rprErrorIntRingGiantPkts OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of packets received in a
       particular 15-minute interval in the past 24 hours from the Physical
       medium dependent layer that exceed the maximum permitted packet size."
   ::= { rprErrorCountersIntervalEntry 6 }

rprErrorIntRingHecPkts OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "The counter associated with the number of packets received in a
       particular 15-minute interval in the past 24 hours from the Physical
       medium dependent layer with HEC error."
   ::= { rprErrorCountersIntervalEntry 7 }

rprErrorIntRingFcsPkts OBJECT-TYPE
   SYNTAX Counter64
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
        "The counter associated with the number of packets received in a
         particular 15-minute interval in the past 24 hours from the Physical
         medium dependent layer with FCS error."
    ::= { rprErrorCountersIntervalEntry 8 }

rprErrorIntRingSelfSrcPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in a
         particular 15-minute interval in the past 24 hours from the Physical
         medium dependent layer that were transmitted
         by the node itself i.e. the source MAC is equal to the interface MAC."
    ::= { rprErrorCountersIntervalEntry 9 }

rprErrorIntRingUnknownProtPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in a
         particular 15 minute interval in the past 24 hours from the Physical
         medium dependent layer with unknown or unsupported type value."
    ::= { rprErrorCountersIntervalEntry 10 }

rprErrorIntRingPmdAbortPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets that were aborted by the
         Physical medium dependent layer in a particular 15 minute interval in the past 24 hours."
    ::= { rprErrorCountersIntervalEntry 11 }

rprErrorIntLocalGiantPkts OBJECT-TYPE
    SYNTAX Counter64
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The counter associated with the number of packets received in a
         particular 15-minute interval in the past 24 hours from the local
         node that exceed the maximum permitted
         packet size."
    ::= { rprErrorCountersIntervalEntry 12 }

rprErrorIntValidData OBJECT-TYPE
    SYNTAX TruthValue
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "This variable indicates if the data for this interval is valid."
    ::= { rprErrorCountersIntervalEntry 13 }


END
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex F

(normative)

Bridging Conformance

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

There are a large number of references to the "802.17 MAC" and "802.17" in this annex. This is inconsistent with normal 802 editorial practice. These references should be changed to "RPR MAC" and "this standard", respectively.

F.1 Bridging Overview

This section of the draft is not intended to define a new Bridging specification, but simply to extend the 802.17 MAC definition to demonstrate conformance to Transparent and VLAN Bridging , as defined in the IEEE 802.1D and 802.1Q standards respectively.

The MAC Bridging reference model for 802.17 is shown in Figure 1-1. The stations on the Ring act as the MAC Bridge, while the Ring acts as the shared media. Stations acting as Bridges are configured to do so through the appropriate Layer Management function.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

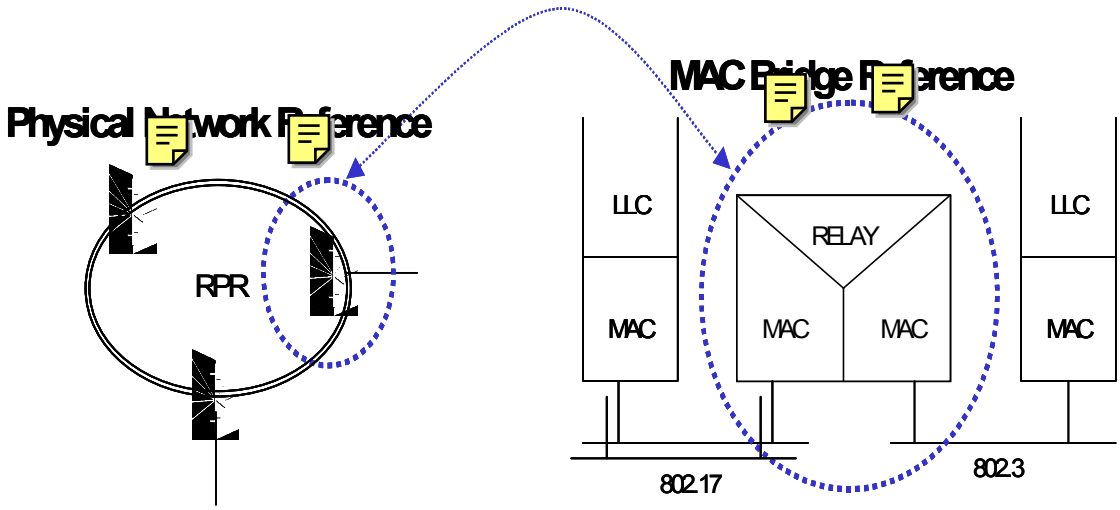


Figure C1—Bridge Architecture Reference

The 802.17 MAC conformance to the aforementioned Bridging standards[1] can be achieved with the following 802.17 MAC capabilities:

1) The 802.17 MAC must provide an Internal Sub-Layer Service (ISS), which is used to interface with the Bridging Entity Relay. The ISS will conform to Section 6.4 of the IEEE Std 802.1D and Section 7.1 of IEEE Std 802.1Q, when appropriate.
2) The 802.17 MAC must be able to communicate with the Bridge Protocol Entity via the LLC sub-layer, in conformance with the Bridging specifications.
3) The 802.17 MAC must be able to receives frames from the ring (i.e., shared media) and determine whether they need to be bridged or not.
4) The 802.17 MAC must be able to transmit bridged frames appropriately over the 802.17 shared media. This includes handling of unknown unicast, broadcast, and multicast frames.

F.2 802.17 MAC Internal Sub-Layer Service

The ISS is provided by a MAC Entity to communicate with the MAC Relay Entity. The interface for this sub-layer is predefined in the 802.1D and 802.1Q specifications. The 802.17 MAC will adhere to these specifications.

> ***Editors' Notes:*** *To be removed prior to final publication.*
>
> *Need more formal definition of above references. Also, it is not clear how this Annex can mandate behavior that is also covered by the MAC clause.*

F.2.1 802.17 MAC Support of Internal Sub-Layer Service

[1] IEEE 802.1D and 802.1Q Standards.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The 802.17 MAC access method is specified in the draft. Clause X specifies the MAC frame structure, and Clause Y specifies the MAC method.

On receipt of a MA_UNITDATA.request primitive, the local 802.17 MAC entity performs Transmit Encapsulation, assembling a frame using the parameters supplied as specified below.

On receipt of an 802.17 MAC frame by Receive Media Access Management, the MAC frame is passed to the reconciliation sub-layer which disassembles the frame into parameters, as specified below, that are supplied with the MA_UNITDATA.indication primitive.

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

The **mac_action** parameter takes only the value *request_with_no_response* and is not explicitly encoded in MAC frames.

The **destination_address** parameter is either 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 provided in a data request primitive is not encoded in the 802.17 MAC frame. The user_priority parameter provided in a data indication primitive takes the value of the Default User Priority parameter for the Port through which the MAC frame was received.

The **access_priority** parameter provided in a data request primitive is derived by a fixed 802.17 hod mapping. The fixed mapping is depicted in Table , “The access_priority parameter provided in a data request primitive is derived by a fixed 802.17 hod mapping. The fixed mapping is depicted in . The values shown are not modifiable my management or other means.,” on page 229. The values shown are not modifiable my management or other means.

User Priority	**Outbound Access Priority per MAC method**									
	802.3	**802.17**	**8802-4**	**8802-5 (default)**	**8802-5 (alternate)**	**8802-6**	**802.9a**	**8802.11**	**8802-12**	**FDDI**
0	0	0	0	0	4	0	0	0	0	0
1	0	1	1	1	4	1	0	0	0	1
2	0	2	2	2	4	2	0	0	0	2
3	0	3	3	3	4	3	0	0	0	3
4	0	4	4	4	4	4	0	0	4	4
5	0	5	5	5	5	5	0	0	4	5
6	0	6	6	6	6	6	0	0	4	6
7	0	7	7	6	6	7	0	0	4	6

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

The **frame_check_sequence** parameter is encoded in the FCS field of the MAC frame. The FCS is computed as a function of the destination address, source address, length, RPR Header, and data fields. If a MA_UNITDATA.request primitive is not accompanied by this parameter, it is calculated in accordance with Clause Z of this draft.

Figure C2 below shows the mapping of the MA-UNITDATA.request primitive parameters to the 802.17 MAC frame fields, and the mapping of the 802.17 MAC frame fields to the MA-UNITDATA.indication primitive parameters.

Table C1—Outbound Access Priorities

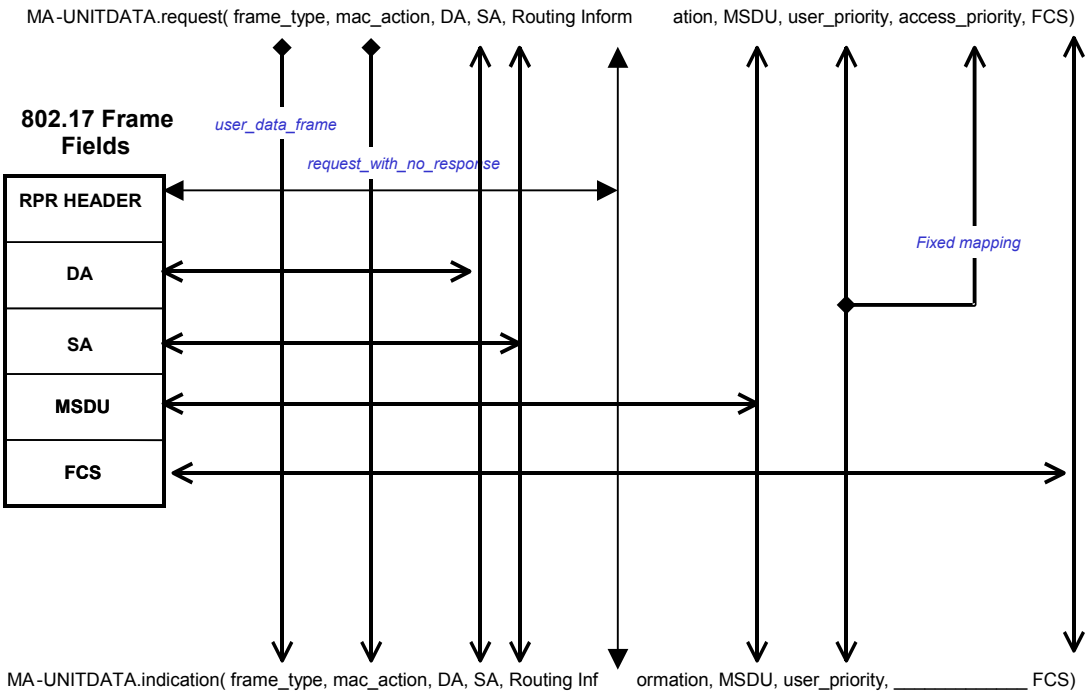


Figure C2—Mapping of MAC Service Primitives

F.2.2 802.17 MAC Support of Enhanced Internal Sub-Layer Service

An Enhanced Internal Sub-Layer Service (E-ISS) is derived from the Internal Sub-Layer Service (ISS, defined in ISO/IEC 15802-2, and by augmenting that specification with elements necessary to support operation of the tagging and un-tagging functions of the MAC Bridge. The E-ISS provided by the 802.17 MAC will conform to Section 7.1 of IEEE 802.1Q Std.

Figure C3 below shows the mapping of the EM-UNITDATA.request primitive parameters to the 802.17 MAC frame fields, and the mapping of the 802.17 MAC frame fields to the EM-UNITDATA.indication primitive parameters.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

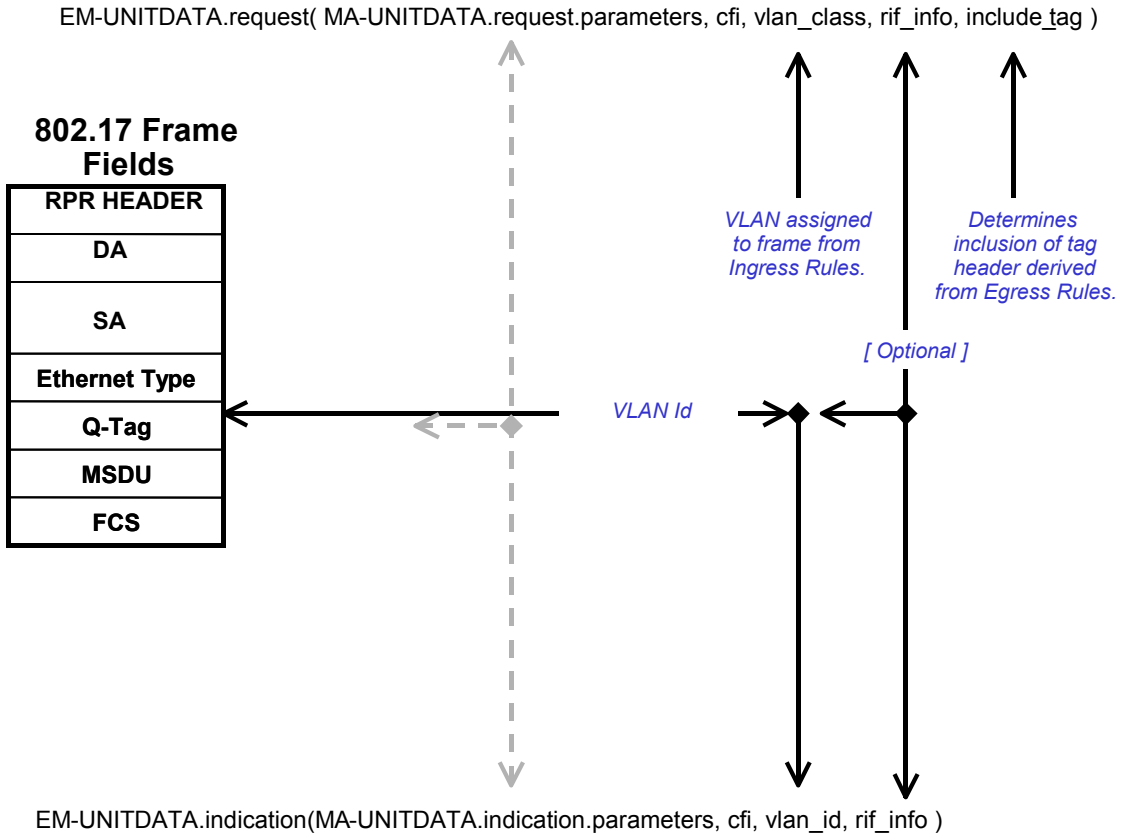


Figure C3—Mapping of Enhanced MAC Service Primitives

F.3 Bridge Protocol Entity Interactions

The 802.17 MAC will provide a MAC sub-layer that will conform to Section 2.2.2 of IEEE 802.2 Std.

F.4 802.17 MAC Handling of Frames to be Bridged

The 802.17 MAC transit data path needs to incorporate logic to determine whether the received frame should be:

Dropped. The frame is stripped from the Ring and passed to a MAC client.
Discarded. The frame is stripped from the Ring and not passed to any MAC client. The frame is discarded.
Passed Through. The frame is passed to the tandem buffer and dispatched to the outgoing Ringlet.
Replicated. The frame is replicated prior to the transit path drop point. One copy of the frame is Passed Through, and the other copy is Dropped.

Editors' Notes: *To be removed prior to final publication.*

The above should be a lettered or numbered list.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Figure C4 depicts a simplified depcition of the 802.17 MAC transit data path. Refer to Clause W of the 802.17 Draft for a complete description of the 802.17 MAC transit path.

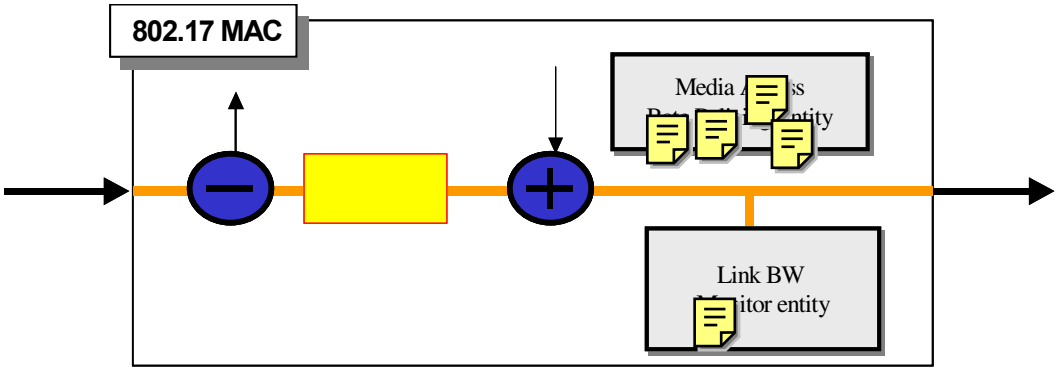


Figure C4—Simplified 802.17 MAC transit path

When bridging is provision on the RPR station, the Layer Management Entity sets a state used by the 802.17 MAC transit path to indicate that Bridging is configured. The Drop/Forward point in the 802.17 MAC transit path needs to incorporate the following logic:

a) If the Destination Address (DA) associated with the received frame is that of the RPR station, and Bridging is configured on the station, the frame is Dropped and passed to the MAC Relay.
b) If the DA of the received frame is not that of the RPR station, and Bridging is configured on the station, the frame is replicated.

The aforementioned logic is an extension of the RPR MAC reception rules outlined in Clause W of the 802.17 Draft.

F.5 802.17 MAC Transmission of Bridged Frames

Bridge relayed frames are submitted for transmission by the Bridging Forwarding Process. The Service request primitive associated with such a frame conveys the values of the Source and Destination Address fields received in the Service indication primitive. Refer to Figure 1-2 and Figure 1-3 for the mappings.

Bridged frames with a Multicast and Broadcast destination address are broadcast around the RPR.

Bridged frames with a destination address of a station on the RPR (i.e., a known destination address) is forwarded to the station using internal 802.17 MAC topology and steering tables.

Bridged frames with an unknown destination address (e.g., a destination address not matching a RPR station address) are flooded over the RPR.

F.5.1 Flooding Packet over 802.17

An 802.17 MAC floods a packet of the 802.17 shared media, by replicating and dispatching the packet(s) over both directions of the Ring. The TTL field, found in the RPR header, is set such that each station on the Ring only sees the packet once. Figure 1-5 below illustrates the operation of flooding a packet over the RPR.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

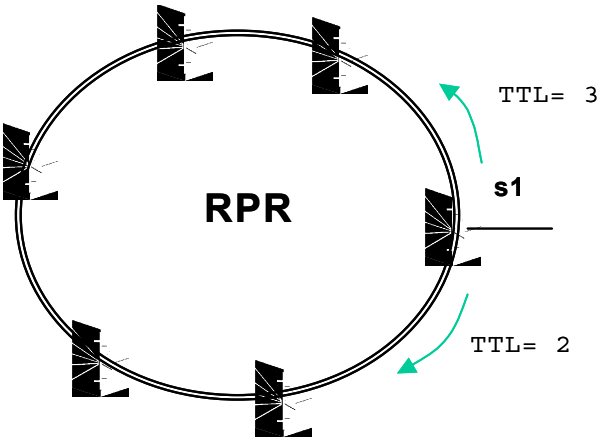


Figure C5—Flooding packets over RPR

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex G

(normative)

CRC Calculation

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

This annex is completely blank. Contributions are requested for material to be placed in it. Alternatively, it should be removed.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex H

(informative)

C Code Examples

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

This Annex provides code examples that illustrate the computation of the CRC and also the operation of the RPR fairness algorithm and MAC datapath. The code in this Annex has been provided purely for informative purposes, and should not be construed as mandating any particular implementation. In the event that a conflict arises between the contents of this Annex and some other normative portion of this standard, the normative portion shall take precedence.

Editors' Note: *To be removed prior to final publication.*

The code style and format are not yet finalized.

The syntax used for the following code examples conforms to ANSI X3T9-1995.

Editors' Note: *To be removed prior to final publication.*

Provide the proper reference for the ANSI C standard above.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// ******************************************************************************
// ***************** Coding Style - V0.10 dated: 01Feb2002 **********************
// ******************************************************************************
//                                                                          1       1       1       1
//      1       2       3       4       5       6       7       8       9       0       1       2       3
//
345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
678901234567890123456789012

//
// ******************************************************************************
// ********************************** COMMENTING ********************************
// ******************************************************************************
// The "//" commenting convention shall be used.
// An in-line comment shall be properly indented.
// comment may follow the statement line:
//
// Sample code is preceeded by "| " , as shown below
// | // Checking for exception conditions
// | if (x > 0)         // Warning processing code
// |   goto Label;
//
// ******************************************************************************
// ***************************** NAMING CONVENTIONS *****************************
// ******************************************************************************
// The code shall use these conventions for symbol name formats:
//   MY_DEFINE    - All caps and underscores between words
//                  Used when naming defines and enum values
//
//   MyType       - First letter is upper case; each run-together word is capitalized
//   MyRoutine()  - Used when naming typedefs and subroutine names
//
//   myData       - First letter is lower case; each run-together word is capitalized
//   myField      - Used when naming variable names, and data-structure fields.
//
// Don't use abbreviations that are unclear to others and save only a few keystrokes.
// Examples follow:
//
// | #define MY_DEFINES
// | enum values {
// |  STATUS_GOOD,
// |  STATUS_POOR,
// |  STATUS_FATAL
// | };
//
// | typedef struct {
// |   int hiPart;
// |   int loPart;
// | } PairOfInts;
//
// | uInt4 myData;
// | sInt4 MyRoutine();

// ******************************************************************************
// ***************************** INDENTATION STYLE ******************************
// ******************************************************************************
// Each level of code uses a 4-space indentation.
// Initial "{" braces shall appear at end of line.
// Final "}" braces shall appear at start of line.
//
// If only a single statement is enclosed, then curly braces
// should not be used. In "if/else" sequences, curly braches
// should be used unless both both blocks are single statements.
// Indentation for common structures is illustrated below.

// | if (condition)
// |   statement1;
// |
// | if (condition)
// |   statement1;
// | else
// |   statement2;
// |
// | if (condition) {
// |   statement1;
// |   statement2;
// | }
// |
// | if (condition) {
// |     statement1;
// |     statement2;
// | } else {
// |     statement3;
// | }
// |
// | if (condition) {
// |     statement1;
// | } else {
// |     statement2;
// |     statement3;
// | }
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// | switch (variable) {
// |  case VAL2:
// |      statement1;
// |      break;
// |  default:
// |      statement2;
// | }

// | while (condition)
// |   statement1;
// |
// |  while (condition) {
// |   statement1;
// |   statement2;
// | }

// | for (n= 0; n < N; n += 1)
// |   statement1;
// |
// |  for (n= 0; n < N; n += 1) {
// |   statement1;
// |   statement2;
// | }


// ******************************************************************************
// ********************* C-code programming constraints ************************
// ****************v*****************************************************
// 1) For clarity, avoid using the following C constructs:
//     x++, y--, --x, and ++y.
//    Post increment or decrement are phrased as x+= 1 or y-= 1.
// 2) For correct ness, never suppress compilation warnings.
//    Following guidelines help eliminate spurious lint warnings
//

// ******************************************************************************
// ********************** Inhibiting warning messages **************************
// ******************************************************************************
// To minimize the number of false lint messages (on a Sun compiler),
// lint directives were used. It's assumed that a preprocessor can change
// these directives to work with other compilers. The lint directives
// (which are included as comments before the offending line) include
// CONSTCOND and FALLTHRU which are used as illustrated below:
//
//   | // CONSTCOND
//   | lineWhichUses(!CONSTANT);
//   |
//   | case X:
//   |   initialCode();
//   |   // FALLTHRU
//   | case Y:
//   |   commonSwitchCode();
//   |   break;

//
// Special DEFAULT_ONLY and DEFAULT_LAST macros are used within switch
// statements, when all of the enumerated values are intended to be used.
// The DEFAULT_ONLY is used when all enumerated values are used.
// It is typically followed by an ERROR() statement, or an ERROR1(arg)
// statement, to catch run-time errors. The "arg" expression inside of
// the ERROR1(arg) macro is used to inhibit gcc warnings, but ensuring
// that variables are set before being used, as illustrated below:
//
// | switch(value) {
// |  case X:
// |    add = 4;
// |    break;
// |  case Y:
// |    add = 5;
// |      break;
// |  DEFAULT_ONLY
// |      ERROR1(add = 4);
// |  }
// | sum += add;
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// The DEAFULT_LAST macro is used when some of the enumerated value are
// not used. Like DEFAULT_ONLY, this macro is either defaulted to a null
// statements (for compile-time checking of enumerated value usage) or
// to a "default:" statement, for run-time checking of enumerated values.
// An example is shown below"
//
// | switch(value) {
// | case X:
// |   ProcessX();
// |     break;
// | case Y:
// |     ProcessY();
// |   break;
// | case Z:
// |   DEFAULT_LAST
// |   ERROR();
// | }

// Depending on the option which is selected, null routines are sometimes
// included in the compilation, but never called. To eliminate lint/gcc
// warnings, the null routine includes a EXECUTE_ERROR2(a1,a2) macro
// which uses all of the calling arguments, as illustrated below:
//
//
// | #if DOP_XXXX_NONE
// | uInt2
// | StubRoutine(CommonParameters *a1, int a2)
// | {
// |     EXECUTE_ERROR2(a1, a2);
// |     return((uInt2)0);
// | }
// | #endif
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// Sample routines are included to illustrate how vendor-dependent routines
// may be constructed. In some cases, the sample routine does not use all
// of its arguments. For such routines, the IS_USED(a) macro is used to
// eliminate "argument not used" warnings, as illustrated below:
//
// | int
// | AddressMatchSample(MemParameters *memPtr, uInt2 this, uInt2 that)
// | {
// |     assert(memPtr != NULL);
// |     IS_USED(that);
// |     return(memPtr->common.nodeId == this);
// | }
//
// *******************************************************
// ************* indent argument values ******************
// *******************************************************
// The code is formatted using indent (GNU version 1.2) with the following
// flags set:
//
// -l132   maximum length of a line is 78 characters
// -c47   code comments start at line position 47
// -cd35  declaration comments start at line position 35
// -i2    indentation level is 2 spaces
// -ci1   continuation lines are indented 1 space
// -di1   put identifiers in first available position after type
// -nbad  no blank line is forced after a declaration
// -nbap  no blank line is forced after a procedure body
// -ncs   no space after a cast operator
// -npcs  no space after a procedure name and '('
// -nfc1  do not format comments in the first column
// -br    put '{' on line with if
// -sc    put '*' at left of comments
// -ce    put else on same line as '}'
// -ncdb  no comment delimiters on blank lines
// -nlp   do not line up continued lines at parentheses
//
// *******************************************************
// *************** compiler tolerance ******************
// *******************************************************
//
// The code is designed to support compilers that are "almost" ANSI compliant,
// by avoiding features that have been found to cause problems:
//
//  1) assert(). ANSI C defines this to behave like a function, returning the
//     void 0 value. Since many compilers use a different convention in their
//     assert.h file, this C code provides its own assert() definition
//     (unless the GNU compiler is being used, which gets this right).
//
//  2) External function declarations. When declaring a function,
//     either within the header or the routine itself, use the
//     typedef names, as illustrated below:
//
// | void
// | DoLocks(Quads2(*LockCompute) (Quads2, Quads2, Quads2), int sub,
// |   int size, Byte * valPtr, Byte * argPtr, Byte * oldPtr, Byte * memPtr)
// | {
// |    // Other DoLocks() code would be located here
// | }
//
//  3) When the function prototype is used to specify the format
//     within a data structure, then the typedef name should be avoided,
//     as illustrated below:
//
// | typedef struct _ChipInitInfo {
// |
// |   // Other structure entry definitions
// |
// |   void (*VdSetup) (struct _ChipInitInfo *);
// |
// | } ChipInitInfo;

// ****************************************************************************
// ************************* Compile-time warnings ****************************
// ****************************************************************************
// Testing for compile-time errors is performed using gcc, using the command
// string given below. Note that -O2 option is required to find the
// "uninitialized" warnings.
//
//         gcc -O2 -Wall -Wshadow -Wpointer-arith -Wcast-qual -Wcast-align
//            -Wmissing-prototypes -c ourCode.c
//
// Symbol length limitations:
// To test that only 31 significant characters are required for
// internal identifiers, gcc is used with the following options:
//
//         gcc -fsyntax-only -Wid-clash-31 ourCode.c
//
//
// ****************************************************************************
// ********************* Compiler-dependent dependencies **********************
// ****************************************************************************
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// The GNU C compiler defines assert() properly, most C compilers do not.
// This assert() definitions are provided for non GNU C compilers.

#include <stdio.h>
#ifdef __GNUC__
#include <assert.h>
#else
#ifdef NDEBUG
#define assert(ex)    ((void)0)
#else
#define assert(ex)    ((void)((ex) ? 0 : ((void)fprintf(stderr, \
 "Assertion failed: file \"%s\",line %d\n",__FILE__,__LINE__), \
 ((void)abort(), 0))))
#endif
#endif

// A null definition for gcc checks; all enum values must be used
#ifdef GCC_CHECK
#define DEFAULT_ONLY                          // NOTREACHED
#define DEFAULT_LAST
#else
#define DEFAULT_ONLY  default:
#define DEFAULT_LAST  default:
#endif

#ifdef PURE_ANSI
int rand(void);                   // ANSI standard random function
#define RAND16()    (uInt2)(((rand()>>4)&0xff)|((rand()<<4)&0xff00))
#define RAND32()    (uInt4)(RAND16()<<16 | RAND16())
#else
long random(void);                // Unix random() function
#define RAND16()    (uInt2)random()
#define RAND32()    (uInt4)random()
#endif

// This line should never be executed, print message and abort
#define ERROR() ((void)((void)fprintf(stderr, "ERROR() executed: file \"%s\",line
%d\n",__FILE__,__LINE__), (void)abort(), 0))

// ******************************************************************************
// ************************ User Modifiable Constants **************************
// ******************************************************************************

// This constant allows the user to test whether reserved fields are really ignored.
// If DEBUG is 1, use random-valued reserved fields, to validate they are properly ignored.
// If DEBUG is 0, use zero-valued reserved fields, to conform to the specification.
#define DEBUG 0                               // Tests ignored fields

// ******************************************************************************
// ************************ Other header definitions ***************************
// ******************************************************************************
// The IS_USED() macros eliminates lint/gcc warnings within uncalled stub routines
#define IS_USED1(a)  ((a)=(a))
#define IS_USED2(a,b)  (IS_USED1(a), IS_USED1(b))
#define IS_USED3(a,b,c)  (IS_USED2(a, b), IS_USED1(c))
#define IS_USED4(a,b,c,d)  (IS_USED2(a, b), IS_USED2(c, d))
#define IS_USED5(a,b,c,d,e)  (IS_USED4(a,b,c,d), IS_USED1(e))
#define IS_USED6(a,b,c,d,e,f)  (IS_USED4(a,b,c,d), IS_USED2(e,f))
#define IS_USED7(a,b,c,d,e,f,g)  (IS_USED4(a,b,c,d), IS_USED3(e,f,g))
#define IS_USED8(a,b,c,d,e,f,g,h)  (IS_USED4(a,b,c,d), IS_USED4(e,f,g,h))
#define IS_USED9(a,b,c,d,e,f,g,h,i)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED1(i))
#define IS_USED10(a,b,c,d,e,f,g,h,i,j)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED2(i,j))
#define IS_USED11(a,b,c,d,e,f,g,h,i,j,k)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED3(i,j,k))
#define IS_USED12(a,b,c,d,e,f,g,h,i,j,k,l)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED4(i,j,k,l))
#define IS_USED13(a,b,c,d,e,f,g,h,i,j,k,l,m)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED5(i,j,k,l,m))
#define IS_USED14(a,b,c,d,e,f,g,h,i,j,k,l,m,n)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED6(i,j,k,l,m,n))
#define IS_USED15(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o)  (IS_USED8(a,b,c,d,e,f,g,h), IS_USED7(i,j,k,l,m,n,o))

#define EXECUTE_ERROR1(a1) ( IS_USED1(a1), ERROR() )
#define EXECUTE_ERROR2(a1,a2) ( IS_USED2(a1,a2), ERROR() )
#define EXECUTE_ERROR3(a1,a2,a3) ( IS_USED3(a1,a2,a3), ERROR() )
#define EXECUTE_ERROR4(a1,a2,a3,a4) ( IS_USED4(a1,a2,a3,a4), ERROR() )
#define EXECUTE_ERROR5(a1,a2,a3,a4,a5) ( IS_USED5(a1,a2,a3,a4,a5), ERROR() )
#define EXECUTE_ERROR6(a1,a2,a3,a4,a5,a6) ( IS_USED6(a1,a2,a3,a4,a5,a6), ERROR() )
#define EXECUTE_ERROR7(a1,a2,a3,a4,a5,a6,a77) ( IS_USED7(a1,a2,a3,a4,a5,a6,a7), ERROR() )
#define EXECUTE_ERROR8(a1,a2,a3,a4,a5,a6,a7,a8) ( IS_USED8(a1,a2,a3,a4,a5,a6,a7,a8), ERROR() )
#define EXECUTE_ERROR9(a1,a2,a3,a4,a5,a6,a7,a8,a99) ( IS_USED9(a1,a2,a3,a4,a5,a6,a7,a8,a9), ERROR() )
#define EXECUTE_ERROR10(a1,a2,a3,a4,a5,a6,a7,a8,a9,a10) ( IS_USED10(a1,a2,a3,a4,a5,a6,a7,a8,a9,a10), ER-
ROR() )
#define EXECUTE_ERROR11(a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11) (
IS_USED11(a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11), ERROR() )

// RESERVED_SET() is used to keep lint quiet
#define RESERVED_SET()  ((int)(DEBUG ? RAND16() : 0))
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
typedef unsigned char uInt1;      // assuming 'char' is an 8-bit value
typedef unsigned short uInt2;     // assuming 'short' is a 16-bit value
typedef unsigned long uInt4;      // assuming 'long' is a 32-bit value
typedef unsigned long long uInt8; // assuming 'long long' is a 64-bit value
typedef signed char sInt1;        // assuming 'char' is an 8-bit value
typedef signed long sInt2;        // assuming 'short' is a 16-bit value
typedef signed long sInt4;        // assuming 'long' is a 32-bit value
typedef signed long long sInt8;   // assuming 'long long' is a 64-bit value


void PrintTable(int, int);
int ValidateCrc(int, uInt4 *, int);
uInt4 GenerateCrc(int, uInt4 *, int);
uInt4 CalculateCrc(int, uInt4 *, int);
uInt4 CrcBits(uInt4, uInt4, int);
uInt4 BitReverse(uInt4);
void Error(char *);

#define MSB32  ((unsigned)1<<31)
#define ONES32 0XFFFFFFFF
#define CRC_COMPUTE ((uInt4)0X04C11DB7)
#define CRC_RESULTS ((uInt4)0XC704DD7B)

int
main(int argc, char **argv)
{
    int i;
    int size = 32, reverse = 1, table = 1;
    int setRev = 0, setHow = 0;
    char *argPtr;

    // Command line specifies number of bits computed in parallel
    for (i = 1; i < argc; i += 1) {
        argPtr = argv[i];
        if (*argPtr != '-')
            Error("Illegal argument, use: -n -r -tdd -c");
        argPtr += 1;
        switch (*argPtr) {
        case 'n':
        case 'r':
            if (setRev)
                Error("Mutually exclusive options: -n -r");
            reverse = (*argPtr == 'r');
            setRev = 1;
            break;
        case 't':
            if (setHow)
                Error("Mutually exclusive options: -tdd -c");
            size = atoi(argPtr + 1);
            if (size != 1 && size != 2 && size != 4 && size != 8 && size != 16 && size != 32)
                Error("Incorrect width; -t1 -t2 -t4 -t8 -t16 or -t32\n");
            table = 1;
            break;

        case 'c':
            if (setHow)
                Error("Mutually exclusive options: -wdd -c");
            table = 0;
            setHow = 1;
            break;

        default:
            Error("Arguments: -n -r -t[1,2,4,8,16,32] -c\n");
            break;
        }
    }
    assert(table);
    PrintTable(reverse, size);
    return (0);
}

void
Error(char *string)
{
    printf(string);
    exit(1);
}
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
char keys[] = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'j', 'k', 'm', 'n', 'p', 'r', 's', 't' };
void
PrintTable(int reverse, int size)
{
    uInt4 last, next, select, mask, sum;
    int i, j, numb;
    for (i = 0; i < 32; i += 1) {
        // Calculate contributing-input values
        select = 1 << (31 - i);
        mask = reverse ? BitReverse(select) : select;
        printf("c%02d= ", i);
        for (j = sum = 0; j < 32; j += 1) {
            last = 1 << (31 - j);
            next = CrcBits(last, (uInt4) 0, size);
            if (next & mask)
                sum |= last;
        }
        for (j = 0; j < 32; j += 1) {
            select = 1 << (31 - j);
            mask = reverse ? BitReverse(select) : select;
            numb = j < 16 ? keys[j] : keys[j - 16] + 'A' - 'a';
            if ((sum & mask) != 0) {
                sum &= ~mask;
                printf("%c", numb);
                if (j != 31)
                    printf(sum != 0 ? "^" : " ");
            } else {
                printf(j != 31 ? "  " : " ");
            }
        }
        printf(";\n");
    }
}

// Generate the CRC in packet containing "sizeInQuads" uInt4 data values
uInt4
GenerateCrc(int reverse, uInt4 * inputs, int sizeInQuads)
{
    uInt4 crcSum;

    assert(sizeInQuads >= 1);                // Packet size including CRC

    crcSum = CalculateCrc(reverse, inputs, sizeInQuads - 1);
    // compute CRC on just the data
    return (~crcSum);
}

// Validate the CRC for a packet containing "size" uInt4 data values
int
ValidateCrc(int reverse, uInt4 * inputs, int sizeInQuads)
{
    uInt4 crcSum, check;

    assert(sizeInQuads >= 1);                // Packet size including CRC

    crcSum = CalculateCrc(reverse, inputs, sizeInQuads);
    // compute CRC on the data and the received CRC
    check = reverse ? BitReverse(crcSum) : crcSum;
    return (check != CRC_RESULTS);
}

// The GenerateCrc() function points to protected values,
// it checks these values and return a final 32 - bit result
uInt4
CalculateCrc(int reverse, uInt4 * inputs, int sizeInQuads)
{
    uInt4 inQuad, crcSum, sum;
    int i;

    // The crcSum value is initialized to all ones
    crcSum = (uInt4) 0XFFFFFFFF;

    // Process each of the uInt4s covered by the CRC value
    for (i = 0; i < sizeInQuads; i += 1) {
        inQuad = reverse ? inputs[i] : BitReverse(inputs[i]);
        crcSum = CrcBits(crcSum, inQuad, 32);
    }
    sum = reverse ? BitReverse(crcSum) : crcSum;
    return (sum);
}

uInt4
CrcBits(uInt4 last, uInt4 input, int size)
{
    uInt4 crcSum, newMask;
    int i, oldBit, newBit, sumBit;
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    // Process each of the bits within the input uInt4 value
    crcSum = last;
    for (newMask = MSB32, i = 0; i < size; newMask >>= 1, i += 1) {
        newBit = ((input & newMask) != 0);    // The next input bit
        oldBit = ((crcSum & MSB32) != 0);     // and MSB of crcSum
        sumBit = oldBit ^ newBit;             // are EXOR'd together

        // Shift the old crcSum left and exclusive - OR the new newBit values
        crcSum = ((crcSum << 1) & ONES32) ^ (sumBit ? CRC_COMPUTE : 0);
    }
    return (crcSum);
}

// Reverse the order of bits within bytes
uInt4
BitReverse(uInt4 old)
{
    uInt4 new, oldMask, newMask;
    int i, j;

    for (i = new = 0; i < 4; i += 1) {
        for (j = 0; j < 8; j += 1) {
            oldMask = 1 << (8 * i + 7 - j);
            newMask = 1 << (8 * i + j);
            new |= (old & oldMask) ? newMask : 0;
        }
    }
    return (new);
}

#define DARWIN 1
#ifdef DARWIN

#define FALSE 0
#define TRUE  1

#define LP_MU_ 1                            // Added by the editor to suppress gcc errors
#define HOPS 1                              // Added by the editor to suppress gcc errors
#define HUH 1                               // Added by the editor to suppress gcc errors

typedef uInt1 Boolean;
typedef struct
{
    uInt4 loTbDepth;              // Low priority transit buffer depth
    uInt4 addRate;                // Count of low priority (LP) and excess medium priority eMP octets
transmitted by client
   uInt4 addRateCongestion;     // Count of LP and eMP octets transmitted by client and destined beyond
the congestion point
    uInt4 lpAddRate;              // addRate run through a low pass filter
    uInt4 nlpAddRate;             // lpAddRate/WEIGHT

    uInt4 lpAddRateContestion;    // addRateCongestion run through a low pass filter
    uInt4 nlpAddRateContestion;   // lpAddRateContestion / WEIGHT
    Boolean addRateOk;            // flag indicating that client is allowed to transmit
   uInt4 allowRateCongestion;   // The fair amount each node is allowed to transmit beyond the congestion
point
    uInt4 fwdRate;                // Count of LP+eMP octets forwarded from the LP transit buffer
    uInt4 lpFwdRate;              // fwdRate run through low pass filter
    uInt4 rate_iMP;               // Count of iMP octets forwarded and added
    uInt4 lp_rate_iMP;            // Rate_iMP run through low pass filter
    uInt4 lpAllowCongestion;      // allowRateCongestion run through low pass filter
    Boolean congested;            // Station congestion: transit buffer occupancy, link utilization or
HoL timer expration
    uInt4 accessDelay;

    uInt2 rcvdAdvertisedRate;     // The fair rate(# of bytes in a decay interval)received from the
downstream neighbor
    uInt2 advertisedRate;        // The fair rate (# of bytes in a decay interval) passed along to the
upstream neighbor
    sInt2 tokenOctets;            // Bytes in RPR-fa dynamic shaper/policer for eMP&LP traffic beyond
congestion point
    sInt2 tokenOctetsLow;         // Bytes in RPR-fa shaper/policer for low priority
    sInt2 tokenOctetsMed;         // Bytes in RPR-fa shaper/policer for medium (iMP+eMP) priority
    sInt2 tokenOctetsHigh;        // Bytes in RPR-fa shaper/policer for high priority
    uInt2 numberActiveSources;    // Number of active sources
    Boolean addPacketEnqueuedHigh;
    Boolean addPacketEnqueued_low;
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    int fairnessOn;                 // Added by the editor to eliminate gcc errors
    int mode;                       // Added by the editor to eliminate gcc errors
    int initial;                    // Added by the editor to eliminate gcc errors
    int nextState;                  // Added by the editor to eliminate gcc errors
    int currentState;               // Added by the editor to eliminate gcc errors
    int reservedRate;               // Added by the editor to eliminate gcc errors
    int aboveHighThreshold;         // Added by the editor to eliminate gcc errors
    int aboveLowThreshold;          // Added by the editor to eliminate gcc errors
    int belowHighThreshold;         // Added by the editor to eliminate gcc errors
    int belowLowThreshold;          // Added by the editor to eliminate gcc errors
    int hysteresis;                 // Added by the editor to eliminate gcc errors
    int loAllow;                    // Added by the editor to eliminate gcc errors
    int lpAllow;                    // Added by the editor to eliminate gcc errors
    int nodeState;                  // Added by the editor to eliminate gcc errors
    int wrapped;                    // Added by the editor to eliminate gcc errors
    int addPacketEnqueued;          // Added by the editor to eliminate gcc errors
    int my_SA;                      // Added by the editor to eliminate gcc errors
    int my_RI;                      // Added by the editor to eliminate gcc errors
}
Station;

typedef struct
{
    uInt1 SA[6];
    Boolean RI;
    uInt2 rate;
}
FairnessPacket;
FairnessPacket fairnessPacket;    //received fairness packet

typedef enum
{
    FAIRNESS_OFF,
    FAIRNESS_ON,
    RAMP_UP,
    RAMP_DOWN,
    DOWNSTREAM_CONGESTED
}
States;

States currentState, nextState;
Boolean initial;


// Constants (Tunable, provisioned LME objects):
uInt2 WEIGHT;                     // Configured weight for this station an integer between 1 and 63
uInt2 w[HOPS];                    // weight vector for all stations on the ring
uInt2 BUCKET_SIZE;                // provisioned RPR-fa dynamic shaper leaky bucket size in octets
uInt4 MAX_ALLOWANCE;              // Configured value for max allowed rate for this node
uInt4 DECAY_INTERVAL;             //    8,000 octet times @ OC-12,
                                  //   32,000 octet times @ OC-48,
                                  //  128,000 octet times @ OC-192
uInt4 ADVERTISEMENT_INTERVAL;     // Between 1 MTU time and DECAY_INTERVAL

uInt2 AGECOEFF = 4;               // Aging coeff for addRate and fwdRate
uInt2 LP_FWD = 64;                // Low pass filter for fwdRate
uInt2 LP_MU = 512;                // Low pass filter for my fair rate
uInt2 LP_MUHistory = 511;         // Low pass filter for history rate
uInt2 LP_ALLOW = 64;              // LP filter for allow rate auto increment
uInt2 LP_ALLOW_COEF = 128;        // low-pass filter for lpAllow
uInt2 NULL_RATE = 0xFFFF;         // All 1's in rcvdAdvertisedRate field.

uInt2 TB_LO_THRESHOLD;            // TB depth at which no more LP client traffic can be sent
uInt4 MAX_LRATE;                  // AGECOEFF * DECAY_INTERVAL =
                                  //  512,000 for OC-192
                                  //  128,000 for OC-48
                                  //   32,000 for OC-12

uInt4 AD_THRESHOLD;               // Default value 1ms

//Each station should have knowledge of reserved rate
uInt4 reservedRate;               // High priority reserved rate (# of bytes in a decay interval
Boolean mode;                     // Advertising mode: 1 conservative, 0 (default) aggressive

void UpdatedEveryClockCycle(Station *);
void UpdatedWhenFairnessPacketReceived(Station *, FairnessPacket *);
void CalculatedEveryDecayInterval(Station *);
void CalculatedEveryAdvertisementInterval(Station *);
void Advertise(Station *);
uInt2 F1(Station *, int);

// THESE ARE UPDATED EVERY CLOCK CYCLE:
// addRate is increment by 1 for every LP/eMP octet that is
// Transmitted by the client (does not include data
// Transmitted from the Transit Buffer).

// addRateCongestion is increment by 1 for every LP/eMP octet that is
// Transmitted by the client (does not include data
// Transmitted from the Transit Buffer) and destined beyond congestion
// point (i.e., TTL > TTL_to_congestion).
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// fwdRate is increment by 1 for every LP/eMP octet that exits the LP Transit Buffer

void
UpdatedEveryClockCycle(Station * sPtr)
{
    // tokenOctets is decremented by 1 for every LP/eMP octet that is transmitted by the client.
    // A packet is sent to completion, why is the below check here?
    if ((sPtr->addRateCongestion < sPtr->allowRateCongestion) &&
     (sPtr->fwdRate + sPtr->addRate + sPtr->rate_iMP < MAX_LRATE - sPtr->reservedRate) &&
     (sPtr->tokenOctets > 0) && !((sPtr->loTbDepth > 0) && (WEIGHT * sPtr->fwdRate < sPtr->addRate)) &&
     (sPtr->addRate < MAX_ALLOWANCE))
        sPtr->addRateOk = TRUE;               // true means OK to send client packets

    // Access delay is:
    //   started when an SOP arrives
    //   reset when packet is transmitted
    //   not increased when a packet is enqueued

    if (sPtr->addPacketEnqueued)
        sPtr->accessDelay += 1;               // Any packet being added except high priority
    if (sPtr->accessDelay >= AD_THRESHOLD)    // Access delay expires
        sPtr->congested = TRUE;
}

// UPDATED WHEN FAIRNESS_PKT IS RECEIVED:
void
UpdatedWhenFairnessPacketReceived(Station * sPtr, FairnessPacket * fPtr)
{
    if ((fPtr->SA[0] == sPtr->my_SA) &&       // Change by editor to suppress GCC errors
     ((sPtr->nodeState == sPtr->wrapped || fPtr->RI == sPtr->my_RI))) {
        sPtr->rcvdAdvertisedRate = NULL_RATE;
    } else {
        sPtr->rcvdAdvertisedRate = fPtr->rate;
    }
}

//steering case:
//  do not forward on,
//  for wrapping case pass set
//treat single link as segment failure.
// THE FOLLOWING IS CALCULATED EVERY DECAY_INTERVAL:
void
CalculatedEveryDecayInterval(Station * sPtr)
{
    if ((sPtr->loTbDepth > TB_LO_THRESHOLD / 2) ||      // LPTB crosses threshold
     ((sPtr->addRate + sPtr->fwdRate) > (MAX_LRATE - sPtr->reservedRate)) ||    // Link utilization cross
high threshold
     (sPtr->accessDelay >= AD_THRESHOLD)) {   // access delay
        sPtr->congested = TRUE;
    } else {
        sPtr->congested = FALSE;
    }

    sPtr->lpAddRate = (sPtr->lpAddRate / (LP_MU_ - 1 / LP_MU) + (sPtr->addRate / LP_MU));       // More
generic weights than LP_MU-1
    sPtr->nlpAddRate = sPtr->lpAddRate / WEIGHT;        // modified weight function
    //addRate is decremented by min(allow_rate/AGECOEFF, addRate/AGECOEFF)

    sPtr->lpFwdRate = ((LP_FWD - 1) * sPtr->lpFwdRate + sPtr->fwdRate) / LP_FWD;
    // fwdRate is decremented by fwdRate/AGECOEFF
    //(Note: lp values calculated prior to decrement of non-lp values).

    // tokenOctets is incremented at a rate which equals to
    // lpAllow /(AGECOEFF * DECAY_INTERVAL)
    sPtr->lpAllow = ((LP_ALLOW_COEF - 1) * sPtr->lpAllow + sPtr->allowRateCongestion) / LP_ALLOW_COEF;

    if (sPtr->rcvdAdvertisedRate != NULL_RATE)
        sPtr->allowRateCongestion = (sPtr->rcvdAdvertisedRate * WEIGHT);
    else
        sPtr->allowRateCongestion += (MAX_LRATE - sPtr->reservedRate - sPtr->allowRateCongestion) /
LP_ALLOW;

    if (sPtr->fairnessOn && sPtr->mode == 1)
        sPtr->allowRateCongestion = sPtr->advertisedRate * WEIGHT;

    // fwd_rate_iMP is increment by 1 for every iMP octet that
    // exits the LP Transit Buffer
}

// THE FOLLOWING IS CALCULATED EVERY ADVERTISEMENT_INTERVAL:
void
CalculatedEveryAdvertisementInterval(Station * sPtr)
{
    Advertise(sPtr);
    if ((sPtr->addRate + sPtr->fwdRate) < (MAX_LRATE - sPtr->reservedRate - sPtr->hysteresis)) {
        sPtr->belowLowThreshold = TRUE;
        sPtr->aboveLowThreshold = FALSE;
    } else {
        sPtr->belowLowThreshold = FALSE;
        sPtr->aboveLowThreshold = TRUE;
    }
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    if ((sPtr->addRate + sPtr->fwdRate) >= (MAX_LRATE - reservedRate)) {
        sPtr->aboveHighThreshold = TRUE;
        sPtr->belowHighThreshold = FALSE;
    } else {
        sPtr->belowHighThreshold = TRUE;
        sPtr->aboveHighThreshold = FALSE;
    }
}

void                            // rate advertisement machine
Advertise(Station * sPtr)
{
    int I = 1, J = 1;           // Added by editor to eliminate gcc errors

    sPtr->nextState = sPtr->currentState;     // No change
    switch (sPtr->currentState) {
    case FAIRNESS_OFF:
        if (sPtr->rcvdAdvertisedRate != NULL_RATE)
            //downstream_congested
            sPtr->nextState = DOWNSTREAM_CONGESTED;     //next state is downstream congested

        if (sPtr->congested)
            sPtr->nextState = FAIRNESS_ON;    // next state is congested
        sPtr->advertisedRate = NULL_RATE;     // advertises is still maximum bw

        // allowRateCongestion += (MAX_LRATE-allowRateCongestion)/LP_ALLOW;
        sPtr->initial = TRUE;

    case FAIRNESS_ON:
        if (sPtr->initial) {
            sPtr->advertisedRate = sPtr->mode ? sPtr->nlpAddRate : (MAX_LRATE - sPtr->reservedRate) /
F1(sPtr, HUH);
            // calculated initial rate:fair rate per unit weight
            sPtr->initial = FALSE;
        }
        // advertise min(low pass add rate, received advertise rate)
        sPtr->advertisedRate = (sPtr->nlpAddRate < sPtr->rcvdAdvertisedRate) ? sPtr->nlpAddRate : sPtr-
>rcvdAdvertisedRate;

        if (sPtr->rcvdAdvertisedRate != NULL_RATE)
            sPtr->nextState = DOWNSTREAM_CONGESTED;     // next state is downstream congested
        if (sPtr->belowLowThreshold)
            sPtr->nextState = RAMP_UP;
        //allowRateCongestion


    case RAMP_UP:
        sPtr->advertisedRate = sPtr->mode ? NULL_RATE : sPtr->lpAddRate + sPtr->lpAddRate * I / J;
        if (sPtr->advertisedRate > (MAX_LRATE - sPtr->reservedRate))
            sPtr->advertisedRate = NULL_RATE;
        if (sPtr->aboveHighThreshold)
            sPtr->nextState = RAMP_DOWN;
        if (sPtr->aboveLowThreshold)
            sPtr->nextState = FAIRNESS_ON;

        if (sPtr->advertisedRate == NULL_RATE)
            sPtr->nextState = FAIRNESS_OFF;
        //allowRateCongestion

    case RAMP_DOWN:
       sPtr->advertisedRate = sPtr->mode ? sPtr->lpAddRate : sPtr->lpAddRate - sPtr->lpAddRate * I / J;
        if (sPtr->belowHighThreshold)
            sPtr->nextState = FAIRNESS_ON;

        if (sPtr->belowLowThreshold)
            sPtr->nextState = RAMP_UP;
        //allowRateCongestion

    case DOWNSTREAM_CONGESTED:
        if (sPtr->lpFwdRate > (sPtr->allowRateCongestion / WEIGHT))
            sPtr->advertisedRate = sPtr->rcvdAdvertisedRate;
        else
            sPtr->advertisedRate = NULL_RATE;
        sPtr->allowRateCongestion = (sPtr->rcvdAdvertisedRate * WEIGHT);
        // WEIGHT is local weight * received control message

        if (sPtr->rcvdAdvertisedRate == NULL_RATE)
            sPtr->nextState = FAIRNESS_OFF;
        sPtr->currentState = sPtr->nextState;
    }
}

uInt2
F1(Station * sPtr, int weightStation)
{
    int i;
    uInt2 weightTotalActive = 0;  // Local variable weightTotalActive
    // input weightStation;
    // mode 1: weight == stationWeight;
    //
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
    for (i = 1; sPtr->numberActiveSources; i += 1)
        weightTotalActive += w[i];            // Add all active weight global variable
    weightTotalActive += weightStation;       // Add local station weight
    return (weightTotalActive);
}

#endif // DARWIN


#if DVJCODE
// ***************************************************************************
// ************************* DVJ C-code illustrations  **********************
// ***************************************************************************
// This code is provided to illustrate possibilities when refining Darwin code

// TBD: 1) class-A0/class-A1 client transparency
//      2) discarding of MAC-transmission excesses
//      3) Coarse accounting granularity
#define HOPS 64                               // Worst-case number of hops
#define BYTES_PER_TICK 16                     // A coarse-grained accounting value
#define MTU (1536/BYTES_PER_TICK)             // An estimated maximum-frame size
#define DTU (MTU+4*MTU)                       // MTU plus MAC-to-client turn-around delay
#define MINIMUM(x,y) (((x)-(y))>0 ? (x):(y))  // Modulo-arithmetic-compatible maximum
#define MAXIMUM(x,y) (((x)-(y))<0 ? (x):(y))  // Modulo-arithmetic-compatible minimum
#define ONE (((uInt8)1)<<32)                  // Scalar for {integer:32, fraction:32} values

typedef struct
{
    //
    // Hop-count insensitive information
    sInt8 creditD;                // Sustaining class-A idles
    sInt8 creditD1;               // Sustaining class-A1 idles
    sInt8 creditBC;               // Ensuring transitBC progress
    sInt8 creditC;                // Limiting class-A transmit bandwidth
    uInt4 ratingA0;               // Ringlet rating for class-A0 traffic
    uInt4 ratingA1;               // Ringlet rating for class-A1 traffic
    uInt4 ratingC;                // Maximum class-C transmit bandwidth
    uInt4 weightC;                // Weighting for class-C traffic
    //
    // Hop-count sensitive information
    sInt8 creditA0[HOPS];         // Hop-count based class-A0 transmission credits
    sInt8 creditA1[HOPS];         // Hop-count based class-A1 transmission credits
    sInt8 creditB[HOPS];          // Hop-count based class-C transmission credits
    sInt8 countC[HOPS];           // Hop-count based class-C transmission counts
    uInt4 rateA0[HOPS];           // Hop-count based class-A0 transmission rates
    uInt4 rateA1[HOPS];           // Hop-count based class-A1 transmission rates
    uInt4 rateB[HOPS];            // Hop-count based class-B transmission rates
    //
    // Information derived from downstream-sourced flow-control frame
    uInt4 sensedDepth;            // Sense indication of downstream transitBC depth
    uInt4 limitC[HOPS];           // Other stations published countC run-rates
    uInt4 lowerC[HOPS];           // Other stations previously published countC
    //
    // Information asserted across the MAC-to-client interface, for flow-control purposes
    uInt1 rangeA0;                // Client indication of class-A0 permission range
    uInt1 rangeA1;                // Client indication of class-A1 permission range
    uInt1 rangeB;                 // Client indication of class-B permission range
    uInt1 rangeC;                 // Client indication of class-C permission range
    //
    // Performance parameters derived during the discovery process
    uInt1 hops;                   // Number of unwrapped hops
    uInt4 loopDelay;              // Ringlet-circulation time, unwrapped and unloaded
}
Station;

void RangeAdjustments(Station *);
void CreditAdjustments(Station *, int, int, int, int, int, int, int, int, int, int, int);
void CreditLimits(int, uInt8 *, uInt4 *, uInt4, uInt4, sInt4, sInt4);
int CreditUpdate(uInt8 *, uInt4, uInt4, uInt4, sInt4, sInt4);
int DepthsToAssist(uInt4, uInt4);
uInt4 DepthToRateBC(uInt4);
uInt4 WeightToScaleC(uInt4);

uInt4 DepthTransitBC(Station *);  // Implementation-specific transitBC depth measurement
int DepthStage(Station *);        // Implementation-specific stage-buffer nearly full
int Staged(Station *);            // One or more stage-buffer entries are ready to send
int Queued(Station *);            // One or more transitBC FIFO entries are ready to send
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// Called to update credits
void
CreditAdjustments(Station * context, int waitT, // Time duration since last called
 int moveA0,                      // Class-A0 client-to-MAC transfer size
 int moveA1,                      // Class-A1 client-to-MAC transfer size
 int moveB,                       // Class-B client-to-MAC transfer size
 int moveC,                       // Class-C client-to-MAC transfer size
 int sendBC,                      // Lower-class transitBC retransmission size
 int sendA0,                      // Class-A0 output-transmission size
 int sendA1,                      // Class-A1 output-transmission size
 int sendB,                       // Class-B output-transmission size
 int sendC,                       // Class-C output-transmission size
 int sendI)                       // Udle-frame equivalent output-transmission size
{
    uInt8 least;
    int assist, n, hops = context->hops;
    uInt4 depthTransitBC, rateBC, rateC;
    uInt4 hiSideA0, hiSideA1, hiSideB, hiSideD, hiSideD1, hiSideBC, loSideBC, scaleC, waitSizeC;

    depthTransitBC = DepthTransitBC(context);
    hiSideA0 = ((MTU * context->ratingA0) / ONE) + DTU;
    CreditLimits(HOPS, context->creditA0, context->rateA0, moveA0, waitT, hiSideA0, -DTU);
    //
    hiSideA1 = ((MTU * context->ratingA1) / ONE) + DTU;
    CreditLimits(HOPS, context->creditA1, context->rateA1, moveA1, waitT, hiSideA1, -DTU);
    //
    hiSideB = 2 * (context->loopDelay + hops * MTU) + DTU;
    CreditLimits(HOPS, context->creditB, context->rateB, moveB, waitT, hiSideB, -DTU);

    scaleC = WeightToScaleC(context->weightC);
    waitSizeC = depthTransitBC >= (ONE / 4) ? moveC : waitT;
    for (n = 0; n < context->hops; n = n + 1)
        CreditUpdate(context->countC + n, scaleC, 0, waitSizeC, context->limitC[n] + DTU, context->low-
erC[n]);
    //
    hiSideD = DepthsToAssist(depthTransitBC, context->sensedDepth) != 0 ? DTU : 0;
    CreditUpdate(&(context->creditD), context->ratingA0 + context->ratingA1, sendA0 + sendA1 + sendI,
sendB + sendC, hiSideD, -DTU);
    //
    hiSideD1 = ((2 * hops * context->ratingA1) / ONE) + DTU;
    CreditUpdate(&(context->creditD1), context->ratingA1, sendA1 + sendI, sendA0 + sendB + sendC,
hiSideD1, -DTU);

    rateBC = DepthToRateBC(depthTransitBC);
    hiSideBC = Staged(context) ? DTU : 0;
    loSideBC = Queued(context) ? DTU : 0;
    CreditUpdate(&(context->creditBC), rateBC, moveB + moveC, sendBC, hiSideBC, -loSideBC);
    //
    CreditUpdate(&(context->creditC), context->ratingC, moveC, waitT, DTU, -DTU);
}

// Called to update range values
void
RangeAdjustments(Station * context)
{
    int n, stopA, stopB, stopC;
    //
    stopA = DepthStage(context) > (ONE / 2);
    stopB = stopA;
    stopB |= (context->creditD > 0);
    stopB |= (context->creditD1 > 0);
    stopB |= (context->creditBC < 0);
    stopC = stopB;
    stopC |= (context->creditC > 0);
    //
    for (n = 0; stopA == 0 && n < HOPS && context->creditA0[n] >= 0; n += 1);
    context->rangeA0 = n;
    //
    for (n = 0; stopA == 0 && n < HOPS && context->creditA1[n] >= 0; n += 1);
    context->rangeA1 = n;
    //
    for (n = 0; stopB == 0 && n < HOPS && context->creditB[n] >= 0; n += 1);
    context->rangeB = n;
    //
    for (n = 0; stopC == 0 && n < HOPS && (context->limitC[n] - (context->countC[n] / ONE)) > 0; n += 1);
    context->rangeC = n;
    return;
}
```


Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

```
// Called to update CreditUpdate() over multiple hop counts
// Excessive longer-hop credits are also discarded
void
CreditLimits(int count, uInt8 * creditPtr, uInt4 * sendRate, uInt4 sendSize, uInt4 waitSize, sInt4 hiSide,
sInt4 loSide)
{
    uInt8 least;
    int i;
    //
    CreditUpdate(creditPtr, sendRate[0], sendSize, waitSize, hiSide, loSide);
    least = creditPtr[0];
    for (i = 1; i < count; i = i + 1) {
        CreditUpdate(creditPtr + i, sendRate[i], sendSize, waitSize, hiSide, loSide);
        least = MINIMUM(least, creditPtr[0]);
        creditPtr[i] = MINIMUM(creditPtr[i], least + DTU);
    }
}

// Arguments for leaky-bucket adjustments:
//   creditPtr - pointer to credit value
//   sendRate  - transmission rate limitation
//   sendSize  - size of rate-limited frame
//   waitSize  - size of other frames or idles
//   hiSide    - maximum credit value limit
//   loSide    - minimum credit value limit
// Return a value of 1 when low threshold is reached
CreditUpdate(uInt8 * creditPtr, uInt4 sendRate, uInt4 sendSize, uInt4 waitSize, sInt4 hiSide, sInt4 lo-
Side)
{
    uInt8 credits, hiLevel, loLevel;
    //
    credits = creditPtr[0];
    hiLevel = hiSide * ONE;
    loLevel = loSide * ONE;
    credits = credits + (waitSize * sendRate) - (sendSize * ONE);
    credits = MINIMUM(credits, hiLevel);
    credits = MAXIMUM(credits, loLevel);
    return (credits != loLevel);
}

uInt4
DepthToRateBC(uInt4 depth)
{
    uInt4 rate;

    rate = ONE - (2 * depth);
    rate = MAXIMUM(rate, (ONE * 7) / 8);
    rate = MINIMUM(rate, 0);
    return (rate);
}

int
DepthsToAssist(uInt4 thisDepth, uInt4 thatDepth)
{
    if (thisDepth <= (ONE / 4))
        return (0);
    if (thisDepth < (ONE / 2))
        return (thisDepth < (2 * thatDepth - ONE / 2));
    return (thisDepth < thatDepth);
}

uInt4
WeightToScaleC(uInt4 weight)
{
    if (weight >= (ONE / 2))
        return ((3 * ONE) / 16 - weight / 8);
    if (weight >= (ONE / 4))
        return ((3 * ONE) / 8 - weight / 2);
    if (weight >= (ONE / 8))
        return ((3 * ONE) / 4 - 2 * weight);
    return (ONE - 4 * weight);
}

#endif
```

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Annex I

(informative)

Implementation Guidelines

Editors' Notes: *To be removed prior to final publication.*

References:
None.

Definitions:
None.

Abbreviations:
None.

Revision History:
Draft 0.1, February 2002Initial draft document for RPR WG review.

Editors' Notes: *To be removed prior to final publication.*

The wording and phrasing of this section are awkward, and should be fixed.

I.1 MAC client behavior

The behavior of the MAC client is presented for descriptive purposes. This section does not impose any behavior for the MAC client. The RPR standard defines a set of primitives at the MSAP interface. The number of queues and the queue managers at the MAC client are a matter of choice.

The simplest MAC client can have one queue for each traffic class. The STOP_LO, STOP_MED, STOP_HI signals will indicate the traffic class that cannot be sent. If the MAC client sends a packet for a stopped traffic class, the MAC policing functionality will not allow any more packets to be sent until that traffic class is allowed. However, the MAC client can decide to send a medium priority packet as an excess bandwidth packet, in which case the RPR MAC will treat that packet as a low priority packet and the status of STOP_LO signal will be important. This means that a MAC client is allowed to send a medium priority packet even when there is STOP_MED signal, provided that STOP_LO is not asserted. When a medium priority packet must be treated as low priority, the RPR MAC will mark the IOP bit in the header as out-of-profile, and the packet will consume rate shaping and fairness credits of the low priority traffic class in the RPR MAC. It is possible to starve low priority traffic by sending excess medium traffic in place of low priority traffic. The MAC client should choose, how much excess medium priority traffic can be scheduled, so as to either starve or not starve low priority traffic.

An example client could behave as indicated in Table D1:

Optionally, the MAC client may implement a more sophisticated queuing scheme to avoid head-of-line blocking and to utilize more bandwidth. This can be accomplished through the use of network congestion information transmitted by the nodes on the network and the collected topology information. For this purpose The MAC client can implement virtual output queues for each destination on the ring for low and/or medium priority.

The MAC client is allowed to send a packet from a virtual output queue for low priority or excess medium priority queues if it can satisfy the necessary condition, which is, for each congestion point before the

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.

Table D1—MAC client interface

Stop_L	Stop_M	Stop_H	Which queue to select
0	0	0	If there is a packet in high class traffic queue, schedule high class. If there is a packet in medium class traffic queue, schedule medium class. If there is a packet in low class traffic queue, schedule low class
0	0	1	If there is a packet in medium class traffic queue, schedule medium class. If there is a packet in low class traffic queue, schedule low class.
0	1	0	If there is a packet in high class traffic queue, schedule high class If there is a packet in medium class traffic queue, schedule medium class(will be treated as low priority) If there is a packet in low class traffic queue, schedule low class
0	1	1	If there is a packet in medium class traffic queue, schedule medium class(will be treated as low priority) If there is a packet in low class traffic queue, schedule low class
1	0	0	If there is a packet in high class traffic queue, schedule high class If there is a packet in medium class traffic queue, schedule medium class
1	0	1	If there is a packet in medium class traffic queue, schedule medium class
1	1	0	If there is a packet in high class traffic queue, schedule high class
1	1	1	Stop scheduling any more packets

selected destination, the total usage beyond the congestion point should be less than the congested nodes fairness value.

At any time there can be more than one virtual output queue that will satisfy the condition, in this case a round-robin approach can be chosen to simplify the solution. However, a better approach will be using deficit-round-robin, which will avoid possible unfairness among virtual output queues.

The calculations of queue add rates and allowed rates beyond a congested point are also important factors to increase utilization and obtain a stable behavior. An acceptable approach is to choose a similar algorithm to update and increment these values as implemented in the RPR MAC client for "allowed_rate" and "add_rate" for each virtual output queue. In addition one should low pass filter the value of per queue add rates to smooth out instantaneous variations. Once the MAC client receives a fairness message from the RPR MAC about a congested node, it will update the allowed_rate of that destination, which represents the total allowed_rate beyond that node. That value will then be increment periodically as long as the MAC client does not receive another fairness message for that node. In essence, the MAC client implements a copy of the MAC fairness algorithm for each destination.

Depending on the client's behavior, the assertion of STOP signals will vary in the RPR MAC. For virtual destination queuing, ideally STOP signals will never be asserted other than rate shaping purposes. If a client malfunctions, the MAC policing functions will prevent the client from abusing the ring.

Copyright © 2002 IEEE. All rights reserved.
This is not an approved IEEE Standard.