

Draft Standard for Local and metropolitan area networks—

6 Timing and Synchronization for 7 Time-Sensitive Applications

8 Unapproved draft, prepared by the
9 Maintenance Task Group of IEEE 802.1

10 Sponsored by the

11 LAN/MAN Standards Committee
12 of the
13 IEEE Computer Society

14 All participants in IEEE standards development have responsibilities under the IEEE patent policy and should
15 familiarize themselves with that policy, see <http://standards.ieee.org/about/sasb/patcom/materials.html>

16 As part of our IEEE 802® process, the text of the PAR (Project Authorization Request) and CSD (Criteria for
17 Standards Development) is reviewed regularly to ensure their continued validity. A vote of "Approve" on this
18 draft is also an affirmation that the PAR is still valid. It is included in these cover pages.

19 The text proper of this draft begins with the title page (1). The cover pages (a), (b), (c) etc. are for 802.1 WG
20 information, and will be removed prior to Sponsor Ballot.

Important Notice

This document is an unapproved draft of a proposed IEEE Standard. IEEE hereby grants the named IEEE SA Working Group or Standards Committee Chair permission to distribute this document to participants in the receiving IEEE SA Working Group or Standards Committee, for purposes of review for IEEE standardization activities. No further use, reproduction, or distribution of this document is permitted without the express written permission of IEEE Standards Association (IEEE SA). Prior to any review or use of this draft standard, in part or in whole, by another standards development organization, permission must first be obtained from IEEE SA (stds-copyright@ieee.org). This page is included as the cover of this draft, and shall not be modified or deleted.

IEEE Standards Association
445 Hoes Lane
Piscataway, NJ 08854, USA

1 Editors' Foreword

2 This draft standard is an amendment. The scope of changes to the base standard is thus strictly limited, as
3 detailed in the [PAR](#).

4 Information on participation in this project, and in the IEEE 802.1 Working Group can be found [here](#).

5 Participation in 802.1 standards development

6 Comments on this draft are encouraged. **NOTE: All issues related to IEEE standards presentation style,
7 formatting, spelling, etc. are routinely handled between the 802.1 Editor and the IEEE Staff Editors
8 prior to publication, after balloting and the process of achieving agreement on the technical content
9 of the standard is complete.** Readers are urged to devote their valuable time and energy only to comments
10 that materially affect either the technical content of the document or the clarity of that technical content.
11 Comments should not simply state what is wrong, but also what might be done to fix the problem.

12 Full participation in the work of IEEE 802.1 requires attendance at IEEE 802 meetings. Information on 802.1
13 activities, working papers, and email distribution lists etc. can be found on the 802.1 Website:

14 <http://ieee802.org/1/>

15 Use of the email distribution list is not presently restricted to 802.1 members, and the working group has a
16 policy of considering ballot comments from all who are interested and willing to contribute to the development
17 of the draft. Individuals not attending meetings have helped to identify sources of misunderstanding and
18 ambiguity in past projects. The email lists exist primarily to allow the members of the working group to
19 develop standards, and are not a general forum. All contributors to the work of 802.1 should familiarize
20 themselves with the IEEE patent policy and anyone using the mail distribution will be assumed to have done
21 so. Information can be found at <http://standards.ieee.org/about/sasb/patcom/materials.html>

22 Comments on this document may be sent to the 802.1 email exploder, to the Editor, or to the Chair of the
23 802.1 Working Group.

24 Silvana Rodrigues Glenn Parsons
25 Editor, P802.1AS-2020-Rev Chair, 802.1 Working Group
26 Email:silvana.rodrigues@huawei.com Email:glenn.parsons@ericsson.com

27 NOTE: Comments whose distribution is restricted in any way cannot be considered, and may not be
28 acknowledged.

29 **All participants in IEEE standards development have responsibilities under the IEEE patent policy and
30 should familiarize themselves with that policy, see
31 <http://standards.ieee.org/about/sasb/patcom/materials.html>**

32 As part of our IEEE 802 process, the text of the PAR and CSD (Criteria for Standards Development, formerly
33 referred to as the 5 Criteria or 5C's) is reviewed on a regular basis in order to ensure their continued validity.
34 A vote of "Approve" on this draft is also an affirmation by the balloter that the PAR is still valid.

1 Project Authorization Request, Scope, Purpose, and Criteria for Standards Development (CSD)

3 The complete PAR, as approved by IEEE NesCom 5 June 2023, can be found at:

4 <https://development.standards.ieee.org/myproject-web/public/view.html#pardetail/10637>

5 The ‘Scope of the Proposed changes’ and the ‘Need for the Project’ specify the changes to be made by this
6 amendment (see below).

7 Scope of the Proposed changes:

8 This standard specifies protocols, procedures, and managed objects used to ensure that the synchronization
9 requirements are met for time-sensitive applications, such as audio, video, and time-sensitive control, across
10 networks, for example, IEEE 802 and similar media. This includes the maintenance of synchronized time
11 during normal operation and following addition, removal, or failure of network components and network
12 reconfiguration. It specifies the use of IEEE 1588(TM) specifications where applicable in the context of IEEE
13 Std 802.1Q(TM). Synchronization to an externally provided timing signal [e.g., a recognized timing standard
14 such as Coordinated Universal Time (UTC) or International Atomic Time (TAI)] is not part of this standard but
15 is not precluded.

16 Need for the Project:

17 This revision project is needed in order to incorporate approved amendments and corrigenda, to incorporate
18 technical and editorial corrections to existing functionality, and to ensure that consistency is maintained in the
19 consolidated text.

20 Criteria for Standards Development:

21 There is no CSD statement since this maintenance project is not intended to provide any new
22 functionality

23 Notes from the Editor:

- 24 1) Changes from IEEE Std 802.1ASdr-2024 are not highlighted, as only terminology was
25 changed. Changes done by Std 802.1AS-2020_Cor1-2021 are not highlighted either.
- 26 2) All changes made in the draft that are based on IEEE Std 802.1ASdm-2024 are in magenta
27 color. Note that text deleted are not marked. Just the title of Clause 18 has the magenta color, as
28 it is obvious that the whole clause was added based on ASdm. Figures that were changed based
29 on ASdm has the magenta color in the figure title.
- 30 3) All changes made in the draft that are based on IEEE Std 802.1ASdn-2024 are in blue color.
31 Note that text deleted are not marked. Just the title of Clause 17 has the blue color, as it is
32 obvious that the whole clause was added based on ASdn. Note that ASdm changed clause 17,
33 and those changes are in magenta color.
- 34 4) The following Maintenance items were addressed in this draft: 369, 371, 372, 375, and 376.
35 Deleted text related to these Maintenance items are in red, changes made based on these
36 maintenance items are in green. Figures that were changed based on the Maintenance items has
37 the green color in the figure title. The changes for Maintenance items 369, 371, 372, and 375
38 are described in
39 https://www.ieee802.org/1/files/public/docs2024/maint-as-garner-discussion-of-fixes-needed-to-resolve-maint-items-369-371-372-0924-v02.pdf
- 41 5) Figure 11-9 was changed in ASdm and as a result of Maintenance item 375, so it has the
42 magenta color and the green color in the figure title.

1

2

(Revision of IEEE Std 802.1AS™-2020)

3

4

5 **Draft Standard for
Local and Metropolitan Area Networks—
7 Timing and Synchronization for
8 Time-Sensitive Applications**

9 Unapproved draft, prepared by the
10 **Maintenance Task Group of IEEE 802.1**

11 Sponsored by the

12 **LAN/MAN Standards Committee**
13 of the
14 **IEEE Computer Society**

15 Copyright 2024 by the IEEE.

16 Three Park Avenue

17 New York, New York 10016-5997, USA

18 All rights reserved.

19 This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to
20 change. USE AT YOUR OWN RISK! IEEE copyright statements SHALL NOT BE REMOVED from draft or
21 approved IEEE standards, or modified in any way. Because this is an unapproved draft, this document must
22 not be utilized for any conformance/compliance purposes. Permission is hereby granted for officers from each
23 IEEE Standards Working Group or Committee to reproduce the draft document developed by that Working
24 Group for purposes of international standardization consideration. IEEE Standards Department must be
25 informed of the submission for consideration prior to any reproduction for international standardization
26 consideration (stds.ipr@ieee.org). Prior to adoption of this document, in whole or in part, by another
27 standards development organization, permission must first be obtained from the IEEE Standards Department
28 (stds.ipr@ieee.org). When requesting permission, IEEE Standards Department will require a copy of the
29 standard development organization's document highlighting the use of IEEE content. Other entities seeking
30 permission to reproduce this document, in whole or in part, must also obtain permission from the IEEE
31 Standards Department.

32

33 IEEE Standards Activities Department
34 445 Hoes Lane
35 Piscataway, NJ 08854, USA

1 Abstract: This is a Maintenance roll-up of IEEE Std 802.1AS™-2020 with the corrigendum IEEE
2 Std 802.1AS-2020/Cor1 and amendments IEEE Std 802.1ASdr, IEEE Std 802.1ASdm, and IEEE
3 Std 802.1ASdn.

4 Keywords: best timeTransmitter, frequency offset, Grandmaster Clock, Grandmaster PTP
5 Instance, IEEE 802.1AS™, IEEE 802.1ASdr™, phase offset, PTP End Instance, PTP Relay
6 Instance, synchronization, syntonization, time-aware system

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

Copyright © 2024 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published xx Month 202x. Printed in the United States of America.

MoCA is a registered trademark of the Multimedia over Coax Alliance.

POSIX is a registered trademark of The Institute of Electrical and Electronics Engineers, Incorporated.

IEEE and IEEE 802 are registered trademarks in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-8-8557-0539-3 STD26794
Print: ISBN 978-8-8557-0540-9 STDPD26794

IEEE prohibits discrimination, harassment and bullying.

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

1 Important Notices and Disclaimers Concerning IEEE Standards Documents

3 IEEE Standards documents are made available for use subject to important notices and legal disclaimers.
4 These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>),
5 appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning
6 IEEE Standards Documents.”

7 Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

9 IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards
10 Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus
11 development process, which brings together volunteers representing varied viewpoints and interests to
12 achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic,
13 and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE
14 or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and
15 establishes rules to promote fairness in the consensus development process, IEEE does not independently
16 evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained
17 in its standards.

18 IEEE makes no warranties or representations concerning its standards, and expressly disclaims all
19 warranties, express or implied, concerning this standard, including but not limited to the warranties of
20 merchantability, fitness for a particular purpose and non-infringement. IEEE Standards documents do not
21 guarantee safety, security, health, or environmental protection, or guarantee against interference with or
22 from other devices or networks. In addition, IEEE does not warrant or represent that the use of the material
23 contained in its standards is free from patent infringement. IEEE Standards documents are supplied “AS IS”
24 and “WITH ALL FAULTS.”

25 Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there
26 are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to
27 the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and
28 issued is subject to change brought about through developments in the state of the art and comments
29 received from users of the standard.

30 In publishing and making its standards available, IEEE is not suggesting or rendering professional or other
31 services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any
32 other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon their
33 own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate,
34 seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

35 IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
36 EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE
37 NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS;
38 OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
39 WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
40 OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE
41 UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND
42 REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

1 Translations

2 The IEEE consensus balloting process involves the review of documents in English only. In the event that an
3 IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

4 Official statements

5 A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board
6 Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its
7 committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures,
8 symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall
9 make it clear that the presenter's views should be considered the personal views of that individual rather
10 than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements
11 made by volunteers may not represent the formal position of their employer(s) or affiliation(s).

12 Comments on standards

13 Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of
14 membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations,**
15 **consulting information, or advice pertaining to IEEE Standards documents.**

16 Suggestions for changes in documents should be in the form of a proposed change of text, together with
17 appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is
18 important that any responses to comments and questions also receive the concurrence of a balance of interests.
19 For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of
20 Governors are not able to provide an instant response to comments, or questions except in those cases where
21 the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation
22 requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard
23 is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the
24 Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).¹ An IEEE Account is
25 needed to access the application.

26 Comments on standards should be submitted using the [Contact Us](#) form.²

27 Laws and regulations

28 Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the
29 provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory
30 requirements. Implementers of the standard are responsible for observing or referring to the applicable
31 regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not
32 in compliance with applicable laws, and these documents may not be construed as doing so.

33 Data privacy

34 Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and
35 data ownership in the context of assessing and using the standards in compliance with applicable laws and
36 regulations.

¹ Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

² Available at: <https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html>.

1 Copyrights

2 IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws.
3 They are made available by IEEE and are adopted for a wide variety of both public and private uses. These
4 include both use, by reference, in laws and regulations, and use in private self-regulation, standardization,
5 and the promotion of engineering practices and methods. By making these documents available for use and
6 adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright
7 to the documents.

8 Photocopies

9 Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license
10 to photocopy portions of any individual standard for company or organizational internal use or individual,
11 non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance
12 Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400;
13 <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational
14 classroom use can also be obtained through the Copyright Clearance Center.

15 Updating of IEEE Standards documents

16 Users of IEEE Standards documents should be aware that these documents may be superseded at any time
17 by the issuance of new editions or may be amended from time to time through the issuance of amendments,
18 corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the
19 document together with any amendments, corrigenda, or errata then in effect.

20 Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years
21 old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of
22 some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that
23 they have the latest edition of any IEEE standard.

24 In order to determine whether a given document is the current edition and whether it has been amended
25 through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).³ For more
26 information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

27 Errata

28 Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).⁴ Search for standard number
29 and year of approval to access the web page of the published standard. Errata links are located under the
30 Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to
31 periodically check for errata.

32 Patents

33 IEEE standards are developed in compliance with the [IEEE SA Patent Policy](#).⁵

34 Attention is called to the possibility that implementation of this standard may require use of subject matter
35 covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the
36 existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has

³ Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

⁴ Available at: <https://standards.ieee.org/standard/index.html>.

⁵ Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

1 filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the
2 IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may
3 indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without
4 compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of
5 any unfair discrimination to applicants desiring to obtain such licenses.

6 Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not
7 responsible for identifying Essential Patent Claims for which a license may be required, for conducting
8 inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or
9 conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing
10 agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that
11 determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their
12 own responsibility. Further information may be obtained from the IEEE Standards Association.

13 IMPORTANT NOTICE

14 Technologies, application of technologies, and recommended procedures in various industries evolve over
15 time. The IEEE standards development process allows participants to review developments in industries,
16 technologies, and practices, and to determine what, if any, updates should be made to the IEEE standard.
17 During this evolution, the technologies and recommendations in IEEE standards may be implemented in
18 ways not foreseen during the standard's development. IEEE standards development activities consider
19 research and information presented to the standards development group in developing any safety
20 recommendations. Other information about safety practices, changes in technology or technology
21 implementation, or impact by peripheral systems also may be pertinent to safety considerations during
22 implementation of the standard. Implementers and users of IEEE Standards documents are responsible for
23 determining and complying with all appropriate safety, security, environmental, health, and interference
24 protection practices and all applicable laws and regulations.

1 Participants

2 <<The following lists will be updated in the usual way prior to publication>>

3 At the time this standard was submitted to the IEEE SA Standards Board for approval, the IEEE 802.1
4 Working Group had the following membership:

5 **Glenn Parsons**, *Chair*

6 **Jessy V. Rouyer**, *Vice Chair*

7 **Mark Hantel**, *Maintenance Task Group Chair*

8 **Silvana Rodrigues**, *Editor IEEE Std 802.1AS*

9

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

3 When the IEEE SA Standards Board approved this standard on 15 February 2024, it had the following
4 membership:

5
6
7
8

David J. Law, *Chair*
Vacant Position, *Vice Chair*
Gary Hoffman, *Past Chair*
Alpesh Shah, *Secretary*

9 *Member Emeritus

10

1 Introduction

This introduction is not part of IEEE Std 802.1AS-2020-Rev, IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications.

2 The first edition of IEEE Std 802.1AS was published in 2011. A first corrigendum, IEEE Std
3 802.1AS-2011/Cor1-2013, provided technical and editorial corrections. A second corrigendum, IEEE Std
4 802.1AS-2011/Cor2-2015 provided additional technical and editorial corrections.

5 The second edition, IEEE Std 802.1AS-2020, added support for multiple gPTP domains, Common Mean
6 Link Delay Service, external port configuration, and Fine Timing Measurement for IEEE 802.11 transport.

7 Backward compatibility with IEEE Std 802.1AS-2011 was maintained. A corrigendum, IEEE Std
8 802.1AS-2020/Cor1-2021, provides technical and editorial corrections.

9 The third edition, IEEE Std 802.1AS-202x, is a roll-up of IEEE Std 802.1ASTM-2020 with the corrigendum
10 IEEE Std 802.1AS-2020/Cor1 and amendments IEEE Std 802.1ASdr, IEEE Std 802.1ASdm, and IEEE Std
11 802.1ASdn.

12

1 Contents

21.	Overview.....	22
3	1.1 Scope.....	22
4	1.2 Purpose.....	22
52.	Normative references.....	23
63.	Definitions	25
74.	Acronyms and abbreviations	28
85.	Conformance.....	31
9	5.1 Requirements terminology	31
10	5.2 Protocol Implementation Conformance Statement (PICS).....	31
11	5.3 Time-aware system requirements	31
12	5.4 PTP Instance requirements and options.....	31
13	5.5 .MAC-specific timing and synchronization methods for full-duplex IEEE 802.3 links	34
14	5.6 MAC-specific timing and synchronization methods for IEEE Std 802.11-2016	34
15	5.7 MAC-specific timing and synchronization methods for IEEE 802.3 EPON	35
16	5.8 MAC-specific timing and synchronization methods for coordinated shared network (CSN).....	35
186.	Conventions	36
19	6.1 General.....	36
20	6.2 Service specification method and notation	36
21	6.3 Lexical form syntax	36
22	6.4 Data types and on-the-wire formats.....	36
237.	Time-synchronization model for a packet network	41
24	7.1 General.....	41
25	7.2 Architecture of a time-aware network	41
26	7.3 Time synchronization	48
27	7.4 PTP Instance architecture	51
28	7.5 Differences between gPTP (IEEE Std 802.1AS) and PTP (IEEE Std 1588-2019)	52
298.	IEEE 802.1AS concepts and terminology	54
30	8.1 gPTP domain.....	54
31	8.2 Timescale	54
32	8.3 Link asymmetry	56
33	8.4 Messages	57
34	8.5 Ports	59
35	8.6 PTP Instance characterization.....	60
369.	Application interfaces	65
37	9.1 Overview of the interfaces.....	65
38	9.2 ClockSourceTime interface	66
39	9.3 ClockTargetEventCapture interface	67
40	9.4 ClockTargetTriggerGenerate interface	68
41	9.5 ClockTargetClockGenerator interface.....	70

1	9.6	ClockTargetPhaseDiscontinuity interface	71
2	10.	Media-independent layer specification	73
3	10.1	Overview.....	73
4	10.2	Time-synchronization state machines.....	75
5	10.3	Best timeTransmitter clock selection, external port configuration, and announce interval setting state machines	107
6	10.4	State machines related to signaling gPTP capability	148
7	10.5	Message attributes.....	155
8	10.6	Message formats	156
9	10.7	Protocol timing characterization	169
10	11.	Media-dependent layer specification for full-duplex point-to-point links.....	173
11	11.1	Overview.....	173
12	11.2	State machines for MD entity specific to full-duplex point-to-point links.....	180
13	11.3	Message attributes.....	212
14	11.4	Message formats	214
15	11.5	Protocol timing characterization	223
16	11.6	Control of computation of neighborRateRatio	225
17	11.7	Control of computation of meanLinkDelay	225
18	12.	Media-dependent layer specification for IEEE 802.11 links	227
19	12.1	Overview.....	227
20	12.2	Messages.....	232
21	12.3	Determination of Timing Measurement and Fine Timing Measurement capability	233
22	12.4	Determination of asCapable.....	233
23	12.5	State machines	234
24	12.6	FTM parameters.....	246
25	12.7	Format of VendorSpecific information element.....	248
26	12.8	Synchronization message interval	248
27	13.	Media-dependent layer specification for interface to IEEE 802.3 Ethernet passive optical network link	250
28	13.1	Overview.....	250
29	13.2	Message attributes.....	254
30	13.3	Message format.....	254
31	13.4	Determination of asCapable.....	257
32	13.5	Layering for IEEE 802.3 EPON links	257
33	13.6	Service interface definitions	258
34	13.7	MD entity global variables	261
35	13.8	State machines	261
36	13.9	Message transmission intervals	264
37	14.	Timing and synchronization management.....	265
38	14.1	General.....	265
39	14.2	Default Parameter Data Set (defaultDS).....	268
40	14.3	Current Parameter Data Set (currentDS)	272
41	14.4	Parent Parameter Data Set (parentDS).....	273
42	14.5	Time Properties Parameter Data Set (timePropertiesDS).....	277
43	14.6	Path Trace Parameter Data Set (pathTraceDS).....	278
44	14.7	Acceptable TimeTransmitter Table Parameter Data Set	

1	(acceptableTimeTransmitterTableDS).....	279
2	14.7a PTP Instance Synchronization Parameter Data Set (ptpInstanceSyncDS).....	280
3	14.7b Drift Tracking Parameter Data Set (driftTrackingDS)	280
4	14.8 Port Parameter Data Set (portDS).....	281
5	14.9 Description Port Parameter Data Set (descriptionPortDS)	293
6	14.10 Port Parameter Statistics Data Set (portStatisticsDS).....	294
7	14.11 Acceptable TimeTransmitter Port Parameter Data Set (acceptableTimeTransmitterPortDS)	298
8	14.12 External Port Configuration Port Parameter Data Set (externalPortConfigurationPortDS)	298
9	14.13 Asymmetry Measurement Mode Parameter Data Set (asymmetryMeasurementModeDS)	299
10	14.14 Common Services Port Parameter Data Set (commonServicesPortDS)	300
11	14.15 Common Mean Link Delay Service Default Parameter Data Set (cmldsDefaultDS)	300
12	14.16 Common Mean Link Delay Service Link Port Parameter Data Set (cmldsLinkPortDS)....	301
13	14.17 Common Mean Link Delay Service Link Port Parameter Statistics Data Set (cmldsLinkPortStatisticsDS)	306
14	14.18 Common Mean Link Delay Service Asymmetry Measurement Mode Parameter Data Set (cmldsAsymmetryMeasurementModeDS).....	308
15	14.19 Hot Standby System Parameter Data Set (hotStandbySystemDS).....	308
16	14.20 Hot Standby System Description Parameter Data Set (hotStandbySystemDescriptionDS)	310
17	20 15. Management Information Base (MIB)	311
18	21 15.1 Internet Standard Management Framework	311
19	22 15.2 Structure of the MIB	311
20	23 15.3 Relationship to MIB in IEEE Std 802.1AS-2011	318
21	24 15.4 Security considerations	319
22	25 15.5 Textual conventions defined in this MIB	320
23	26 15.6 IEEE 802.1AS MIB module,	320
24	27 16. Media-dependent layer specification for CSN.....	472
25	28 16.1 Overview.....	472
26	29 16.2 Coordinated Shared Network characteristics.....	472
27	30 16.3 Layering for CSN links.....	473
28	31 16.4 Path delay measurement over a CSN backbone	474
29	32 16.5 Synchronization messages	477
30	33 16.6 Specific CSN requirements.....	481
31	34 16.7 Grandmaster PTP Instance capability	482
32	35 16.8 CSN clock and node performance requirements	482
33	36 17. YANG data model	483
34	37 17.1 YANG framework	483
35	38 17.2 IEEE 802.1AS YANG data model	484
36	39 17.3 Structure of the YANG data model	488
37	40 17.4 Security considerations	489
38	41 17.5 YANG schema tree definitions.....	490
39	42 17.6 YANG modules,	495
40	43 18. Hot standby	549
41	44 18.1 General.....	549
42	45 18.2 Overview.....	549
43	46 18.3 PTP Instance configuration.....	550
44	47 18.4 PtpInstanceSyncStatus state machine	550
45	48 18.5 HotStandbySystem state machine.....	554

1	18.6 PrimarySecondaryOffset state machine	559
2	Annex A (normative) Protocol Implementation Conformance Statement (PICS) proforma	561
3	A.1 Introduction.....	561
4	A.2 Abbreviations and special symbols.....	561
5	A.3 Instructions for completing the PICS proforma.....	562
6	A.4 PICS proforma for IEEE Std 802.1AS-2020	564
7	A.5 Major capabilities	565
8	A.6 Media access control methods	566
9	A.7 Minimal time-aware system.....	566
10	A.8 Signaling	568
11	A.9 Best timeTransmitter clock	569
12	A.10 Grandmaster-capable PTP Instance	571
13	A.11 Media-independent timeTransmitter.....	572
14	A.12 Media-independent performance requirements	573
15	A.13 Media-dependent, full-duplex point-to-point link	573
16	A.14 Media-dependent IEEE 802.11 link.....	576
17	A.15 Media-dependent IEEE 802.3 EPON link	577
18	A.16 Media-dependent CSN link	578
19	A.17 Media-dependent MoCA link	578
20	A.18 Media-dependent ITU-T G.hn link.....	578
21	A.19 Remote management.....	579
22	A.20 Application interfaces	579
23	A.21 External port configuration.....	579
24	Annex B	580
25	B.1 LocalClock requirements	580
26	B.2 PTP Instance requirements	585
27	B.3 End-to-end time-synchronization performance	586
28	B.4 End-to-end jitter and wander performance	586
29	Annex C	588
30	C.1 Overview.....	588
31	C.2 TAI and UTC	588
32	C.3 NTP and GPS.....	590
33	C.4 Timescale conversions	590
34	C.5 Time zones and GMT	591
35	Annex D	592
36	Annex E	593
37	Annex F (informative) PTP profile included in this standardGeneral	594
38	F.1 PTP profile included in this standardGeneral	594
39	F.2 Identification.....	594
40	F.3 PTP attribute values	595
41	F.4 PTP options.....	595
42	F.5 LocalClock and PTP Instance performance requirements.....	596
43	Annex G (informative) The asymmetry compensation measurement procedure based on line-swapping ..	597
44	G.1 Introduction.....	597
45	G.2 Pre-conditions for measurement	597

1	G.3	Measurement procedure.....	597
2	Annex H.....		600

1 List of figures

2 Figure 7-1—Time-aware network example	42
3 Figure 7-2—Time-aware network of Figure 7-1 after an access network link failure	43
4 Figure 7-3—Time-aware network example for multiple gPTP domains	44
5 Figure 7-4—Time-aware network example for synchronization path redundancy, with one clock 6 source providing time to two domains	45
7 Figure 7-5—Time-aware network example for GM redundancy with one primary GM and one hot- 8 standby GM, which are separated in two gPTP domains	46
9 Figure 7-6—Time-aware network example for hot standby with both GM and partial path 10 redundancy	47
11 Figure 7-7—Conceptual medium delay measurement	48
12 Figure 7-8—Model for a PTP Instance and its interfaces to higher-layer applications	51
13 Figure 8-1—Propagation asymmetry	56
14 Figure 8-2—Definition of message timestamp point, reference plane, timestamp 15 measurement plane, and latency constants	58
16 Figure 9-1—Application interfaces	66
17 Figure 10-1—Model for media-independent layer of PTP Instance	74
18 Figure 10-2—Time-synchronization state machines—overview and interrelationships	76
19 Figure 10-3—SiteSyncSync state machine	93
20 Figure 10-4—PortSyncSyncReceive state machine	95
21 Figure 10-5—ClockTimeTransmitterSyncSend state machine	98
22 Figure 10-6—ClockTimeTransmitterSyncOffset state machine	100
23 Figure 10-7—ClockTimeTransmitterSyncReceive state machine	102
24 Figure 10-8—PortSyncSyncSend state machine	105
25 Figure 10-9—ClockTimeReceiverSync state machine	107
26 Figure 10-10—Example timeTransmitter/timeReceiver hierarchy of PTP Instances	109
27 Figure 10-11—Best timeTransmitter clock selection state machines—overview and interrelationships ..	116
28 Figure 10-12—External port configuration state machines—overview and interrelationships	118
29 Figure 10-13—PortAnnounceReceive state machine	127
30 Figure 10-14—PortAnnounceInformation state machine	130
31 Figure 10-15—PortStateSelection state machine	134
32 Figure 10-16—PortAnnounceInformationExt state machine	136
33 Figure 10-17—PortStateSettingExt state machine	139
34 Figure 10-18—PortAnnounceTransmit state machine	141
35 Figure 10-19—AnnounceIntervalSetting state machine	144
36 Figure 10-20—SyncIntervalSetting state machine	147
37 Figure 10-21—GptpCapableTransmit state machine	150
38 Figure 10-22—GptpCapableReceive state machine	152
39 Figure 10-23—GptpCapableIntervalSetting state machine	154
40 Figure 11-1—Propagation delay measurement using peer-to-peer delay mechanism	174
41 Figure 11-2—Transport of time-synchronization information	176
42 Figure 11-3—Model for a PTP Instance of a time-aware system with full-duplex point-to-point links ...	179
43 Figure 11-4—Detail of MD entity time-synchronization state machines for full-duplex 44 point-to-point links	180
45 Figure 11-5—Peer-to-peer delay mechanism state machines— 46 overview and interrelationships	181
47 Figure 11-6—MDSyncReceiveSM state machine	190
48 Figure 11-7—MDSyncSendSM state machine	195
49 Figure 11-8—OneStepTxOperSetting state machine	197
50 Figure 11-9—MDPdelayReq state machine	205
51 Figure 11-10—MDPdelayResp state machine	208
52 Figure 11-11—LinkDelayIntervalSetting state machine	211

1	Figure 12-1—Timing measurement procedure for IEEE 802.11 links	228
2	Figure 12-2—Fine Timing Measurement procedure for IEEE 802.11 links.....	229
3	Figure 12-3—Illustration of Fine Timing Measurement burst	230
4	Figure 12-4—Media-dependent and lower entities in stations with IEEE 802.11 links	232
5	Figure 12-5—TimeTransmitter state machine A	
6	(a) For TM, receives information from the PortSync entity and sends to timeReceiver, and	
7	(b) for FTM, receives and stores information from the PortSync entity	235
9	Figure 12-6—TimeTransmitter state machine B	
10	(a) For TM, not invoked and	
11	(b) for FTM, receives initial FTM request from timeReceiver and sends information received from upstream to timeReceiver in successive FTM frames	236
13	Figure 12-7—TimeReceiver state machine	242
14	Figure 12-8—Format of VendorSpecific information element when Type = 0	248
15	Figure 13-1—IEEE 802.3 EPON time-synchronization interfaces	253
16	Figure 13-2—IEEE 802.3 EPON interface model.....	258
17	Figure 13-3—State machine for IEEE 802.3 EPON requester.....	262
18	Figure 13-4—State machine for IEEE 802.3 EPON responder.....	264
19	Figure 16-1—Example of CSN backbone in a TSN LAN	472
20	Figure 16-2—Media-dependent and lower entities in CSN nodes	473
21	Figure 16-3—Path types over CSN as IEEE 802.1AS backbone.....	474
22	Figure 16-4—Propagation delay and residence time over a CSN backbone.....	474
23	Figure 16-5—CSN node-to-node path delay measurement.....	475
24	Figure 16-6—IEEE 802.1AS Sync message propagation over the CSN backbone	477
25	Figure 17-1—Overview of YANG tree	485
26	Figure 17-2—PTP Instance detail.....	486
27	Figure 17-3—PTP Port detail	487
28	Figure 17-4—Common services detail.....	488
29	Figure 18-1—Model for hot standby entity in a time-aware system, and its interfaces to higher-layer applications.....	549
31	Figure 18-2—PtpInstanceSyncStatus state machine	553
32	Figure 18-3—HotStandbySystem state machine	559
33	Figure 18-4—PrimarySecondaryOffset state machine	560
34	Figure B-1—Wander generation (TDEV) requirement for LocalClock entity	582
35	Figure B-2—ADEV limit corresponding to wander generation requirement of Figure B-1	583
37	Figure B-3—PTPDDEV limit corresponding to wander generation requirement of Figure B-1	584
39	Figure B-4—MTIE masks met for maximum endpoint filter bandwidths of Table B-4	587
41	Figure G-1—Asymmetry compensation measurement procedure	598

1 List of tables

2 Table 6-1—Primitive data types	37
3 Table 8-1—Default values for priority1, for the respective media.....	61
4 Table 8-2—TimeSource enumeration	63
5 Table 10-1—Summary of scope of global variables used by time synchronization state machines 6 (see 10.2.4 and 10.2.5).....	84
7 Table 10-2—PTP Port state definitions	108
8 Table 10-3—Summary of scope of global variables used by best timeTransmitter clock selection, 9 external port configuration, and announce interval setting state machines (see 10.3.9 10 and 10.3.10)	119
11 Table 10-4—Destination address for Announce and Signaling messages	155
12 Table 10-5—EtherType for Announce and Signaling messages	156
13 Table 10-7—PTP message header	157
14 Table 10-6—Propagate TLVs of IEEE Std 1588-2019	157
15 Table 10-8—Values for messageType field	158
16 Table 10-9—Values of flag bits.....	159
17 Table 10-10—messageTypeSpecific semantics	160
18 Table 10-11—Announce message fields	161
19 Table 10-12—Path trace TLV	162
20 Table 10-13—Signaling message fields	163
21 Table 10-14—Message interval request TLV	164
22 Table 10-15—Interpretation of special values of logLinkDelayInterval.....	165
23 Table 10-16—Interpretation of special values of logTimeSyncInterval	165
24 Table 10-17—Interpretation of special values of logAnnounceInterval	166
25 Table 10-18—Definitions of bits of flags field 26 of message interval request TLV	166
27 Table 10-19—gPTP-capable TLV	167
28 Table 10-20—gPTP-capable message interval request TLV.....	168
29 Table 10-21—Interpretation of special values of logGptpCapableMessageInterval.....	169
30 Table 11-1—Value of meanLinkDelayThresh for various links	182
31 Table 11-2 —Summary of scope of global variables used by 32 time synchronization state machines (see 10.2.4 and 10.2.5).....	184
33 Table 11-3—Destination address for Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, 34 and Pdelay_Resp_Follow_Up messages	213
35 Table 11-4—EtherType for Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, 36 and Pdelay_Resp_Follow_Up messages	213
37 Table 11-5—Values for messageType field	215
38 Table 11-7—References for sequenceId value exceptions.....	216
39 Table 11-6—Value of correctionField.....	216
40 Table 11-8—Sync message fields if twoStep flag is TRUE.....	217
41 Table 11-9—Sync message fields if twoStep flag is FALSE.....	217
42 Table 11-10—Follow_Up message fields	218
43 Table 11-11—Follow_Up information TLV	219
44 Table 11-11a—Drift_Tracking TLV	221
45 Table 11-12—Pdelay_Req message fields	222
46 Table 11-13—Pdelay_Resp message fields.....	222
47 Table 11-14—Pdelay_Resp_Follow_Up message fields	223
48 Table 12-1—Values of bits of tmFtmSupport	233
49 Table 12-2—FTM parameters relevant to time-synchronization transport	247
50 Table 12-3—Values of Burst Duration and Min Delta FTM, 51 for each value of currentLogSyncInterval	247
52 Table 12-4—Values of the Type field in the VendorSpecific information element.....	248

1 Table 13-1—TIMESYNC message fields	255
2 Table 14-1—defaultDS table	271
3 Table 14-2—currentDS table	274
4 Table 14-3—parentDS table	276
5 Table 14-4—timePropertiesDS table	278
6 Table 14-6—acceptableTimeTransmitterTableDS table	279
7 Table 14-5—pathTraceDS table	279
8 Table 14-6b—driftTrackingDS table	281
9 Table 14-6a—ptpInstanceSyncDS table	281
10 Table 14-7—portState enumeration	282
11 Table 14-8—delayMechanism enumeration	282
12 Table 14-9—Description of pdelayTruncatedTimestampsArray	289
13 Table 14-10—portDS table	291
14 Table 14-11—descriptionPortDS table	293
15 Table 14-12—portStatisticsDS table	297
16 Table 14-13—acceptableTimeTransmitterPortDS table	298
17 Table 14-14—externalPortConfigurationPortDS table	299
18 Table 14-15—asymmetryMeasurementModeDS table	299
19 Table 14-16—commonServicesPortDS table	300
20 Table 14-17—cmldsDefaultDS table	301
21 Table 14-18—cmldsLinkPortDS table	305
22 Table 14-19—cmldsLinkPortStatisticsDS table	307
23 Table 14-20—cmldsAsymmetryMeasurementModeDS table	308
24 Table 14-22—hotStandbySystemDescriptionDS table	310
25 Table 14-21—hotStandbySystemDS table	310
26 Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference	312
27 Table 16-1—CSN TLV	479
28 Table 16-2—Definitions and option selections per link technology	481
29 Table 17-1—Summary of the YANG modules	489
30 Table B-1—Wander generation TDEV requirement for LocalClock entity	582
31 Table B-2—ADEV limit corresponding to wander generation requirement 32 of Table B-1	583
33 Table B-3—PTPDEV limit corresponding to wander generation requirement 34 of Table B-1	584
35 Table B-4—Maximum endpoint filter bandwidths needed to meet respective 36 MTIE masks and peak-to-peak jitter limits	586
37 Table B-5—Breakpoints for Mask 1	587
38 Table B-6—Breakpoints for Mask 2	587
39 Table C-1—Timescale parameters	588
40 Table C-2—Timescale conversions	591

² **Draft Standard for
3 Local and Metropolitan Area Networks—**

⁴ **Timing and Synchronization for
5 Time-Sensitive Applications**

⁶ **1. Overview**

⁷ **1.1 Scope**

⁸ This standard specifies protocols, procedures, and managed objects used to ensure that the synchronization
⁹ requirements are met for time-sensitive applications, such as audio, video, and time-sensitive control, across
¹⁰ networks, for example, IEEE 802 and similar media. This includes the maintenance of synchronized time
¹¹ during normal operation and following addition, removal, or failure of network components and network
¹² reconfiguration. It specifies the use of IEEE 1588TM specifications where applicable in the context of
¹³ IEEE Std 802.1QTM.¹ Synchronization to an externally provided timing signal [e.g., a recognized timing
¹⁴ standard such as Coordinated Universal Time (UTC) or International Atomic Time (TAI)] is not part of this
¹⁵ standard but is not precluded.

¹⁶ **1.2 Purpose**

¹⁷ This standard enables systems to meet the respective jitter, wander, and time-synchronization requirements
¹⁸ for time-sensitive applications, including those that involve multiple streams delivered to multiple end
¹⁹ stations. To facilitate the widespread use of packet networks for these applications, synchronization
²⁰ information is one of the components needed at each network element where time-sensitive application data
²¹ are mapped or demapped or a time-sensitive function is performed. This standard leverages the work of the
²² IEEE 1588 Working Group by developing the additional specifications needed to address these
²³ requirements.

24

¹ Information on references can be found in Clause 2.

1 2. Normative references

2 The following referenced documents are indispensable for the application of this standard (i.e., they must be
 3 understood and used; therefore, each referenced document is cited in text, and its relationship to this
 4 document is explained). For dated references, only the edition cited applies. For undated references, the
 5 latest edition of the referenced document (including any amendments or corrigenda) applies.

6 IEEE Std 754™-2008, IEEE Standard for Floating-Point Arithmetic.^{2, 3}

7 IEEE Std 802®-2014, IEEE Standard for Local and Metropolitan Area Networks—Overview and
 8 Architecture.

9 IEEE Std 802c™-2017, IEEE Standard for Local and Metropolitan Area Networks—Overview and
 10 Architecture—Amendment 2: Local Medium Access Control (MAC) Address Usage.

11 IEEE Std 802.1AC™-2016, IEEE Standard for Local and metropolitan area networks—Media Access
 12 Control (MAC) Service Definition.

13 IEEE Std 802.1AX™-2014, IEEE Standard for Local and metropolitan area networks—Link Aggregation.

14 IEEE Std 802d™-2017, IEEE Standard for Local and Metropolitan Area Networks—Overview and
 15 Architecture—Amendment 1: Allocation of Uniform Resource Name (URN) Values in IEEE 802®
 16 Standards.^{4, 5}

17 IEEE Std 802.1Q™-2018, IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged
 18 Networks.

19 IEEE Std 802.3™-2018, IEEE Standard for Ethernet.

20 IEEE Std 802.11™-2016, IEEE Standard for Information technology—Telecommunications and
 21 information exchange between systems—Local and metropolitan area networks—Specific requirements,
 22 Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

23 IEEE Std 1588™-2019, IEEE Standard for a Precision Clock Synchronization Protocol for Networked
 24 Measurement and Control Systems.

25 IEEE Std 1588e™-2024, IEEE Standard for Precision Clock Synchronization Protocol for Networked
 26 Measurement and Control Systems—Amendment 5: MIB and YANG Modules.

27 IERS Bulletin C (see <https://www.iers.org/IERS/EN/Publications/Bulletins/bulletins.html>).

28 IETF RFC 2863 (June 2000), The Interfaces Group MIB, K. McCloghrie and F. Kastenholz.⁶

29 IETF RFC 3410 (Dec. 2002), Introduction and Applicability Statements for Internet Standard Management
 30 Framework, J. Case, R. Mundy, D. Partain, and B. Stewart.

² IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org>).

³ The IEEE standards or products referenced in this clause are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

⁴ The IEEE standards or products referred to in 2 are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

⁵ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org>).

⁶ IETF Requests for Comments (RFCs) are available from the Internet Engineering Task Force (<https://www.rfc-editor.org>).

1 IETF RFC 3418 (Dec. 2002), Management Information Base (MIB) for the Simple Network Management
2 Protocol (SNMP), R. Presuhn, ed.

3 [IETF RFC 7950, The YANG 1.1 Data Modeling Language, August 2016.](#)⁷

4 ISO 80000-3:2006, Quantities and units — Part 3: Space and time.⁸

5 ITU-T Recommendation G.984.3, Amendment 2, Gigabit-capable Passive Optical Networks (G-PON):
6 Transmission convergence layer specification—Time-of-day distribution and maintenance updates and
7 clarifications.⁹

8 ITU-T Recommendation G.9960, Unified high-speed wire-line based home networking transceivers—
9 System architecture and physical layer specification [with ITU-T G.9961, commonly referred to as “G.hn”].

10 ITU-T Recommendation G.9961, Data link layer (DLL) for unified high-speed wire-line based home
11 networking transceivers [with ITU-T G.9960, commonly referred to as “G.hn”].

12 MoCA® MAC/PHY Specification v2.0, MoCA-M/P-SPEC-V2.0-20100507, Multimedia over Coax
13 Alliance (MoCA).¹⁰

14

⁷ IETF RFCs are available from the Internet Engineering Task Force (<https://www.ietf.org/>).

⁸ ISO publications are available from the International Organization for Standardization (<https://www.iso.org>) and the American National Standards Institute (<https://www.ansi.org>).

⁹ ITU-T publications are available from the International Telecommunications Union (<https://www.itu.int>).

¹⁰ MoCA specifications are available from the Multimedia over Coax Alliance (<http://www.mocalliance.org/specs>).

1 3. Definitions

2 For the purposes of this standard, the following terms and definitions apply. The *IEEE Standards Dictionary*
3 *Online* should be consulted for terms not defined in this clause.¹¹

4 **3.1 accuracy:** The mean of the time or frequency error between the clock under test and a reference clock
5 over an ensemble of measurements (see IEEE Std 1588-2019).

6 **3.2 Bridge:** Either a MAC Bridge or a VLAN-aware Bridge, as specified in Clause 5 of IEEE Std 802.1Q-
7 2018.

8 **3.3 clock:** A physical device that is capable of providing a measurement of the passage of time since a
9 defined epoch.

10 **3.3a default:** The value of an implementation parameter or capability selection, in the absence of an
11 overriding profile standard requirement or explicit configuration.

12 <<<Editor's note: SA editor added new subclauses (throughout the ASdm published standard) with a
13 letter (as seen above "3.3a") instead of renumbering it (in this case, "3.4"). For AS-revision, these
14 subclauses should be renumbered. The numbering of the clauses has been kept as the ASdm
15 published standard for this draft to ease the review of this first draft. They will be renumbered at the
16 next draft.>>>>

17 **3.4 device:** An entity implementing some functionality, e.g., a clock, a time-aware system, a port.

18 **3.5 direct communication:** A communication of IEEE 802.1AS information between two PTP Instances
19 with no intervening PTP Instance.

20 **3.6 end station:** A device attached to a local area network (LAN) or metropolitan area network (MAN) that
21 acts as a source of, and/or destination for, traffic carried on the LAN or MAN.

22 **3.6a epoch:** The origin of a timescale.

23 **3.7 event message:** A message that is timestamped on egress from a PTP Instance and ingress to a PTP
24 Instance.

25 NOTE—See 8.4.3.¹²

26 **3.8 fractional frequency offset:** The fractional offset, y , between a measured clock and a reference clock as
27 defined by the following:

$$28 \quad y = \frac{f_m - f_r}{f_r}$$

29 where f_m is the frequency of the measured clock and f_r is the frequency of the reference clock. The
30 measurement units of f_m and f_r are the same.

31 **3.9 frequency offset:** The offset between a measured frequency and a reference frequency as defined by
32 $f_m - f_r$, where f_m is the frequency of the measured clock and f_r is the frequency of the reference clock. The
33 measurement units of f_m and f_r are the same.

34 **3.10 general message:** A message that is not timestamped.

¹¹ IEEE Standards Dictionary Online is available at <https://dictionary.ieee.org/>.

¹² Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement the standard.

1 **3.11 gPTP communication path:** A segment of a generalized precision time protocol (gPTP) domain that
2 enables direct communication between two PTP Instances.

3 NOTE—See 8.1.

4 **3.12 grandmaster-capable PTP Instance:** A PTP Instance that is capable of being a Grandmaster
5 PTP Instance.

6 **3.13 Grandmaster Clock:** In the context of a single PTP domain, the synchronized time of a PTP Instance
7 that is the source of time to which all other PTP Instances in the domain are synchronized.

8 **3.14 Grandmaster PTP Instance:** A PTP Instance containing the Grandmaster Clock.

9 NOTE—None of the PTP Ports of a Grandmaster PTP Instance is in the TimeReceiverPort state.

10 **3.15 local area network (LAN):** A network of devices, whether indoors or outdoors, covering a limited
11 geographic area, e.g., a building or campus.

12 **3.16 local clock:** A free-running clock, embedded in a respective entity (e.g., PTP Instance, CSN node), that
13 provides a common time to that entity relative to an arbitrary epoch.

14 **3.17 message timestamp point:** A point within an event message serving as a reference point for when a
15 timestamp is taken.

16 **3.18 message type:** The message type of a message is the name of the respective message, e.g., Sync,
17 Announce, Timing Measurement Frame.

18 **3.19 precision:** A measure of the deviation from the mean of the time or frequency error between the clock
19 under test and a reference clock (see IEEE Std 1588-2019).

20 **3.20 primary reference:** A source of time and/or frequency that is traceable to international standards.
21 See also: traceability.

22 **3.20a profile standard:** A standard specifying the capabilities, options, and parameter values of one or
23 more base standards for use in a specific target environment or application.

24 **3.21 PTP End Instance:** A PTP Instance that has exactly one PTP Port.

25 **3.22 PTP Instance:** An instance of the IEEE 802.1AS protocol, operating in a single time-aware system
26 within exactly one domain. A PTP Instance implements the portions of IEEE Std 802.1AS indicated as
27 applicable to either a PTP Relay Instance or a PTP End Instance.

28 NOTE—As used in IEEE Std 802.1AS, the term *PTP Instance* refers to an IEEE 1588 PTP Instance that conforms to the
29 requirements of IEEE Std 802.1AS.

30 **3.23 PTP Link:** Within a domain, a network segment between two PTP Ports using the peer-to-peer delay
31 mechanism of IEEE Std 802.1AS. The peer-to-peer delay mechanism is designed to measure the
32 propagation time over such a link.

33 **3.24 NOTE—A PTP profile standard:** A standard specifying the capabilities, options, and parameter values
34 of one or more base standards for use in a specific target environment or application.o other PTP Ports, using
35 the IEEE 802.1AS protocol.

36 NOTE—A PTP Relay Instance could, for example, be contained in a bridge, a router, or a multi-port end station.

37 **3.25 recognized timing standard:** A recognized standard time source that is a source external to IEEE 1588
38 precision time protocol (PTP) and provides time that is traceable to the international standards laboratories

1 maintaining clocks that form the basis for the International Atomic Time (TAI) and Coordinated Universal
 2 Time (UTC) timescales. Examples of these sources are National Institute of Standards and Technology
 3 (NIST) timeservers and global navigation satellite systems (GNSSs).

4 **3.26 reference plane:** The boundary between a PTP Port of a PTP Instance and the network physical
 5 medium. Timestamp events occur as frames cross this interface.

6 **3.27 residence time:** The duration of the time interval between the receipt of a time-synchronization event
 7 message by a PTP Instance and the sending of the next subsequent time-synchronization event message on
 8 another PTP Port of that PTP Instance. Residence time can be different for different PTP Ports. The term
 9 *residence time* applies only to the case where syncLocked is TRUE.

10 NOTE 1—See 10.2.5.15 for the definition of the variable syncLocked.

11 NOTE 2—If a PTP Port of a PTP Instance sends a time-synchronization event message without having received a time-
 12 synchronization event message, e.g., if syncLocked is FALSE or if sync receipt timeout occurs (see 10.7.3.1), the
 13 duration of the interval between the most recently received time-synchronization event message and the sent time-
 14 synchronization event message is mathematically equivalent to residence time; however, this interval is not normally
 15 called a *residence time*.

16 **3.27a sdoId:** An attribute that is the primary mechanism for providing isolation of PTP Instances operating
 17 under a PTP profile specified by one Qualified Standards Development Organization (QSDO) from PTP
 18 Instances operating under a PTP profile specified by a different QSDO (see 7.1.3 and 7.1.4 of IEEE Std
 19 1588-2019).

20 **3.28 stability (of a clock or clock signal):** A measure of the variations over time of the frequency error (of
 21 the clock or clock signal). The frequency error typically varies with time due to aging and various
 22 environmental effects, e.g., temperature.

23 **3.29 synchronized time:** The time of an event relative to the Grandmaster Clock.

24 NOTE—If there is a change in the Grandmaster PTP Instance or its time base, the *synchronized time* can experience a
 25 phase and/or frequency step.

26 **3.30 synchronized clocks:** Absent relativistic effects, two clocks are synchronized to a specified uncertainty
 27 if they have the same epoch and their measurements of the time of a single event at an arbitrary time differ
 28 by no more than that uncertainty.

29 NOTE—See 8.2.2.

30 **3.31 syntonized clocks:** Absent relativistic effects, two clocks are syntonized to a specified uncertainty if
 31 the duration of a second is the same on both, which means the time as measured by each advances at the
 32 same rate within the specified uncertainty. The two clocks might or might not share the same epoch.

33 **3.32 time-aware system:** A device that contains one or more PTP Instances and/or PTP services (e.g.,
 34 Common Mean Link Delay Service).

35 NOTE 1—See 11.2.17 for a description of the Common Mean Link Delay Service.

36 NOTE 2—A time-aware system can contain more than one PTP Instance in the same domain and/or different domains.

37 **3.32a timescale:** A measure of elapsed time since an epoch.

38 **3.33 timestamp measurement plane:** The plane at which timestamps are captured. If the timestamp
 39 measurement plane is different from the reference plane, the timestamp is corrected for ingressLatency and/
 40 or egressLatency. See: **reference plane**.

41 NOTE—For timestamping on egress and ingress, see 8.4.3.

¹ **3.34 traceability:** See 3.1.81 in IEEE Std 1588-2019.

2

1 4. Acronyms and abbreviations

2 Ack	acknowledgment
3 ADEV	Allan deviation
4 ARB	arbitrary
5 BC	Boundary Clock
6 BTC	best timeTransmitter clock
7 BTCA	best timeTransmitter clock algorithm
8 CID	Company identification (allocated by the IEEE)
9 CMLDS	Common Mean Link Delay Service
10 CSN	coordinated shared network
11 CTC	channel time clock
12 EPON	IEEE 802.3™ Ethernet Passive Optical Network, as specified in IEEE Std 802.3-2018
13 FTM	Fine Timing Measurement
14 G.hn	ITU-T G.9960 and ITU-T G.9961
15 GM	Grandmaster
16 GMT	Greenwich mean time
17 GNSS	global navigation satellite system
18 GPS	global positioning (satellite) system
19 gPTP	generalized precision time protocol (IEEE Std 802.1AS)
20 IERS	International Earth Rotation and Reference Systems Service
21 IP	Internet Protocol
22 ISS	Internal Sublayer Service
23 LAN	local area network
24 LCI	location configuration information
25 LLC	logical link control
26 MAC	media access control
27 MACsec	media access control security

1 MIB	Management Information Base
2 MLME	IEEE 802.11™ MAC layer management entity
3 MPCP	IEEE 802.3 multipoint control protocol
4 MPCPDU	IEEE 802.3 MPCP data unit
5 MD	media-dependent
6 NETCONF	Network Configuration Protocol
7 NMS	Network Management System
8 NTP	network time protocol ¹³
9 OLT	IEEE 802.3 optical line terminal
10 ONU	IEEE 802.3 optical network unit
11 OSSP	organization-specific slow protocol
12 OUI	Organizationally Unique Identifier
13 P2P	peer-to-peer
14 PICS	Protocol Implementation Conformance Statement
15 POSIX®	portable operating system interface (see ISO/IEC 9945:2003 [B17] ¹⁴)
16 PTP	IEEE 1588 precision time protocol
17 PTPDEV	PTP deviation
18 QSDO	Qualified Standards Development Organization
19 RTT	round-trip time
20 SI	international system of units
21 SMI	Structure of Management Information
22 SMIv2	Structure of Management Information version 2
23 SNMP	Simple Network Management Protocol
24 STA	station
25 TAI	International Atomic Time
26 TC	Transparent Clock

¹³ Information available at <https://www.ietf.org/rfc/rfc1305.txt>.

¹⁴ The numbers in brackets correspond to the numbers in the bibliography in Annex H.

1 TDEV	time deviation
2 TDM	time division multiplexing
3 TDMA	time division multiple access
4 TLV	type, length, value
5 TM	Timing Measurement
6 UCT	unconditional transfer
7 UML®	Unified Modeling Language™ ¹⁵
8 UTC	Coordinated Universal Time
9 VLAN	virtual local area network
10 WLAN	wireless local area network

¹⁵ UML® is a registered trademark of the Object Management Group, Inc., and Unified Modeling Language™ is a trademark of the Object Management Group, Inc.

1 5. Conformance

2 This clause specifies the mandatory and optional capabilities provided by conformant implementations of
3 this standard.

4 5.1 Requirements terminology

5 For consistency with existing IEEE and IEEE 802.1 standards, requirements placed upon conformant
6 implementations of this standard are expressed using the following terminology:

- 7 a) ***shall*** is used for mandatory requirements.
- 8 b) ***may*** is used to describe implementation or administrative choices (“may” means “is permitted to,”
9 and hence, “may” and “may not” mean precisely the same thing).
- 10 c) ***should*** is used for recommended choices (the behaviors described by “should” and “should not” are
11 both permissible but not equally desirable choices).

12 The Protocol Implementation Conformance Statement (PICS) proforma (see Annex A) reflects the
13 occurrences of the words shall, may, and should within the standard. The words shall, may, and should, as
14 used in Annex A itself, reflect the use of the PICs and not conformance to the standard.

15 The standard avoids needless repetition and apparent duplication of its formal requirements by using ***is***, ***is not***, ***are***, and ***are not*** for definitions and the logical consequences of conformant behavior. Behavior that is
16 permitted but is neither always required nor directly controlled by an implementer or administrator, or
17 whose conformance requirement is detailed elsewhere, is described by ***can***. Behavior that never occurs in a
18 conformant implementation or system of conformant implementations is described by ***cannot***. The word
20 ***allow*** is used as a replacement for the phrase “support the ability for,” and the word ***capability*** means “can
21 be configured to.”

22 5.2 Protocol Implementation Conformance Statement (PICS)

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

26 5.3 Time-aware system requirements

27 An implementation of a time-aware system shall support at least one IEEE 1588 precision time protocol
28 (PTP) Instance.

29 5.4 PTP Instance requirements and options

30 5.4.1 PTP Instance requirements

31 An implementation of a PTP Instance shall:

- 32 a) Implement the generalized precision time protocol (gPTP) requirements specified in Clause 8.
- 33 b) Support the requirements for time-synchronization state machines (10.1.2, 10.2.1, 10.2.2, 10.2.3,
34 10.2.4, 10.2.5, and 10.2.6).
- 35 c) Support at least one PTP Port.
- 36 d) On each supported PTP Port, implement the PortSyncSyncReceive state machine (10.2.8).

- 1 e) Implement the ClockTimeReceiverSync state machine (10.2.13).
- 2 f) Implement the SiteSyncSync state machine (10.2.7).
- 3 g) Implement the state machines related to signaling gPTP capability (10.4).

4 NOTE 1—5.4.1 g) does not include the GptpCapableIntervalSetting state machine (10.4.3), despite its name
5 containing “GptpCapable”.

6 NOTE 2—The GptpCapableTransmit and GptpCapableReceive state machines are required; however, they can
7 be disabled by setting the managed object gptpCapableStateMachinesEnabled (see 14.8.54a) to FALSE.

- 8 h) For receipt of all messages and for transmission of all messages except Announce (see 10.6.3),
9 support the message requirements as specified in 10.5, 10.6, and 10.7.

10 NOTE 3—The support of Signaling messages is mandatory [5.4.1 h)] because the state machines related to the
11 signaling of gPTP capability are mandatory [5.4.1 g)].

- 12 i) Support the performance requirements in B.1 and B.2.4.

13 5.4.2 PTP Instance options

14 An implementation of a PTP Instance should:

- 15 a) Support the performance requirements in B.2.2 and B.2.3.

16 An implementation of a PTP Instance may:

- 17 b) Support the following media-independent timeTransmitter capability on at least one PTP Port:
 - 18 1) Implement the PortSyncSyncSend state machine (10.2.12).
 - 19 2) Implement the PortAnnounceTransmit state machine (10.3.16).
 - 20 3) Implement the AnnounceIntervalSetting state machine (10.3.17).
 - 21 4) For transmit of the Announce message, support the message requirements as specified in 10.5,
22 10.6, and 10.7.
- 23 c) Support the following for Grandmaster PTP Instance capability:
 - 24 1) Support the media-independent timeTransmitter capability specified in item b) of 5.4.2.
 - 25 2) Support the requirements for a grandmaster-capable PTP Instance (10.1.3).
 - 26 3) Implement the ClockTimeTransmitterSyncSend state machine (10.2.9).
 - 27 4) Implement the ClockSyncOffset state machine (10.2.10).
 - 28 5) Implement the ClockTimeTransmitterSyncReceive state machine (10.2.11).
- 29 d) Support more than one PTP Port as a PTP Relay Instance (5.4.3).
- 30 e) Support more than one PTP Instance; such support allows for more than one domain (7.2.3).
- 31 f) Support the following external port configuration capability on at least one port:
 - 32 1) Implement specifications for externalPortConfigurationEnabled value of true (10.3.1).
 - 33 2) Implement the PortAnnounceInformationExt state machine (10.3.14).
 - 34 3) Implement the PortStateSettingExt state machine (10.3.15).
- 35 g) Implement the SyncIntervalSetting state machine (10.3.18).
- 36 h) Implement the GptpCapableIntervalSetting state machine.
- 37 i) Implement one or more of the application interfaces specified in Clause 9; A PTP Instance that
38 claims to support application interfaces shall state which application interfaces are supported.
- 39 j) Support timing and synchronization management as specified in Clause 14.

- 1 k) Support the use of a remote management protocol. A PTP Instance that claims to support remote
2 management shall:
 - 3 1) State which remote management protocol standard(s) or specification(s) are supported
4 (see A.19).
 - 5 2) State which standard(s) or specification(s) for managed object definitions and encodings are
6 supported for use by the remote management protocol (see A.19).
 - 7 3) If the Simple Network Management Protocol (SNMP) is supported as a remote management
8 protocol, support the managed object definitions specified as Structure of Management
9 Information version 2 (SMIV2) Management Information Base (MIB) modules in Clause 15.
 - 10 4) If YANG is supported with a remote management protocol, support the YANG data model
11 ieee802-dot1as-gptp in Clause 17.
 - 12 5) If YANG is supported with a remote management protocol, and if hot standby is supported,
13 support the YANG data model ieee802-dot1as-hs in Clause 17.
 - 14 l) Implement both BTCA and external port configuration on domains other than domain 0; if both
15 possibilities are implemented on domains other than domain 0, the default value of
16 externalPortConfigurationEnabled shall be FALSE.
 - 17 m) Implement hot standby as specified in Clause 18, 14.7a, 9.3.3.4, 9.4.3.4, 9.5.3.4, and 9.6.2.6.
 - 18 n) Implement the Drift_Tracking TLV as specified in 10.2.4.26, 10.2.4.27, 10.2.4.28, 11.2.14, 11.2.15,
19 and 11.4.4.

20 5.4.2a PTP Instance defaults and recommendations

21 This standard identifies specific capabilities and parameter values as defaults. If these defaults are not
22 further qualified by reference to the use of other capabilities or by profile standards, the defaults are intended
23 to facilitate deployment and to provide interoperability in target environments and applications with
24 time-aware system implementations conformant to this standard (including its 2011 edition). Mandatory
25 requirements that this standard does not identify as defaults, or call out in some other way, are not subsetted
26 or modified by profile standards.

27 Changes to capability support or parameter defaults arising from conformance to a profile standard should
28 be identified in the PICS for that profile standard. Changes to defaults by pre-configuration of parameter
29 values by the supplier of a protocol implementation should be identified by Additional Information
30 (see A.3.1) in a completed PICS for this standard.

31 An implementation of a PTP Instance shall, by default, support the following best timeTransmitter clock
32 algorithm (BTCA) requirements:

- 33 a) Implement the BTCA (10.3.1.1, 10.3.1.2, 10.3.2, 10.3.3, 10.3.4, 10.3.5, 10.3.6, 10.3.8, and 10.3.10).
- 34 b) For domain 0, implement specifications for an externalPortConfigurationEnabled value of FALSE
35 (10.3.1).
- 36 c) Implement the PortAnnounceReceive state machine (10.3.11).
- 37 d) Implement the PortAnnounceInformation state machine (10.3.12).
- 38 e) Implement the PortStateSelection state machine (10.3.13).
- 39 f) Have the BTCA as the default mode of operation, with externalPortConfigurationEnabled FALSE,
40 on domain 0.
- 41 g) Implement at least one of the possibilities for externalPortConfigurationEnabled (i.e., FALSE,
42 meaning the BTCA is used, and TRUE, meaning external port configuration is used) on domains
43 other than domain 0.

1 5.4.3 PTP Relay Instance requirements

2 An implementation of a PTP Relay Instance shall:

- 3 a) Support more than one PTP Port.
- 4 b) Support the media-independent timeTransmitter capability specified in item b) of 5.4.2.

5 5.5 .MAC-specific timing and synchronization methods for full-duplex IEEE 802.3 links

7 An implementation of a time-aware system with IEEE 802.3 media access control (MAC) services to 8 physical ports shall:

- 9 a) Support full-duplex operation, as specified in 4.2 and Annex 4A of IEEE Std 802.3-2018.
- 10 b) Support the requirements as specified in Clause 11 [with the exception of requirements more 11 specifically addressed in this subclause (5.5)].
- 12 c) **Support two-step capability on receive as specified in 11.2.14.**
- 13 d) **Support two-step capability on transmit as specified in 11.2.15**
- 14 e) Provide the Common Mean Link Delay Service (CMLDS) if the time-aware system implements 15 more than one domain, as specified in 11.2.17 and 11.2.18.

16 An implementation of a PTP Instance with IEEE 802.3 MAC services to physical ports may:

- 17 f) Support asymmetry measurement mode as specified in 10.3.12, 10.3.13, 10.3.16, 11.2.14, 11.2.15, 11.2.19, and 14.8.45.
- 19 g) Support one-step capability on receive as specified in 11.2.14.
- 20 h) Support one-step capability on transmit as specified in 11.2.15.
- 21 i) Support the OneStepTxOperSetting state machine specified in 11.2.16.
- 22 j) Support propagation delay averaging, as specified in 11.2.19.3.4.
- 23 k) Provide the Common Mean Link Delay Service (CMLDS) if the time-aware system implements 24 only one domain, as specified in 11.2.17 and 11.2.18.
- 25 l) **Implement the LinkDelayIntervalSetting state machine (11.2.21).**

26 5.6 MAC-specific timing and synchronization methods for IEEE Std 802.11-2016

27 An implementation of a time-aware system with IEEE 802.11 MAC services to physical ports shall:

- 28 a) Support the requirements as specified in Clause 12.
- 29 b) Support at least one of
 - 30 1) the media-dependent timeTransmitter state machines (12.5.1), or
 - 31 2) the media-dependent timeReceiver state machine (12.5.2).

32 An implementation of a PTP End Instance with IEEE 802.11 MAC services to physical ports shall:

- 33 c) Support at least one of TIMINGMSMT as specified in IEEE Std 802.11-2016 or 34 FINETIMINGMSMT as specified in IEEE Std 802.11-2016.

35 An implementation of a PTP Relay Instance with IEEE 802.11 MAC services to physical ports shall:

- 1 d) Support the requirements of TIMINGMSMT as specified in IEEE Std 802.11-2016.
- 2 An implementation of a PTP Relay Instance with IEEE 802.11 MAC services to physical ports shall:
- 3 e) Support FINETIMINGMSMT as specified in IEEE Std 802.11-2016.

4 NOTE—In order to maintain backward compatibility with existing TM-based PTP End Instances that support only
5 Timing Measurement (TM), the PTP Relay Instance is required to support TM. PTP End Instances are allowed to
6 support TM or Fine Timing Measurement (FTM), or both to permit PTP End Instances compliant with this standard to
7 implement only the FTM standard, which requires a PTP Relay Instance that supports FTM.

8 **5.7 MAC-specific timing and synchronization methods for IEEE 802.3 EPON**

9 An implementation of a time-aware system with IEEE 802.3 Ethernet Passive Optical Network (EPON)
10 MAC services to physical ports shall:

- 11 a) Support the requirements as specified in IEEE Std 802.3-2018 for multipoint MAC Control (64.2 and 64.3) and multipoint physical coding sublayer (PCS) and physical medium attachment (PMA) extensions (Clause 65).
- 14 b) Support the requirements as specified in Clause 13.

15 **5.8 MAC-specific timing and synchronization methods for coordinated shared 16 network (CSN)**

17 An implementation of a time-aware system with CSN MAC services to physical ports shall:

- 18 a) Support the requirements as specified in Clause 16.
- 19 b) Support at least one MoCA port (16.6.2) or ITU-T G.hn port (16.6.3).

1 6. Conventions

2 6.1 General

3 This clause defines various conventions and notation used in the standard, i.e., naming conventions, service
 4 specification method and notation, and data type definitions.

5 6.2 Service specification method and notation

6 The method and notation for specifying service interfaces is described in Clause 7 of IEEE Std 802.1AC-
 7 2016.

8 6.3 Lexical form syntax

9 A lexical form refers to the following:

- 10 — A name
- 11 — A data type

12 The conventions illustrated in the following list regarding lexical forms are used in this standard:

- 13 a) Type names: e.g., ClockQuality (no word separation, initial letter of each word capitalized).
- 14 b) Enumeration members and global constants: e.g., ATOMIC_CLOCK (word separation underscored,
 15 all letters capitalized).
- 16 c) Fields within PTP messages, instances of structures, and variables: e.g., secondsField, clockQuality,
 17 clockIdentity (two-word field names at a minimum, no word separation, initial word not capitalized,
 18 initial letter capitalization on subsequent words).
- 19 d) Members of a structure: e.g., clockQuality.clockClass (no word separation, structure name followed
 20 by a period followed by the member name).
- 21 e) Data set names: e.g., defaultDS, parentDS, portDS, currentDS, timePropertiesDS (no word
 22 separation, initial word not capitalized, initial letter capitalization on subsequent words, end marked
 23 by the letters DS).
- 24 f) Data set members: e.g., defaultDS.clockQuality.clockClass (data set name followed by a period
 25 followed by a member name followed by a period followed by a member name).
- 26 g) PTP message names: e.g., Sync, Pdelay_Req (word separation underscored, initial letter of each
 27 word capitalized).

28 When a lexical form appears in text, as opposed to in a type, or a format definition, the form is to be
 29 interpreted as singular, plural, or possessive as appropriate to the context of the text.

30 6.4 Data types and on-the-wire formats

31 6.4.1 General

32 The data types specified for the various variables and message fields define logical properties that are
 33 necessary for correct operation of the protocol or interpretation of IEEE 1588 precision time protocol (PTP)
 34 or IEEE 802.11 message content.

35 NOTE—Implementations are free to use any internal representation of data types if the internal representation does not
 36 change the semantics of any quantity visible via communications using the IEEE 802.1AS protocol or in the specified
 37 operations of the protocol.

1 6.4.2 Primitive data types specifications

2 All non-primitive data types are derived from the primitive types in Table 6-1. Signed integers are
 3 represented in two's complement form.

Table 6-1—Primitive data types

Data type	Definition
Boolean	TRUE or FALSE
EnumerationN	N-bit enumerated value
UIntegerN	N-bit unsigned integer
IntegerN	N-bit signed integer
Nibble	4-bit field not interpreted as a number
Octet	8-bit field not interpreted as a number
OctetN	N-octet field not interpreted as a number, with N > 1
Float64 (see NOTE)	IEEE 754™ binary64 (64-bit double-precision floating-point format)
NOTE—The Float64 data type was called Double in the 2011 edition of this standard. The semantics of the data type has not changed.	

4 6.4.3 Derived data type specifications

5 6.4.3.1 ScaledNs

6 The ScaledNs type represents signed values of time and time interval in units of 2^{-16} ns.

7 `typedef Integer96 ScaledNs;`

8 For example: -2.5 ns is expressed as:

9 `0xFFFF FFFF FFFF FFFF FFFD 8000`

10 Positive or negative values of time or time interval outside the maximum range of this data type are encoded
 11 as the largest positive or negative value of the data type, respectively.

12 6.4.3.2 UScaledNs

13 The UScaledNs type represents unsigned values of time and time interval in units of 2^{-16} ns.

14 `typedef UInteger96 UScaledNs;`

15 For example: 2.5 ns is expressed as:

16 `0x0000 0000 0000 0000 0002 8000`

17 Values of time or time interval greater than the maximum value of this data type are encoded as the largest
 18 positive value of the data type.

1 **6.4.3.3 TimeInterval**

2 The TimeInterval type represents time intervals, in units of 2^{-16} ns.

3 `typedef Integer64 TimeInterval;`

4 For example: 2.5 ns is expressed as:

5 `0x0000 0000 0002 8000`

6 Positive or negative time intervals outside the maximum range of this data type are encoded as the largest
7 positive and negative values of the data type, respectively.

8 **6.4.3.4 Timestamp**

9 The Timestamp type represents a positive time with respect to the epoch.

```
10 struct Timestamp
11 {
12     UIInteger48 seconds;
13     UIInteger32 nanoseconds;
14 };
```

15 The seconds member is the integer portion of the timestamp in units of seconds.

16 The nanoseconds member is the fractional portion of the timestamp in units of nanoseconds.

17 The nanoseconds member is always less than 10^9 .

18 For example:

19 `+2.000000001` seconds is represented by seconds = `0x0000 0000 0002` and nanoseconds= `0x0000 0001`

20 **6.4.3.5 ExtendedTimestamp**

21 The ExtendedTimestamp type represents a positive time with respect to the epoch.

```
22 struct ExtendedTimestamp
23 {
24     UIInteger48 seconds;
25     UIInteger48 fractionalNanoseconds;
26 };
```

27 The seconds member is the integer portion of the timestamp in units of seconds.

28 The fractionalNanoseconds member is the fractional portion of the timestamp in units of 2^{-16} ns.

29 The fractionalNanoseconds member is always less than $(2^{16})(10^9)$.

30 For example:

31 `+2.000000001` seconds is represented by seconds = `0x0000 0000 0002` and fractionalNanoseconds =
32 `0x0000 0001 0000`

1 **6.4.3.6 ClockIdentity**

2 The ClockIdentity type identifies a PTP Instance.

3 `typedef Octet8 ClockIdentity;`

4 **6.4.3.7 PortIdentity**

5 The PortIdentity type identifies a port of a PTP Instance.

```
6 struct PortIdentity
7 {
8     ClockIdentity clockIdentity;
9     UIInteger16 portNumber;
10};
```

11 **6.4.3.8 ClockQuality**

12 The ClockQuality represents the quality of a clock.

```
13 struct ClockQuality
14 {
15     UIInteger8 clockClass;
16     Enumeration8 clockAccuracy;
17     UIInteger16 offsetScaledLogVariance;
18};
```

19 **6.4.4 Protocol data unit (PDU) formats**

20 **6.4.4.1 General**

21 The data types defined in 6.4.2 and 6.4.3 shall be mapped onto the wire according to the mapping rules for
22 the respective medium, e.g., IEEE Std 802.3-2018 and IEEE Std 802.11-2016, and the terms of 6.4.4.

23 IEEE 802.1AS PDUs consist of the messages defined or referenced in Clause 10, Clause 11, Clause 12, and
24 Clause 13, based on the data types defined in 6.4.2 and 6.4.3. The internal ordering of the fields of the
25 IEEE 802.1AS PDUs is specified in 6.4.4.3 to 6.4.4.5.

26 **6.4.4.2 Numbering of bits within an octet**

27 Bits are numbered with the most significant bit being 7 and the least significant bit being 0.

28 NOTE—The numbering and ordering of bits within an octet of a PDU, described here, is independent of and unrelated to
29 the order of transmission of the bits on the underlying physical layer.

30 **6.4.4.3 Primitive data types**

31 Numeric primitive data types defined in 6.4.2 shall be formatted with the most significant octet nearest to the
32 beginning of the PDU followed in order by octets of decreasing significance.

33 The Boolean data type TRUE shall be formatted as a single bit equal to 1 and FALSE as a single bit equal
34 to 0.

1 Enumerations of whatever length shall be formatted as though the assigned values are unsigned integers of
2 the same length, e.g., Enumeration16 shall be formatted as though the value had type UInteger16.

3 **6.4.4.4 Arrays of primitive types**

4 All arrays shall be formatted with the member having the lowest numerical index nearest to the beginning of
5 the PDU followed by successively higher numbered members, without any padding. In octet arrays, the
6 octet with the lowest numerical index is termed the most significant octet.

7 When a field containing more than one octet is used to represent a numeric value, the most significant octet
8 shall be nearest to the beginning of the PDU, followed by successively less significant octets.

9 When a single octet contains multiple fields of primitive data types, the bit positions within the octet of each
10 of the primitive types as defined in the message field specification shall be preserved. For example, the first
11 field of the header of PTP messages is a single octet composed of two fields, one of type Nibble bit 4
12 through bit 7, and one of type Enumeration4 bit 0 through bit 3 (see 11.4.2 and 10.6.2).

13 **6.4.4.5 Derived data types**

14 Derived data types defined as structs shall be formatted with the first member of the struct nearest to the
15 beginning of the PDU followed by each succeeding member, without any padding. Each member shall be
16 formatted according to its data type.

17 Derived data types defined as typedefs shall be formatted according to its referenced data type.

1 7. Time-synchronization model for a packet network

2 7.1 General

3 This clause provides a model for understanding the operation of the generalized precision time protocol
4 (gPTP), which specifies the operation of time-aware systems on a packet network. Although this standard is
5 based on the precision time protocol (PTP) described in IEEE Std 1588-2019 (and, indeed, is a proper
6 profile of IEEE Std 1588-2019 in particular configurations), there are differences, which are summarized
7 in 7.5.

8 Although this standard has been written as a stand-alone document, it is useful to understand the IEEE 1588
9 architecture as described in Clause 6 of IEEE Std 1588-2019.

10 7.2 Architecture of a time-aware network

11 7.2.1 General

12 A time-aware network consists of a number of interconnected time-aware systems that support the gPTP
13 defined within this standard. These time-aware systems can be any networking device, including, for
14 example, bridges, routers, and end stations. A set of time-aware systems that are interconnected by gPTP-
15 capable network elements is called a *gPTP network*. Each instance of gPTP that the time-aware systems
16 support is in one *gPTP domain*, and the instances of gPTP are said to be part of that gPTP domain. A time-
17 aware system can support, and therefore be part of, more than one gPTP domain. The entity of a single time-
18 aware system that executes gPTP in one gPTP domain is called a *PTP Instance*. A time-aware system can
19 contain multiple PTP Instances, which are each associated with a different gPTP domain. There are two
20 types of PTP Instances:

- 21 a) PTP End Instance, which, if not a Grandmaster PTP Instance, is a recipient of time information, and
22 b) PTP Relay Instance, which, if not a Grandmaster PTP Instance, receives time information from the
23 Grandmaster PTP Instance (perhaps indirectly through other PTP Relay Instances), applies
24 corrections to compensate for delays in the local area network (LAN) and the PTP Relay Instance
25 itself, and retransmits the corrected information.

26 This standard defines mechanisms for delay measurements using standard-based procedures for the
27 following:

- 28 c) IEEE 802.3 Ethernet using full-duplex point-to-point links (11)
29 d) IEEE 802.3 EPON links (Clause 13)
30 e) IEEE 802.11 wireless (Clause 12)
31 f) Generic coordinated shared networks (CSNs, e.g., MoCA and G.hn) (Clause 16)

32 This standard specifies time distribution mechanisms that are tolerant to some faults if the network paths are
33 redundant. One of these time distribution mechanisms updates the distribution path in reaction to changes
34 (see 10.3.1.2), whereas the other one relies on redundant PTP instances on top of the redundant network
35 paths (see Clause 18).

36 7.2.2 Time-aware network consisting of a single gPTP domain

37 Figure 7-1 illustrates an example time-aware network consisting of a single gPTP domain, using all the
38 above network technologies [i.e., item c) through item f) of 7.2.1], where end stations on several local

1 networks are connected to a Grandmaster PTP Instance on a backbone network via an EPON access network.:
 2

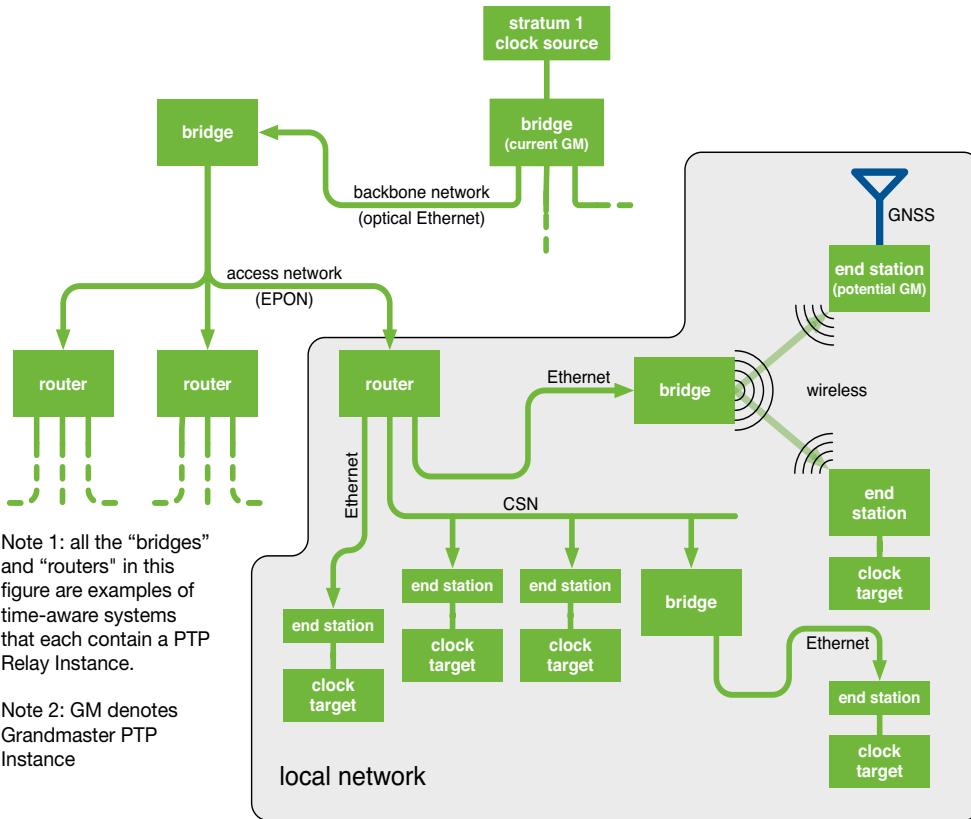


Figure 7-1—Time-aware network example

3 Any PTP Instance with clock sourcing capabilities can be a potential Grandmaster PTP Instance, and a
 4 selection method (the *best timeTransmitter clock algorithm*, or BTCA) ensures that all of the PTP Instances
 5 in *the* gPTP domain use the same Grandmaster PTP Instance.¹⁶ The BTCA is largely identical to that used in
 6 IEEE Std 1588-2019, but somewhat simplified. In Figure 7-1 the BTCA process has resulted in the
 7 Grandmaster PTP Instance being on the network backbone. If, however, the access network fails, the
 8 systems on a local network automatically switch over to one of the potential Grandmaster PTP Instances on
 9 the local network that is as least as “good” as any other. For example, in Figure 7-2, the access network link
 10 has failed, and a potential Grandmaster PTP Instance that has a GNSS reference source has become the
 11 active Grandmaster PTP Instance. As a result, now two gPTP domains exist where there used to be one..

12

¹⁶ There are, however, short periods during network reconfiguration when more than one Grandmaster PTP Instance might be active while the BTCA process is taking place.

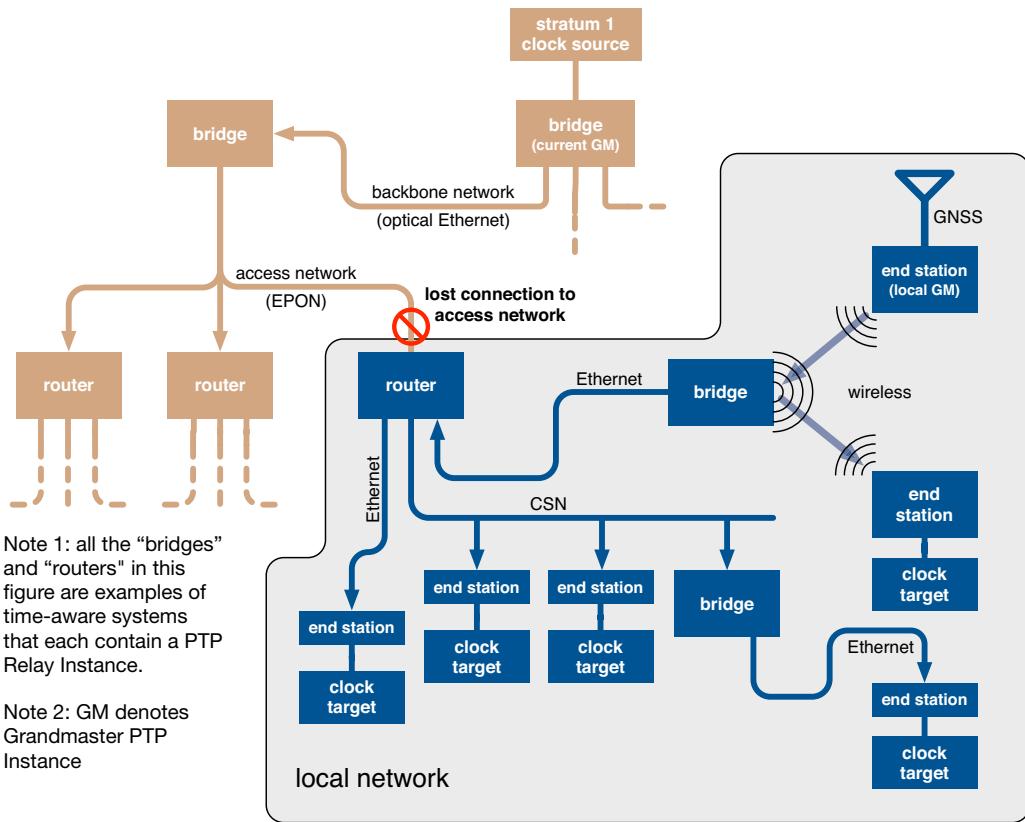


Figure 7-2—Time-aware network of Figure 7-1 after an access network link failure

1 7.2.3 Time-aware network consisting of multiple gPTP domains

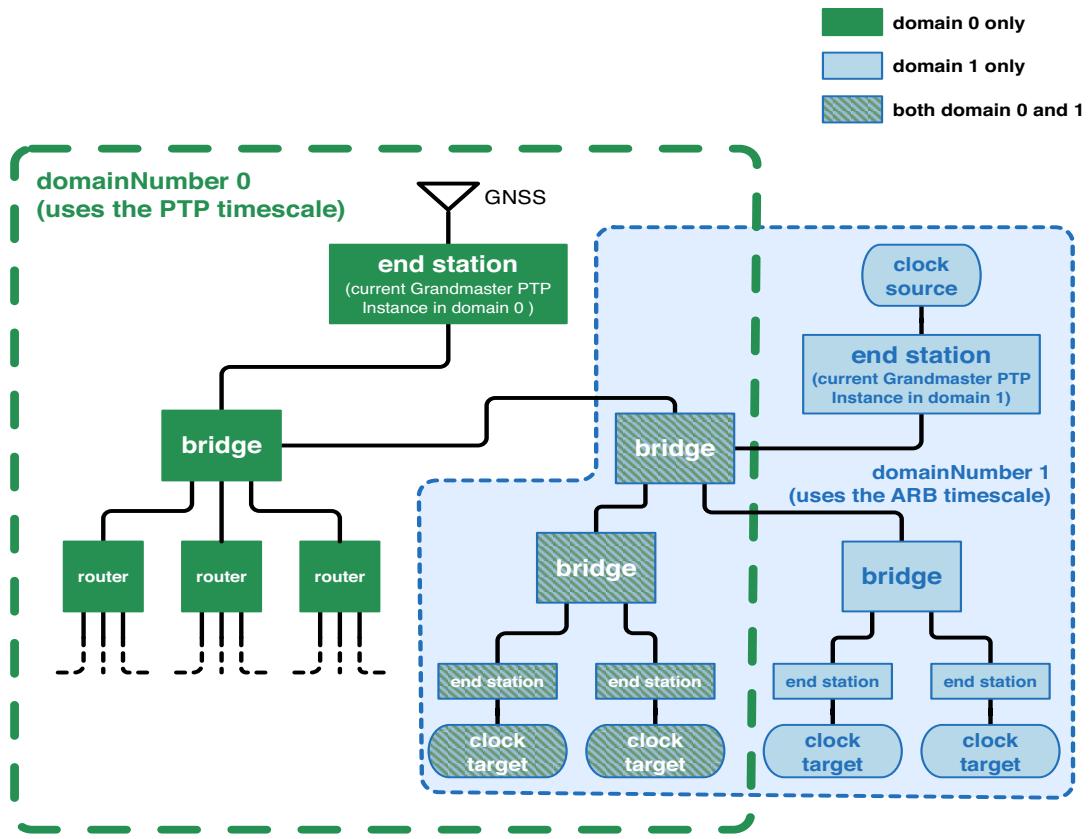
2 Figure illustrates an example time-aware network consisting of multiple gPTP domains that could be used
 3 in an industrial application. Specifically, in this example the network has two timescales/domains, where
 4 domain 0 uses the PTP timescale and domain 1 uses the arbitrary (ARB) timescale (see 8.2). Notice that not
 5 all PTP Instances in domain 1 (within the blue shorter-dashed area) have domain 0 active in this example,
 6 even though every time-aware system supports domain 0 for backward compatibility with the 2011 edition
 7 of this standard. In addition, it is required that all PTP Instances belonging to the same domain have direct
 8 connections among them in their physical topology (e.g., time cannot be transported from one PTP Instance
 9 in domain 0 to another PTP Instance in domain 0 via a time-aware system that does not have domain 0
 10 active). In addition, the time-aware systems for which both domains are active are depicted by slanted
 11 internal hatching, representing two independent, active PTP Instances.

12 As in the single-domain case, any of the network technologies of 7.2.1 can be used. The Grandmaster PTP
 13 Instance of each domain is selected by the BTCA; in this case, a separate, independent instance of the BTCA
 14 is invoked in each domain.

15

16

17



NOTE—All the “bridges” and “routers” in this figure are examples of time-aware systems that contain at least one PTP Relay Instance, and the end stations are time-aware systems that contain at least one PTP End Instance. The PTP Links in this figure can use any of the media specified in this standard.

Figure 7-3—Time-aware network example for multiple gPTP domains

1 7.2.4 Time-aware networks using BTCA

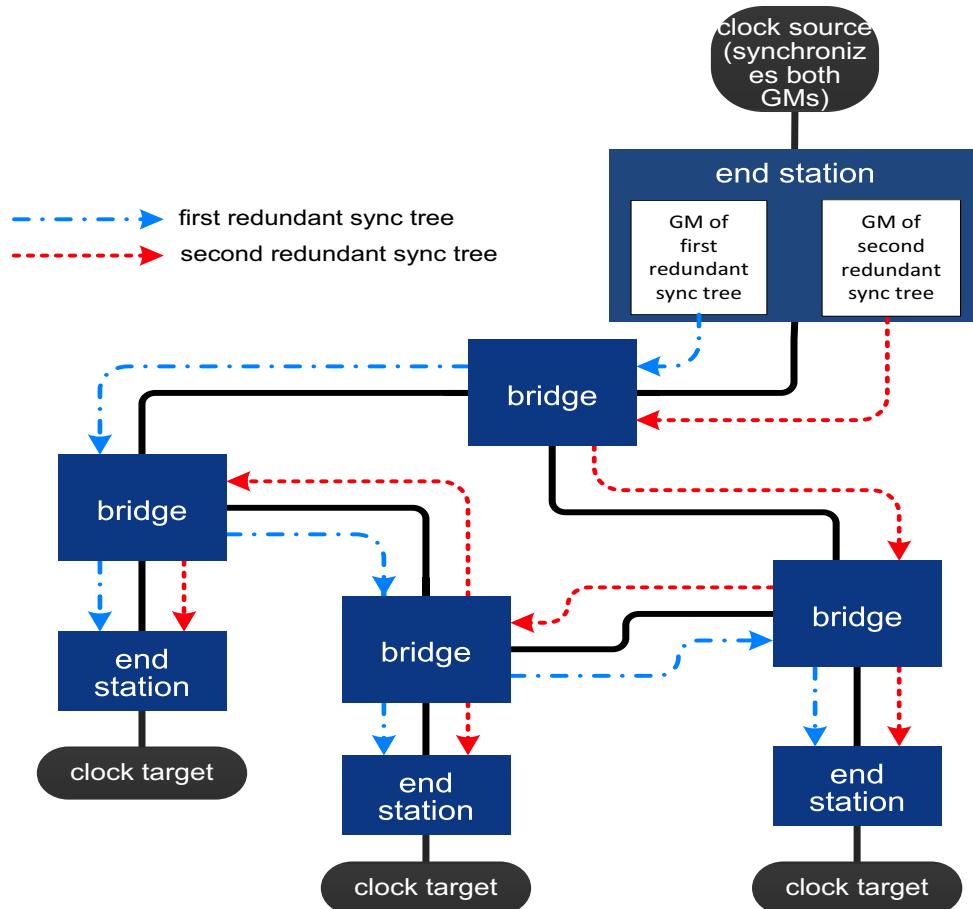
2 This standard provides a basic level of redundancy as follows:

- 3 — A detection component that triggers when no Sync or Announce messages are received for a defined period of time.
- 5 — A correction component that triggers the best timeTransmitter clock algorithm (BTCA) and the sending of Announce messages so that a new Grandmaster PTP Instance can be elected.
- 7 — An action component, where the winning Grandmaster PTP Instance starts sending Announce messages and Sync messages.

9 In addition to providing the basic level of redundancy, this standard provides the ability to support more sophisticated network configurations that provide additional levels of Grandmaster PTP Instance and clock path redundancy. Figure 7-4 and Figure 7-5 are examples of such networks that provide these additional levels of redundancy as a way to deal with these failures. The information necessary to implement and configure these network configurations is contained in this standard.

1 In order to take advantage of these failure correction configurations, new types of fault detection are
 2 required. The category of fault detection where a Grandmaster PTP Instance completely fails and stops
 3 sending clock information is supported as mentioned above.

4 Figure 7-4 shows an example network realizing two redundant synchronization trees from a single GM, each
 5 with its own GM. The two GMs receive timing from the single clock source.



NOTE 1—The methods used for merging the redundant Sync messages received at each end station are not specified in this standard.

NOTE 2—All the “bridges” in this figure are examples of time-aware systems that contain PTP Relay Instances, and the end stations are examples of time-aware systems that contain PTP End Instances.

NOTE 3—GM denotes Grandmaster PTP Instance.

NOTE 4—This figure differs from the 2020 edition of this standard, with IEEE Std 802.1AS-2020/Cor 1-2021, IEEE Std 802.1ASdr-2024, and IEEE Std 802.1ASdn-2024 applied, in that the end station at the top of the figure explicitly shows the two GM PTP Instances and the clock source is indicated as synchronizing both GMs.

Figure 7-4—Time-aware network example for synchronization path redundancy, with one clock source providing time to two domains

1 Figure 7-5 shows an example network with two redundant GMs, one as primary GM and the other as
 2 secondary GM, where each GM has one of the two redundant synchronization trees originating from it. This
 3 example supports hot-standby operating mode. In this mode, the secondary GM has to be synchronized to
 4 the primary GM, because it is part of the synchronization tree of the primary GM as shown in the figure. The
 5 secondary GM does not start sending Sync messages until it is synchronized to the primary GM:

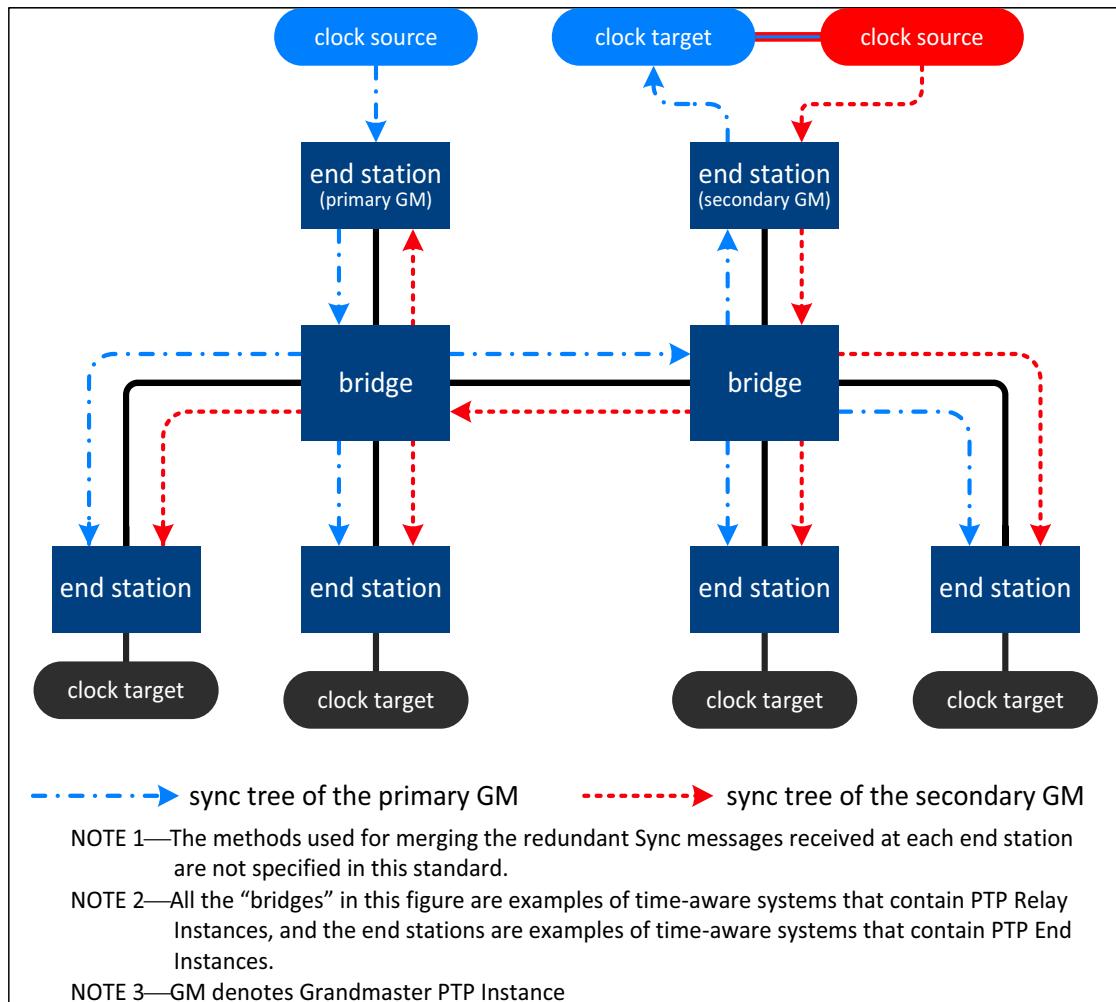


Figure 7-5—Time-aware network example for GM redundancy with one primary GM and one hot-standby GM, which are separated in two gPTP domains

6 7.2.5 Time-aware network with hot standby

7 Figure 7-6 shows an example of a time-aware network with hot standby (see Clause 18). The network has
 8 GM redundancy and path redundancy to all of the Bridges and end stations 1, 4, and 6 (path redundancy for
 9 end stations 2, 3, and 5 is possible, but is not illustrated to avoid clutter in the figure). In addition, for
 10 simplicity the optional split functionality (see 18.5.3.4) is assumed to be disabled.

11 End station 1 is the primary GM (see Clause 18), and its synchronization spanning tree is illustrated by the
 12 arrows of the primary grandmaster sync tree. End station 4 is the secondary GM, and its synchronization
 13 spanning tree is illustrated by the arrows of the secondary grandmaster sync tree. Except for the links to end
 14 stations 2, 3, and 5, the network can tolerate the failure of any single link and the Bridges and end stations

1 remain synchronized by either the primary or the secondary GM. For example, if the link between bridge 3
 2 and end station 6 fails, end station 6 is still synchronized by the primary domain via bridge 2. If the link
 3 between bridge 5 and bridge 6 fails, bridge 6 is still synchronized by the primary domain and bridge 5 is
 4 synchronized by the secondary domain. If the secondary GM fails, all the Bridges and end stations are still
 5 synchronized by the primary GM.

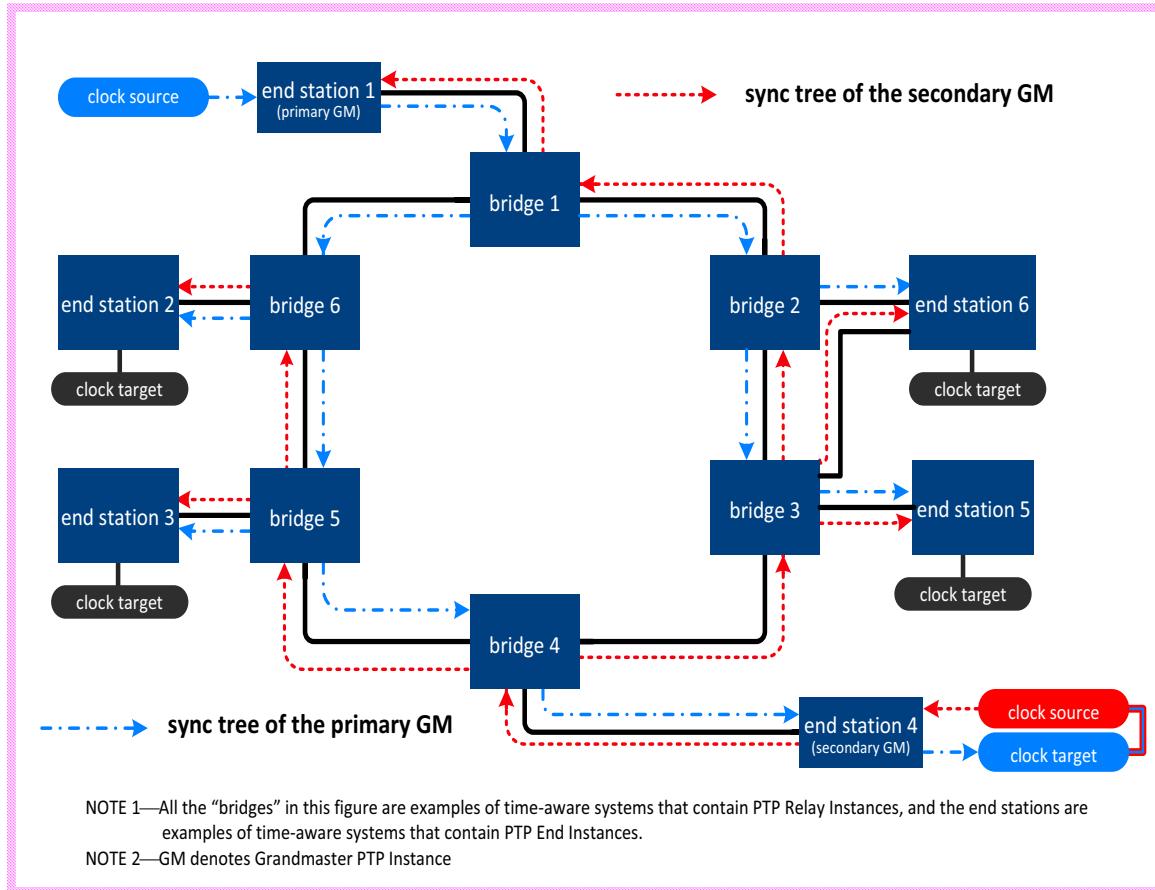


Figure 7-6—Time-aware network example for hot standby with both GM and partial path redundancy

6 If the primary GM fails, all the Bridges and end stations are still synchronized by the secondary GM. In this
 7 case, if syncLocked is TRUE the primary PTP Instance of end station 4 stops receiving Sync messages and
 8 isSynced changes to FALSE; the hotStandbySystemState of end station 4 is then changed to
 9 NOT_REDUNDANT (see Clause 18). If syncLocked is FALSE, and assuming the Bridges and end stations
 10 other than the GMs are not grandmaster capable, the time difference between the primary and secondary
 11 domain eventually exceeds the primarySecondaryOffsetThresh (see Clause 18) and the
 12 hotStandbySystemState of end station 4 changes to NOT_REDUNDANT. Irrespective of the syncLocked
 13 state, the secondary GM no longer takes timing from the primary domain.

14 If the link between bridge 4 and bridge 5 fails, all the PTP Instances except for bridge 4 and end station 4 are
 15 then synchronized by the primary domain. Bridge 4 and end station 4 are synchronized by the secondary
 16 domain; in addition, bridges 1, 2, and 3, and end stations 1, 4, 5, and 6 receive timing on the secondary
 17 domain. Since end station 4 does not receive timing on the primary domain, its hotStandbySystemState
 18 changes to NOT_REDUNDANT, and the secondary domain GM does not take timing from the primary
 19 domain.

1 7.3 Time synchronization

2 7.3.1 General

3 Time synchronization in gPTP is done the same way (in the abstract) as is done in IEEE Std 1588-2019: a
 4 Grandmaster PTP Instance sends information including the current synchronized time to all directly attached
 5 PTP Instances. Each of these PTP Instances must correct the received synchronized time by adding the
 6 propagation time needed for the information to transit the gPTP communication path from the Grandmaster
 7 PTP Instance. If the PTP Instance is a PTP Relay Instance, then it must forward the corrected time
 8 information (including additional corrections for delays in the forwarding process) to all the other attached
 9 PTP Instances.

10 To make this all work, there are two time intervals that must be precisely known: the forwarding delay
 11 (called the *residence time*), and the time taken for the synchronized time information to transit the gPTP
 12 communication path between two PTP Instances. The residence time measurement is local to a PTP Relay
 13 Instance and easy to compute, while the gPTP communication path delay is dependent on many things
 14 including media-dependent properties and the length of the path.

15 7.3.2 Delay measurement

16 Each type of LAN or gPTP communication path has different methods for measuring propagation time, but
 17 they are all based on the same principle: measuring the time that a well-known part of a message is
 18 transmitted from one device and the time that the same part of the same message is received by the other
 19 device, then sending another message in the opposite direction and doing the same measurement as shown in
 20 Figure 7-7.

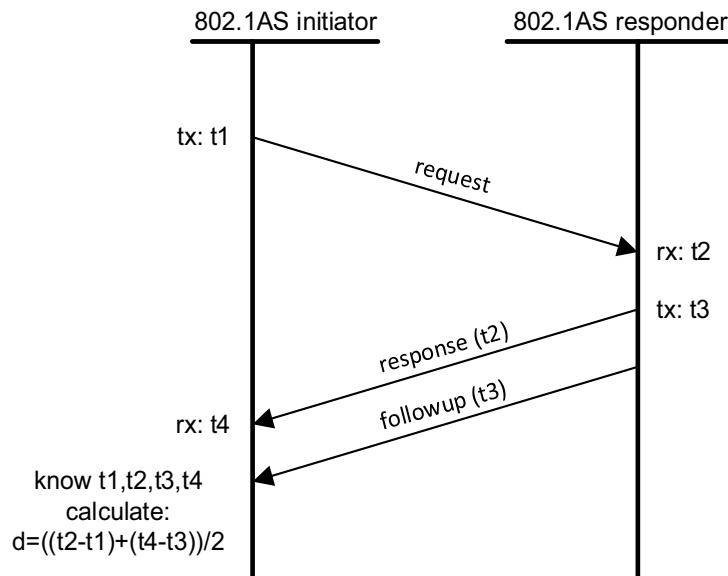


Figure 7-7—Conceptual medium delay measurement

21 This basic mechanism is used in the various LANs in the following ways:

22 a) Full-duplex Ethernet LANs use the two-step peer-to-peer (P2P) path delay algorithm as defined in
 23 IEEE Std 1588-2019, where the messages are called Pdelay_Req, Pdelay_Resp, and
 24 Pdelay_Resp_Follow_Up (see Figure 11-1).

- 1 b) IEEE 802.11 wireless LANs use the Timing Measurement (TM) procedure or the Fine Timing
2 Measurement (FTM) procedure defined in IEEE Std 802.11-2016. The Timing measurement
3 messages are the “Timing Measurement frame” and its corresponding “Ack” (see Figure 12-1). The
4 Fine Timing Measurement messages are the “initial FTM request frame” and the “Fine Timing
5 Measurement frame” and its “Ack” (see Figure 12-2).
- 6 c) EPON LANs use the discovery process, where the messages are “GATE” and “REGISTER_REQ”
7 (see Clause 64 and Clause 77 of IEEE Std 802.3-2018).
- 8 d) CSNs either use the same mechanism as full-duplex Ethernet or use a method native to the particular
9 CSN (similar to the way native methods are used by IEEE 802.11 networks and EPON)
10 (see Figure 16-5).

11 7.3.3 Logical syntonization

12 The time-synchronization correction previously described is dependent on the accuracy of the delay and
13 residence time measurements. If the clock used for this purpose is frequency locked (syntonized) to the
14 Grandmaster Clock, then all the time interval measurements use the same time base. Since actually adjusting
15 the frequency of an oscillator (e.g., using a phase-lock loop) is slow and prone to gain peaking effects, PTP
16 Relay Instances can correct time interval measurements using the Grandmaster Clock frequency ratio.

17 Each PTP Instance measures, at each PTP Port, the ratio of the frequency of the PTP Instance at the other
18 end of the link attached to that PTP Port to the frequency of its own clock. The cumulative ratio of the
19 Grandmaster Clock frequency to the local clock frequency is accumulated in a standard organizational type,
20 length, value (TLV) attached to the Follow_Up message (or the Sync message if the optional one-step
21 processing is enabled). The frequency ratio of the Grandmaster Clock relative to the local clock is used in
22 computing synchronized time, and the frequency ratio of the neighbor relative to the local clock is used
23 in correcting the propagation time measurement.

24 The Grandmaster Clock frequency ratio is measured by accumulating neighbor frequency ratios for two
25 main reasons. First, if there is a network reconfiguration and a new Grandmaster PTP Instance is elected, the
26 nearest neighbor frequency ratios do not have to be newly measured as they are constantly measured using
27 the Pdelay messages. This results in the frequency offset relative to the new Grandmaster Clock being
28 known when the first Follow_Up message (or first Sync message if the optional one-step processing is
29 enabled) is received, which reduces the duration of any transient error in synchronized time during the
30 reconfiguration. This is beneficial to many high-end audio applications. Second, there are no gain peaking
31 effects because an error in frequency offset at one PTP Relay Instance, and resulting residence time error,
32 does not directly affect the frequency offset at a downstream PTP Relay Instance.

33 7.3.4 Grandmaster PTP Instance (best timeTransmitter) selection and network 34 establishment

35 All PTP Instances participate in best timeTransmitter selection so that the IEEE 802.1AS protocol can
36 determine the synchronization spanning tree. This synchronization spanning tree can be different from the
37 forwarding spanning tree determined by IEEE 802.1Q™ Rapid Spanning Tree Protocol (RSTP) since the
38 spanning tree determined by RSTP can be suboptimal or even inadequate for synchronization or can be for a
39 different topology of nodes from the synchronization spanning tree.

40 gPTP requires that all systems in the gPTP domain be time-aware systems, i.e., the protocol does not transfer
41 timing over systems that are not time-aware (e.g., those that meet the requirements of IEEE Std
42 802.1Q-2018, but do NOT meet the requirements of the present standard). A time-aware system uses the
43 peer-to-peer delay mechanism on each PTP Port to determine if a non-time-aware system is at the other end
44 of the link or between itself and the Pdelay responder. If, on sending Pdelay_Req,

- 45 a) No response is received,

- 1 b) Multiple responses are received, or
- 2 c) The measured propagation delay exceeds a specified threshold, then

3 the protocol concludes that a non-time-aware system or end-to-end Transparent Clock (TC) (see
4 IEEE Std 1588-2019) is present. In this case, the link attached to the PTP Port is deemed not capable of
5 running gPTP, and the BTCA ignores it. However, the PTP Port continues to attempt the measurement of
6 propagation delay using the peer-to-peer delay mechanism (for full-duplex IEEE 802.3 links), multipoint
7 control protocol (MPCP) messages (for EPON), or IEEE 802.11 messages (for IEEE 802.11 links), and
8 periodically checks whether the link is or is not capable of running the IEEE 802.1AS protocol.

9 7.3.5 Energy efficiency

10 Sending PTP messages at relatively high rates when there is otherwise little or no traffic conflicts with the
11 goal of reducing energy consumption. This standard specifies a way to request that a neighbor PTP Port
12 reduce the rate of sending Sync (and Follow_Up if optional one-step processing is not enabled), peer delay,
13 and Announce messages and also to inform the neighbor not to compute neighbor rate ratio and/or
14 propagation delay on this link. A time-aware system could do this when it enters low-power mode, but this
15 standard does not specify the conditions under which this is done; it specifies only the actions a time-aware
16 system takes.

1 7.4 PTP Instance architecture

2 The model of a PTP Instance and its interfaces to higher-layer applications are shown in Figure 7-8. The
 3 interfaces are those specified in Clause 9.

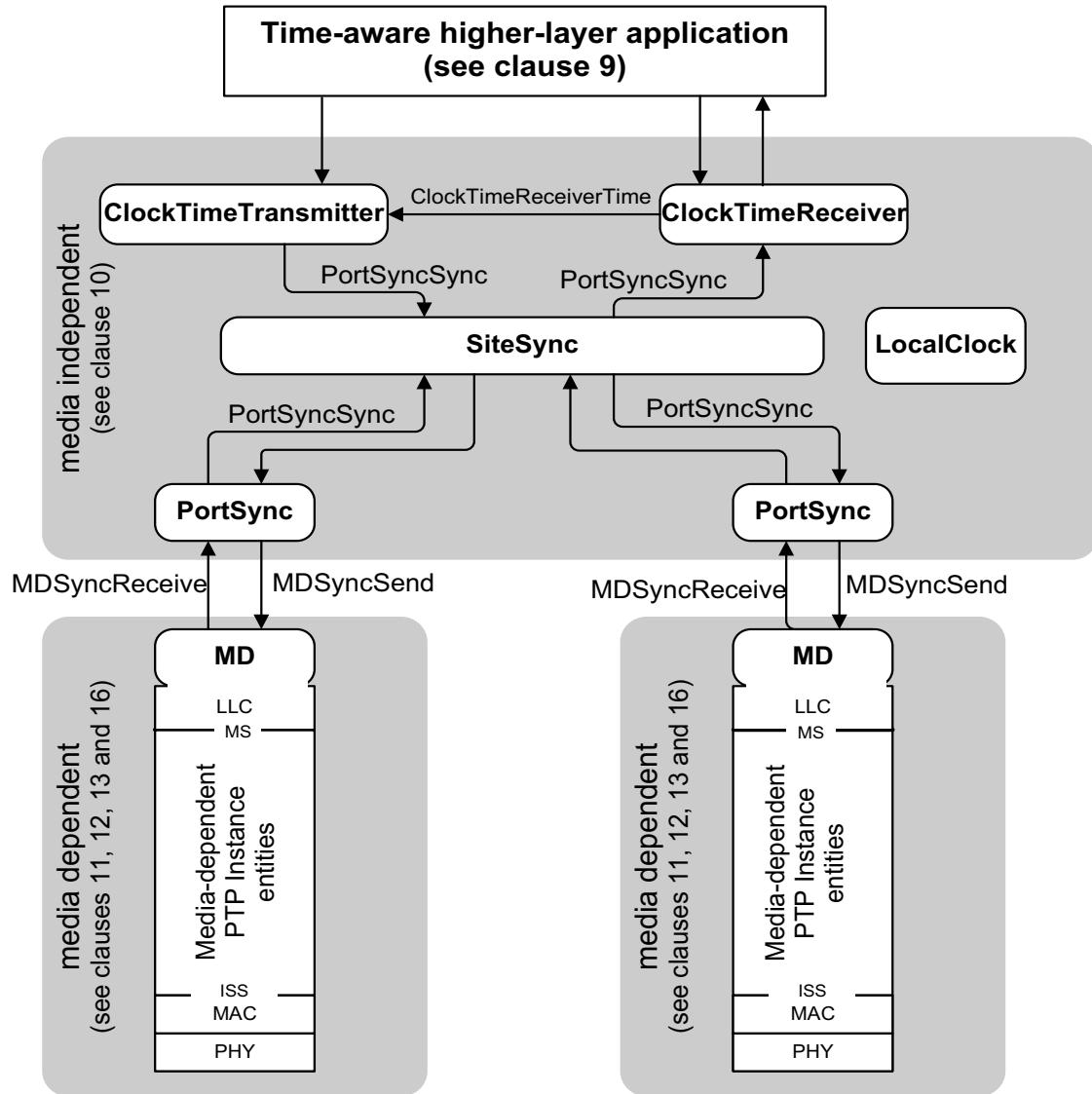


Figure 7-8—Model for a PTP Instance and its interfaces to higher-layer applications

4 A PTP Instance consists of the following major parts:

- 5 a) If the PTP Instance includes application(s) that either use or source time information, then they
 interface with the gPTP information using the application interfaces specified in Clause 9.
- 6 b) A single media-independent part that consists of ClockTimeTransmitter, ClockTimeReceiver, and
 SiteSync logical entities, one or more PortSync entities, and a LocalClock entity. The BTCA and
 forwarding of time information between logical ports and the ClockTimeReceiver and
 ClockTimeTransmitter is done by the SiteSync entity, while the computation of PTP Port-specific
 delays needed for time-synchronization correction is done by the PortSync entities.

1 c) Media-dependent ports, which translate the abstract “MDSyncSend” and “MDSyncReceive”
 2 structures received from or sent to the media-independent layer and corresponding methods used for
 3 the particular LAN attached to the port.

4 For full-duplex Ethernet ports, IEEE 1588 Sync and Follow_U (or just Sync if the optional one-step
 5 processing is enabled) messages are used, with an additional TLV in the Follow_U (or the Sync if the
 6 optional one-step processing is enabled) used for communication of rate ratio and information on phase and
 7 frequency change when there is a change in Grandmaster PTP Instance. The path delay is measured using
 8 the two-step IEEE 1588 peer-to-peer delay mechanism. This is defined in Clause 11.

9 For IEEE 802.11 ports, timing information is communicated using the MAC Layer Management Entity to
 10 request a “Timing Measurement” or “Fine Timing Measurement” (as defined in IEEE Std 802.11-2016),
 11 which also sends everything that would be included in the Follow_up message for full-duplex Ethernet. The
 12 Timing Measurement or Fine Timing Measurement result includes all the information to determine the path
 13 delay. This is defined in Clause 12.

14 For EPON, timing information is communicated using a “slow protocol” as defined in Clause 13. CSNs use
 15 the same communication system used by full-duplex Ethernet, as defined in Clause 16.

16 7.5 Differences between gPTP (IEEE Std 802.1AS) and PTP (IEEE Std 1588-2019)

- 17 a) gPTP assumes all communication between PTP Instances is done only using IEEE 802 MAC PDUs
 18 and addressing, while IEEE Std 1588-2019 supports various layer 2 and layer 3-4 communication
 19 methods.
- 20 b) gPTP specifies a media-independent sublayer that simplifies the integration within a single timing
 21 domain of multiple different networking technologies with radically different media access
 22 protocols. gPTP specifies a media-dependent sublayer for each medium. The information exchanged
 23 between PTP Instances has been generalized to support different packet formats and management
 24 schemes appropriate to the particular networking technology. IEEE Std 1588-2019, on the other
 25 hand, has introduced a new architecture based on media-independent and media-dependent
 26 sublayers (see 6.5.2, Figure 5, and Figure 6 of IEEE Std 1588-2019); however, this architecture is
 27 optional. The architecture of IEEE Std 1588-2008 [B10], which is not based on media-independent
 28 and media-dependent layers, has been retained for Internet Protocol (IP) version 4, IP version 6,
 29 Ethernet LANs, and several industrial automation control protocols. The intent in IEEE Std 1588-
 30 2019 is that the new architecture, based on media-independent and media-dependent layers, will be
 31 used for IEEE 802.11 networks, IEEE 802.3 EPON, and CSN using the specifications of gPTP, and
 32 that the architecture must be used for transports that define native timing mechanisms if those native
 33 timing mechanisms are used.
- 34 c) In gPTP there are only two types of PTP Instances: PTP End Instances and PTP Relay Instances,
 35 while IEEE Std 1588-2019 has Ordinary Clocks, Boundary Clocks, end-to-end Transparent Clocks,
 36 and P2P Transparent Clocks. A PTP End Instance corresponds to an IEEE 1588 Ordinary Clock, and
 37 a PTP Relay Instance is a type of IEEE 1588 Boundary Clock where its operation is very tightly
 38 defined, so much so that a PTP Relay Instance with Ethernet ports can be shown to be
 39 mathematically equivalent to a P2P Transparent Clock in terms of how synchronization is
 40 performed, as shown in 11.1.3. In addition, a PTP Relay Instance can operate in a mode (i.e., the
 41 mode where the variable syncLocked is TRUE; see 10.2.5.15) where the PTP Relay Instance is
 42 equivalent to a P2P Transparent Clock in terms of when time-synchronization messages are sent. A
 43 time-aware system measures link delay and residence time and communicates these in a correction
 44 field. In summary, a PTP Relay Instance conforms to the specifications for a Boundary Clock in
 45 IEEE Std 1588-2019, but a PTP Relay Instance does not conform to the complete specifications for
 46 a P2P Transparent Clock in IEEE Std 1588-2019 because:

- 1 1) When syncLocked is FALSE, the PTP Relay Instance sends Sync according to the
2 specifications for a Boundary Clock, and
- 3 2) The PTP Relay Instance invokes the BTCA and has PTP Port states.
- 4 d) PTP Instances communicate gPTP information only directly with other PTP Instances. That is, a
5 gPTP domain consists ONLY of PTP Instances. Non-PTP Relay Instances cannot be used to relay
6 gPTP information. In IEEE Std 1588-2019, it is possible to use non-IEEE-1588-aware relays in an
7 IEEE 1588 domain, although this slows timing convergence and introduce extra jitter and wander
8 that must be filtered by any IEEE 1588 clock.
- 9 e) For full-duplex Ethernet links, gPTP requires the use of the peer-to-peer delay mechanism, while
10 IEEE Std 1588-2019 also allows the use of end-to-end delay measurement.
- 11 f) For full-duplex Ethernet links, gPTP requires the use of two-step processing (use of Follow_U and
12 Pdelay_Resp_Follow_U messages to communicate timestamps), with an optional one-step
13 processing mode that embeds timestamps in the Sync “on the fly” as they are being transmitted
14 (gPTP does not specify one-step processing for peer delay messages). IEEE Std 1588-2019 allows
15 either two-step or one-step processing to be required (for both Sync and peer delay messages)
16 depending on a specific profile.
- 17 g) All PTP Instances in a gPTP domain are logically syntonized; in other words, they all measure time
18 intervals using the same frequency. This is done by the process described in 7.3.3 and is mandatory.
19 Syntonization in IEEE Std 1588-2019 is optional. The syntonization method used by gPTP is
20 supported as an option in IEEE Std 1588-2019, but uses a TLV standardized as part of
21 IEEE Std 1588-2019 (this feature is new for IEEE Std 1588-2019), while gPTP uses the
22 ORGANIZATION_EXTENSION TLV specified in 11.4.4.3.
- 23 h) Finally, this standard includes formal interface definitions, including primitives, for the time-aware
24 applications (see Clause 9). IEEE Std 1588-2019 describes external interfaces without describing
25 specific interface primitives.

26

1 8. IEEE 802.1AS concepts and terminology

2 8.1 gPTP domain

3 A gPTP domain, hereafter called simply a *domain*, consists of one or more PTP Instances and links that meet
 4 the requirements of this standard and communicate with each other as defined by the IEEE 802.1AS
 5 protocol. A gPTP domain defines the scope of gPTP message communication, state, operations, data sets,
 6 and timescale.

7 A domain is identified by two attributes: **domainNumber** and **sdoId**. The **sdoId** of a domain is a 12-bit
 8 unsigned integer. The **sdoId** is structured as a two-part attribute as follows:

- 9 — The most significant 4 bits are named the **majorSdoId**, and
- 10 — The least significant 8 bits are named the **minorSdoId**.

11 A time-aware system shall support one or more domains, each with a distinct **domainNumber** in the range 0
 12 through 127. Unless otherwise specified in this standard, the operation of gPTP and the timescale in any
 13 given domain is independent of operation in any other domain.

14 The value of **majorSdoId** for a gPTP domain shall be 0x1. The value of **minorSdoId** for a gPTP domain shall
 15 be 0x00 (see 16.5 of IEEE Std 1588-2019).

16 Since the IEEE 802.1 Working Group has only one single unique **sdoId** value, the PTP profile specified in
 17 the present standard is isolated from other PTP profiles (see 16.5.2 of IEEE Std 1588-2019).

18 Both the **domainNumber** and the **sdoId** are carried in the common header of all PTP messages (see 10.6.2.2).

19 NOTE 1—The above requirements for **majorSdoId** and **minorSdoId** are for gPTP domains. The requirements for the
 20 Common Mean Link Delay Service (CMLDS) are given in 11.2.17.

21 NOTE 2—In the 2011 edition of this standard, the attribute **majorSdoId** was named **transportSpecific**, and its value was
 22 specified as 0x1 in 10.5.2.2.1 of Corrigendum 1. The attribute **minorSdoId** did not exist in the 2011 edition, but its
 23 location in the common header was a reserved field, which was specified to be transmitted as 0 and ignored on receipt.

24 Unless otherwise stated, information in the remainder of this document is per domain.

25 NOTE 3—In steady state, all PTP Instances in a gPTP domain are traceable to a single Grandmaster PTP Instance.

26 8.2 Timescale

27 8.2.1 Introduction

28 The timescale for a gPTP domain is established by the Grandmaster Clock. There are two types of
 29 timescales supported by gPTP:

- 30 — The timescale PTP: The epoch is the PTP epoch (see 8.2.2), and the timescale is continuous. The
 31 unit of measure of time is the second defined by International Atomic Time (TAI) (for the definition
 32 of TAI, see Service de la Rotation Terrestre [B28], with further amplification in IAU [B27]; and for
 33 more information on TAI, see Jekeli [B22], international system of units (SI) brochure [B14], and
 34 Petit and Luzum [B26]). The timescale of domain 0 shall be PTP. See IEEE Std 1588-2019 for
 35 more details.
- 36 — The timescale ARB (arbitrary): The epoch is the domain startup time and can be set by an
 37 administrative procedure. Between invocations of the administrative procedure, the timescale is

1 continuous. Additional invocations of the administrative procedure can introduce discontinuities in
2 the overall timescale. The unit of measure of time is determined by the Grandmaster Clock. The
3 second used in the operation of the protocol can differ from the SI second.

4 **8.2.2 Epoch**

5 The epoch is the origin of the timescale of a gPTP domain.

6 The PTP epoch (epoch of the timescale PTP) is 1 January 1970 00:00:00 TAI.

7 NOTE—The common portable operating system interface (POSIX) algorithms can be used for converting elapsed
8 seconds since the PTP epoch to the ISO 8601:2004 [B15] printed representation of time of day on the TAI timescale (see
9 also ISO/IEC 9945:2003 [B17]).

10 See Annex C for information on converting between common timescales.

11 **8.2.3 UTC offset**

12 When the timescale is PTP, it is possible to calculate Coordinated Universal Time (UTC) time using the
13 value of currentUtcOffset. The value of currentUtcOffset is given by

14 $\text{currentUtcOffset} = \text{TAI} - \text{UTC}$

15 where the difference TAI – UTC is derived from UTC – TAI (the negative of currentUtcOffset) specified in
16 IERS Bulletin C.

17 The value of currentUtcOffset for the current Grandmaster PTP Instance is maintained in the
18 currentUtcOffset member of the time properties data set (see 14.5.2).

19 NOTE 1—As of 0 hours 1 January 2017 UTC, UTC was behind TAI by 37 s, i.e., $\text{TAI} - \text{UTC} = +37$ s. At that moment,
20 the IEEE-802.1AS-defined value of currentUtcOffset was +37 s, as designated in the applicable IERS Bulletin C
21 (see Clause 2; see also Service de la Rotation Terrestre [B28] and U.S. Naval Observatory [B29]).

22 NOTE 2—Leap second events and the value of UTC – TAI are posted well in advance in IERS Bulletin C. A list of all
23 leap second events is maintained by the U.S. Naval Observatory [B29], which also offers an extensive discussion of
24 timescales, leap seconds, and related time issues.

25 NOTE 3—The value of currentUtcOffset represents the difference between TAI and UTC. Since 1972, only integral
26 values are permitted for this difference. Due to leap seconds, UTC cannot be correctly represented as a single integer but
27 can be expressed in the ISO 8601:2004 [B15] print form. See also C.2 for an example of the use of the POSIX algorithm
28 to compute the correct print form.

29 When the timescale is PTP, it is possible to calculate local time from the time provided by a gPTP domain
30 using the currentUtcOffset, leap59, and leap61 member values of the time properties data set and knowledge
31 of the local time zone and whether and when daylight savings time is observed.

32 When the timescale is ARB, the values of currentUtcOffset, leap59, and leap61 cannot be used to compute
33 UTC.

34 The mechanism for computing UTC or any other time does not change the synchronized time (see 3.29), i.e.,
35 the PTP Instance time, of a PTP Instance.

36 **8.2.4 Measurement of time within a gPTP domain**

37 Time in a gPTP domain shall be measured as elapsed time since the epoch of the timescale of that domain.

1 8.3 Link asymmetry

2 This standard requires the measurement of the mean propagation time (also known as the *mean propagation*
 3 *delay*) between the time-aware systems that comprise the two endpoints of a link. The measurement is
 4 performed when one of the time-aware systems (the initiator time-aware system) sends a message to the
 5 other time-aware system (the responder time-aware system). The responder then sends a message back to
 6 the initiator at a later time. The departure of the message sent by the initiator time-aware system is
 7 timestamped, and the timestamp value is retained by that system. The arrival of this message at the
 8 responder time-aware system is timestamped, and the timestamp value is conveyed to the initiator time-
 9 aware system in a subsequent message. The departure of the response message sent by the responder time-
 10 aware system (in response to the message it receives from the initiator time-aware system) is timestamped,
 11 and the timestamp value is conveyed to the initiator time-aware system in a subsequent message. The arrival
 12 of this response message at the initiator time-aware system is timestamped, and the timestamp value is
 13 retained by that system. The mean propagation time is computed by the initiator time-aware system after
 14 receiving the response message, from the four timestamp values it has at this point.

15 Typically, the propagation time is not exactly the same in both directions, and the degree to which it differs
 16 in the two directions is characterized by the delay asymmetry. The relation between the individual
 17 propagation times in the two directions, the mean propagation time, and the delay asymmetry is as follows.
 18 Let t_{ir} be the propagation time from the initiator to the responder, t_{ri} be the propagation time from the
 19 responder to the initiator, meanLinkDelay be the mean propagation time (see 10.2.5.8), and
 20 delayAsymmetry be the delay asymmetry. The propagation times in the two directions are illustrated in
 21 Figure 8-1.

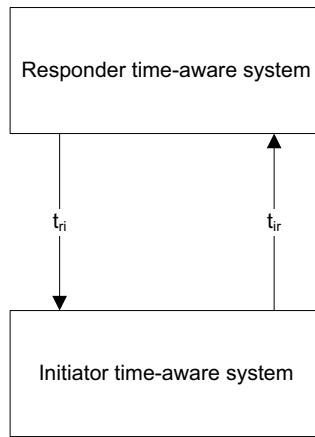


Figure 8-1—Propagation asymmetry

22 The meanLinkDelay is the mean value of t_{ir} and t_{ri} , i.e., $\text{meanLinkDelay} = (t_{ir} + t_{ri}) / 2$. The
 23 delayAsymmetry is defined as:

24 $t_{ir} = \text{meanLinkDelay} - \text{delayAsymmetry}$

25 $t_{ri} = \text{meanLinkDelay} + \text{delayAsymmetry}$

26 In other words, delayAsymmetry is defined to be positive when the responder to initiator propagation time is
 27 longer than the initiator to responder propagation time.

28 This standard does not explicitly require the measurement of delayAsymmetry; however, if
 29 delayAsymmetry is modeled, it shall be modeled as specified in this clause.

1 NOTE 1—A time-aware system can change the value of delayAsymmetry during operation (see 14.8.10, 14.16.8, and
2 Annex G).

3 NOTE 2—A time-aware system PTP Port cannot measure the value of delayAsymmetry during live operation of the
4 system (i.e., asymmetryMeasurementMode is FALSE; see 10.2.5.2); therefore, the value of delayAsymmetry must be
5 defined separately using information from the supplier or additional testing before running the live system. The methods
6 for measuring asymmetry are not specified in this standard. These values can be added to the system configuration to
7 improve the accuracy of time synchronization. The inaccuracy caused by asymmetry is half the value of the difference
8 between t_{ri} and t_{ir} , and these inaccuracies can either accumulate over successive hops or, if the successive asymmetries
9 have different signs, cancel each other over the successive hops.

10 8.4 Messages

11 8.4.1 General

12 All communications occur via PTP messages and/or media-specific messages.

13 8.4.2 Message attributes

14 8.4.2.1 General

15 All messages used in this standard have the following attributes:

- 16 a) Message class
- 17 b) Message type

18 The message class attribute is defined in this clause. The message type attribute is defined in 3.18. Some
19 messages have additional attributes; these are defined in the subclauses where the respective messages are
20 defined.

21 8.4.2.2 Message class

22 There are two message classes, the event message class and the general message class. Event messages are
23 timestamped on egress from a PTP Instance and ingress to a PTP Instance. General messages are not
24 timestamped. Every message is either an event message or a general message.

25 8.4.3 Generation of event message timestamps

26 All event messages are timestamped on egress and ingress. The timestamp shall be the time, relative to the
27 LocalClock entity (see 10.1) at which the message timestamp point passes the reference plane marking the
28 boundary between the PTP Instance and the network media.

29 The definition of the timestamp measurement plane (see 3.33), along with the corrections defined as
30 follows, allows transmission delays to be measured in such a way (at such a low layer) that they appear fixed
31 and symmetrical to gPTP even though the MAC client might otherwise observe substantial asymmetry and
32 transmission variation. For example, the timestamp measurement plane is located below any retransmission
33 and queuing performed by the MAC.

34 NOTE 1—If an implementation generates event message timestamps using a point other than the message timestamp
35 point, then the generated timestamps should be appropriately corrected by the time interval (fixed or otherwise) between
36 the actual time of detection and the time the message timestamp point passed the reference plane. Failure to make these
37 corrections results in a time offset between PTP Instances.

38 NOTE 2—In general, the timestamps can be generated at a timestamp measurement plane that is removed from the
39 reference plane. Furthermore, the timestamp measurement plane, and therefore the time offset of this plane from the

1 reference plane, is likely to be different for inbound and outbound event messages. To meet the requirement of this
 2 clause, the generated timestamps should be corrected for these offsets. Figure 8-2 illustrates these offsets. Based on this
 3 model the appropriate corrections are as follows:

- 4 egressTimestamp = egressMeasuredTimestamp + egressLatency
- 5 ingressTimestamp = ingressMeasuredTimestamp - ingressLatency

6 where the timestamps relative to the reference plane, egressTimestamp and ingressTimestamp, are computed from the
 7 timestamps relative to the timestamp measurement plane, egressMeasuredTimestamp and ingressMeasuredTimestamp,
 8 respectively, using their respective latencies, egressLatency and ingressLatency. Failure to make these corrections results
 9 in a time offset between the timeReceiver and timeTransmitter clocks.

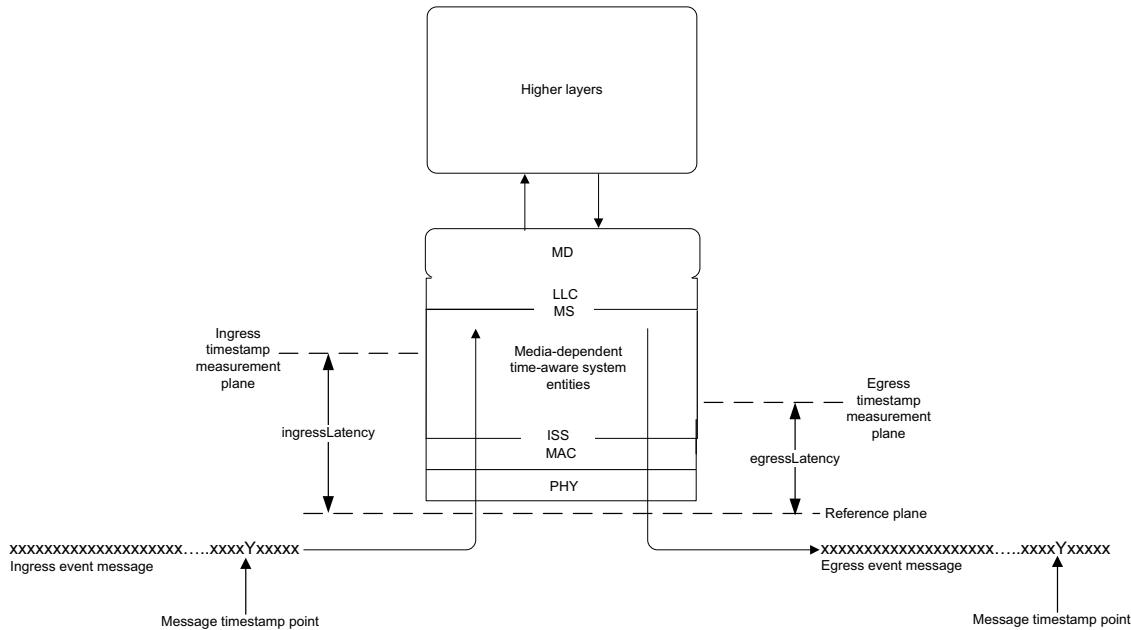


Figure 8-2—Definition of message timestamp point, reference plane, timestamp measurement plane, and latency constants

10 8.4.4 Priorities

11 IEEE Std 802.1AS messages shall be transmitted in expedited manner compared to other traffic (e.g., best effort).

13 NOTE 1—If IEEE Std 802.1AS messages are not expedited in the internal queueing, long bursts of other traffic can cause loss of synchronization due to timeouts.

15 NOTE 2—For example, two outbound queues can be supported: one outbound queue is used to transmit nonexpedited traffic (e.g., best-effort), and another outbound queue is used to transmit frames in an expedited manner, including IEEE 802.1AS messages. Outbound queues are often implemented in hardware, but software implementation is possible.

19 NOTE 3—When IEEE Std 802.1Q-2018 is supported, the outbound queue used for transmitting IEEE 802.1AS messages uses a traffic class (see 3.268 of IEEE Std 802.1Q-2018) greater than zero.

21 NOTE 4—Frames carrying IEEE 802.1AS messages are neither VLAN-tagged nor priority-tagged, i.e., they are untagged (see 11.3.3).

1 8.5 Ports

2 8.5.1 General

3 The PTP Instances in a gPTP domain interface with the network media via physical ports. gPTP defines a
4 logical port, i.e., a PTP Port, in such a way that communication between PTP Instances is point-to-point
5 even over physical ports that are attached to shared media. One logical port, consisting of one PortSync
6 entity and one media-dependent (MD) entity, is instantiated for each PTP Instance with which the
7 PTP Instance communicates. For shared media, multiple logical ports can be associated with a single
8 physical port.

9 Unless otherwise qualified, each instance of the term *port* refers to a *logical port*.

10 8.5.2 Port identity

11 8.5.2.1 General

12 A PTP Port is identified by a port identity of type PortIdentity (see 6.4.3.7). The value is maintained in
13 portDS.portIdentity (see 14.8.2). A port identity consists of the following two attributes:

- 14 a) portIdentity.clockIdentity
15 b) portIdentity.portNumber

16 8.5.2.2 clockIdentity

17 The clockIdentity attribute shall be as specified in 7.5.2.2 of IEEE Std 1588-2019.

18 8.5.2.3 Port number

19 The portNumber values for the PTP Ports on a time-aware system shall be distinct in the range **0x0001**
20 **through** **0xFFFF**.

21 The portNumber value 0 is assigned to the interface between the ClockTimeTransmitter and ClockSource
22 entities (see 10.1 and Figure 10-1). The value 0xFFFF is reserved. **The assignment of portNumber 0 to this**
23 **interface helps to simplify the SiteSync and PortStateSelection state machines; this interface is not a PTP**
24 **Port.**

25 8.5.2.4 Ordering of clockIdentity and portIdentity values

26 Two clockIdentity values X and Y are compared as follows. Let x be the unsigned integer formed by
27 concatenating octets 0 through 7 of X such that octet $j+1$ follows octet j (i.e., is less significant than octet j)
28 in x ($j = 0, 1, \dots, 6$). Let y be the unsigned integer formed by concatenating octets 0 through 7 of Y such that
29 octet $j+1$ follows octet j (i.e., is less significant than octet j) in y ($j = 0, 1, \dots, 6$). Then:

- 30 a) X = Y if and only if $x = y$,
31 b) X > Y if and only if $x > y$, and
32 c) X < Y if and only if $x < y$.

33 Two portIdentity values A and B with members clockIdentity and portNumber are compared as follows. Let
34 a be the unsigned integer formed by concatenating octets 0 through 7 of A.clockIdentity, such that octet $j+1$
35 follows octet j (i.e., is less significant than octet j) in a ($j = 0, 1, \dots, 6$), followed by octet 0 of A.portNumber,
36 followed by octet 1 of A.portNumber. Let b be the unsigned integer formed by concatenating octets 0

1 through 7 of B.clockIdentity, such that octet $j+1$ follows octet j (i.e., is less significant than octet j) in
2 b ($j = 0, 1, \dots, 6$), followed by octet 0 of B.portNumber, followed by octet 1 of B.portNumber. Then:

- 3 d) $A = B$ if and only if $a = b$,
- 4 e) $A > B$ if and only if $a > b$, and
- 5 f) $A < B$ if and only if $a < b$.

6 A portIdentity A with members clockIdentity and portNumber and a clockIdentity B are compared as
7 follows. The unsigned integer a is formed from portIdentity A as described above. The unsigned integer b is
8 formed by first forming a portIdentity B' whose clockIdentity is B and portNumber is 0. b is then formed
9 from B' as described above. A and B are then compared as described in items d) through f) in this subclause.

10 8.6 PTP Instance characterization

11 8.6.1 PTP Instance type

12 There are two types of PTP Instances used in a gPTP domain, as follows:

- 13 a) PTP End Instance
- 14 b) PTP Relay Instance

15 All PTP Instances are identified by clockIdentity.

16 In addition, PTP Instances are characterized by the following attributes:

- 17 c) priority1
- 18 d) clockClass
- 19 e) clockAccuracy
- 20 f) offsetScaledLogVariance
- 21 g) priority2
- 22 h) clockIdentity
- 23 i) timeSource
- 24 j) numberPorts

25 NOTE—Attributes c) through i) can be considered to be associated with the ClockTimeTransmitter entity of the PTP
26 Instance.

27 8.6.2 PTP Instance attributes

28 8.6.2.1 priority1

29 priority1 is used in the execution of the BTCA (see 10.3). The value of priority1 is an integer selected from
30 the range 0 through 255. The ordering of priority1 in the operation of the BTCA (see 10.3.4 and 10.3.5) is
31 specified as follows. A ClockTimeTransmitter A shall be deemed better than a ClockTimeTransmitter B if
32 the value of priority1 of A is numerically less than that of B.

33 The value of priority1 shall be 255 for a PTP Instance that is not grandmaster-capable. The value of priority1
34 shall be less than 255 for a PTP Instance that is grandmaster-capable. The value 0 shall be reserved for
35 management use, i.e., the value of priority1 shall be set to 0 only via management action. The default value
36 shall be set to one of the values listed in Table 8-1, and the choice of value from Table 8-1 is left to the
37 implementer of the PTP Instance.

Table 8-1—Default values for priority1, for the respective media

PTP Instance type	Default value for priority1
PTP Instances that are a central/critical part of the network and are not expected to ever be turned off during normal network operation (e.g., Bridges, wireless access points, and grandmaster-capable end nodes that are intended by the manufacturer to be Grandmaster PTP Instances). (End node Grandmaster PTP Instances are not a ‘central’ part of the network, but they are “critical” to this gPTP operation.)	246
PTP Instances that (A) can be turned off at any time and therefore cannot be considered a central/critical part of the network; or (B) are a central/critical part of the network, are not expected to ever be turned off during normal network operation, and are grandmaster-capable, but are not intended by the manufacturer to be Grandmaster PTP Instances. PTP Instances that can be turned off and are turned off affect only the function(s) they support and do not affect any other functions of the network (e.g., desktop computers, fixed (heavy or otherwise) end nodes, speakers, receivers, amplifiers, and televisions).	248
PTP Instances that can go away (physically or otherwise) at any time (e.g., PTP Instances that are designed to be transient to the network such as laptop computers, cell phones, and battery-powered speakers).	250
PTP Instance that is not grandmaster-capable.	255

1 NOTE 1—Care must be applied to multi-function device design, specifically for an end station that also contains a
 2 Bridge. The Bridge function inside multi-function devices must not be powered down when the end station function is
 3 powered down, or else the data and controls (like gPTP) passing through the Bridge function to other network devices
 4 will cease. Example devices of this are a television and/or amplifier/receiver, each of which contains a Bridge.

5 NOTE 2—The BTCA (see 10.3) considers priority1 before other attributes; the priority1 attribute can therefore be used
 6 to force a desired ordering of PTP Instances for best timeTransmitter selection.

7 NOTE 3—The settings for priority1 in Table 8-1 guarantee that a PTP Instance that is grandmaster-capable is always
 8 preferred by the BTCA over a PTP Instance that is not grandmaster-capable.

9 NOTE 4—These values are assigned so that PTP Instances with priority1 value of 246 are selected as Grandmaster PTP
 10 Instances over PTP Instances with priority1 values of 248 or 250. (PTP Instances with priority1 value of 255 are never
 11 selected as Grandmaster PTP Instances.)

12 NOTE 5—These default values are suitable for applications in which the availability of the Grandmaster PTP Instance is
 13 the most important criterion for Grandmaster PTP Instance selection. A PTP Instance built for a specific application for
 14 which this is not the case can be capable of having priority1 changed via management.

15 8.6.2.2 clockClass

16 The clockClass attribute denotes the traceability of the synchronized time distributed by a
 17 ClockTimeTransmitter when it is the Grandmaster PTP Instance.

18 The value shall be selected as follows:

- 19 a) If defaultDS.gmCapable is TRUE (see 14.2.7), then
 - 20 1) clockClass is set to the value that reflects the combination of the LocalClock and ClockSource
 21 entities; else
 - 22 2) If the value that reflects the LocalClock and ClockSource entities is not specified or not known,
 23 clockClass is set to 248;
 - 24 b) If the defaultDS.gmCapable is FALSE, clockClass is set to 255 (see 8.6.2.1).

1 The ordering of clockClass in the operation of the best timeTransmitter clock algorithm (see 10.3.4 and
 2 10.3.5) is specified as follows. When comparing clockClass values, PTP Instance A shall be deemed better
 3 than PTP Instance B if the value of the clockClass of A is lower than that of B.

4 See 7.6.2.5 of IEEE Std 1588-2019 for a more detailed description of clockClass.

5 NOTE—The PTP Instance has a LocalClock entity, which can be the free-running quartz crystal that just meets the IEEE
 6 802.3 requirements, but could also be better. There can be a ClockSource entity, e.g., timing taken from a GNSS,
 7 available in the local system that provides timing to the ClockSource entity. The time provided by the PTP Instance, if it
 8 is the Grandmaster PTP Instance, is reflected by the combination of these two entities, and the clockClass reflects this
 9 combination as specified in 7.6.2.5 of IEEE Std 1588-2019. For example, when the LocalClock entity uses a quartz
 10 oscillator that meets the requirements of IEEE Std 802.3-2018 and B.1 of this standard, clockClass is set to 248. But, if a
 11 GNSS receiver is present and synchronizes the PTP Instance, then the clockClass is set to the value 6, indicating
 12 traceability to a primary reference time source (see 7.6.2.5 of IEEE Std 1588-2019).

13 8.6.2.3 clockAccuracy

14 The clockAccuracy attribute indicates the expected time accuracy of a ClockTimeTransmitter.

15 The value shall be selected as follows:

- 16 a) clockAccuracy is set to the value that reflects the combination of the LocalClock and ClockSource
 17 entities; else
- 18 b) If the value that reflects the LocalClock and ClockSource entities is not specified or unknown,
 19 clockAccuracy is set to 254 (FE₁₆).

20 The ordering of clockAccuracy in the operation of the best timeTransmitter clock algorithm (see 10.3.4 and
 21 10.3.5) is specified as follows. When comparing clockAccuracy values, PTP Instance A shall be deemed
 22 better than PTP Instance B if the value of the clockAccuracy of A is lower than that of B.

23 See 7.6.2.6 of IEEE Std 1588-2019 for more detailed description of clockAccuracy.

24 8.6.2.4 offsetScaledLogVariance

25 The offsetScaledLogVariance is a scaled, offset representation of an estimate of the PTP variance. The PTP
 26 variance characterizes the precision and frequency stability of the ClockTimeTransmitter. The PTP variance
 27 is the square of PTP Deviation (PTPDEV) (see B.1.3.2).

28 The value shall be selected as follows:

- 29 a) offsetScaledLogVariance is set to the value that reflects the combination of the LocalClock and
 30 ClockSource entities; else
- 31 b) If the value that reflects these entities is not specified or not known, offsetScaledLogVariance is set
 32 to 17258 (436A₁₆). This value corresponds to the value of PTPDEV for observation interval equal to
 33 the default Sync message transmission interval (i.e., observation interval of 0.125 s; see 11.5.2.3
 34 and B.1.3.2).

35 The ordering of offsetScaledLogVariance in the operation of the best timeTransmitter clock algorithm (see
 36 10.3.4 and 10.3.5) is specified as follows. When comparing offsetScaledLogVariance values, PTP Instance
 37 A shall be deemed better than PTP Instance B if the value of the offsetScaledLogVariance of A is lower than
 38 that of B.

39 See 7.6.3 of IEEE Std 1588-2019 for more detailed description of PTP variance and
 40 offsetScaledLogVariance. (Subclause 7.6.3.3 of IEEE Std 1588-2019 provides a detailed description of the
 41 computation of offsetScaledLogVariance from PTP variance, along with an example.)

1 8.6.2.5 priority2

2 priority2 is used in the execution of the BTCA (see 10.3). The value of priority2 shall be an integer selected
 3 from the range 0 through 255. The ordering of priority2 in the operation of the BTCA is the same as the
 4 ordering of priority1 (see 8.6.2.1).

5 The default value of priority2 shall be 247 or 248. The default value for a PTP Relay Instance should be 247.
 6 The default value for a PTP End Instance should be 248. See 7.6.2.4 of IEEE Std 1588-2019 for a more
 7 detailed description of priority2.

8 NOTE—IEEE 802.1AS performance is improved when the number of hops between the Grandmaster PTP Instance and
 9 a timeReceiver PTP End Instance is reduced. When BTCA attributes are equal in a network, the preceding
 10 recommendations for priority2 select a PTP Relay Instance in order to reduce the number of hops (rather than use
 11 clockIdentity alone).

12 8.6.2.6 clockIdentity

13 The clockIdentity value for a PTP Instance shall be as specified in 8.5.2.2.

14 8.6.2.7 timeSource

15 The timeSource is an information only attribute indicating the type of source of time used by a
 16 ClockTimeTransmitter. The value is not used in the selection of the Grandmaster PTP Instance. The data
 17 type of timeSource shall be TimeSource, which is an Enumeration8. The values of TimeSource are specified
 18 in Table 8-2. These represent categories. For example, the global positioning system (GPS) entry includes
 19 not only the GPS system of the U.S. Department of Defense but the European Galileo system and other
 20 present and future GNSSs.

Table 8-2—TimeSource enumeration

Value	Time source	Description
0x10	ATOMIC_CLOCK	Any PTP Instance, or PTP Instance directly connected to such a device, that is based on atomic resonance for frequency and that has been calibrated against international standards for frequency and time.
0x20	GPS (see NOTE 1)	Any PTP Instance synchronized to any of the satellite systems that distribute time and frequency tied to international standards.
0x30	TERRESTRIAL_RADIO	Any PTP Instance synchronized via any of the radio distribution systems that distribute time and frequency tied to international standards.
0x40	PTP	Any PTP Instance synchronized to an IEEE 1588 PTP-based source of time external to the gPTP domain (see NOTE 2).
0x50	NTP	Any PTP Instance synchronized via the network time protocol (NTP) to servers that distribute time and frequency tied to international standards.
0x60	HAND_SET	Used in all cases for any PTP Instance whose time has been set by means of a human interface based on observation of an international standards source of time to within the claimed clock accuracy.

Table 8-2—TimeSource enumeration (continued)

Value	Time source	Description
0x90	OTHER	Any source of time and/or frequency not covered by other values, or for which the source is not known.
0xA0	INTERNAL_OSCILLATOR	Any PTP Instance whose frequency is not based on atomic resonance nor calibrated against international standards for frequency, and whose time is based on a free-running oscillator with epoch determined in an arbitrary or unknown manner.
NOTE 1—In this standard, this value refers to any GNSS or Regional Navigation Satellite System (i.e., not only GPS).		
NOTE 2—For example, a clock that implements both a gPTP domain and a separate IEEE 1588 (i.e., PTP) domain, and is synchronized by the separate IEEE 1588 domain, would have time source of PTP in the gPTP domain.		

1 All unused values are reserved.

2 See 7.6.2.8 of IEEE Std 1588-2019 for a more detailed description of timeSource.

3 The initialization value is selected as follows:

- 4 a) If the timeSource (8.6.2.7 and Table 8-2) is known at the time of initialization, the value is derived from the table, else
- 5 b) The value is set to A0₁₆ (INTERNAL_OSCILLATOR).

7 8.6.2.8 numberPorts

8 The numberPorts indicates the number of PTP Ports of the PTP Instance.

1 **9. Application interfaces**

2 **9.1 Overview of the interfaces**

3 Unless otherwise stated, information in this clause is per domain.

4 The following subclauses define one application interface between the ClockSource entity and
5 ClockTimeTransmitter entity (see 10.1.2) and four application interfaces between the ClockTarget entity and
6 ClockTimeReceiver entity (see 10.1.2). The ClockSource is an entity that can be used as an external timing
7 source for the gPTP domain. The ClockSource entity either contains or has access to a clock (see 3.3). The
8 ClockTarget entity represents any application that uses information provided by the ClockTimeReceiver
9 entity via any of the application interfaces.

10 NOTE 1—The manner in which the ClockSource entity obtains time from a clock is outside the scope of this standard.
11 The manner in which the ClockTarget uses the information provided by application interfaces is outside the scope of this
12 standard.

13 The five interfaces are illustrated in Figure 9-1:

- 14 a) ClockSourceTime interface, which provides external timing to a PTP Instance,
15 b) ClockTargetEventCapture interface, which returns the synchronized time of an event signaled by a
16 ClockTarget entity,
17 c) ClockTargetTriggerGenerate interface, which causes an event to be signaled at a synchronized time
18 specified by a ClockTarget entity,
19 d) ClockTargetClockGenerator interface, which causes a periodic sequence of results to be generated,
20 with a phase and rate specified by a ClockTarget entity, and
21 e) ClockTargetPhaseDiscontinuity interface, which supplies information that an application can use to
22 determine if a discontinuity in Grandmaster Clock phase or frequency has occurred.

23 NOTE 2—The application interfaces described in this clause are models for behavior and not application program
24 interfaces. Other application interfaces besides items a) through e) in this subclause are possible, but are not described
25 here. In addition, there can be multiple instances of a particular interface.

1

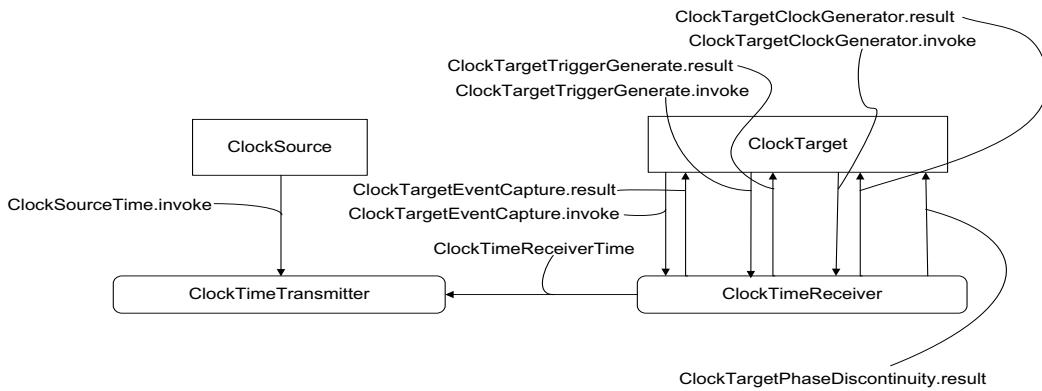


Figure 9-1—Application interfaces

2 9.2 ClockSourceTime interface

3 9.2.1 General

4 This interface is used by the ClockSource entity to provide time to the ClockTimeTransmitter entity of a
 5 PTP Instance. The ClockSource entity invokes the **ClockSourceTime.invoke** function. The function
 6 provides the time, relative to the ClockSource, at which the function was invoked.

7 9.2.2 ClockSourceTime.invoke function parameters

```

8 ClockSourceTime.invoke {
9     domainNumber,
10    sourceTime,
11    timeBaseIndicator,
12    lastGmPhaseChange,
13    lastGmFreqChange
14 }
    
```

15 9.2.2.1 domainNumber (UInteger8)

16 This parameter is the **domainNumber** of the gPTP domain to which this ClockSource entity is providing
 17 time.

18 9.2.2.2 sourceTime (ExtendedTimestamp)

19 The value of sourceTime is the time this function is invoked by the ClockSource entity.

20 9.2.2.3 timeBaseIndicator (UInteger16)

21 The timeBaseIndicator is a value that is set by the ClockSource entity. The ClockSource entity changes the
 22 value whenever its time base changes. The ClockSource entity shall change the value of timeBaseIndicator
 23 if and only if there is a phase or frequency change.

1 NOTE—While the clock that supplies time to the ClockSource entity can be lost, the ClockSource entity itself is not
 2 lost, the ClockSource entity ensures that timeBaseIndicator changes if the source of time is lost.

3 **9.2.2.4 lastGmPhaseChange (ScaledNs)**

4 The value of lastGmPhaseChange is the phase change (i.e., change in sourceTime) that occurred on the most
 5 recent change in timeBaseIndicator. The value is initialized to 0.

6 **9.2.2.5 lastGmFreqChange (Float64)**

7 The value of lastGmFreqChange is the fractional frequency change (i.e., frequency change expressed as a
 8 pure fraction) that occurred on the most recent change in timeBaseIndicator. The value is initialized to 0.

9 **9.3 ClockTargetEventCapture interface**

10 **9.3.1 General**

11 This interface is used by the ClockTarget entity to request the synchronized time of an event that it signals to
 12 the ClockTimeReceiver entity of a PTP Instance. The ClockTarget entity invokes the
 13 ClockTargetEventCapture.invoke function to signal an event to the ClockTimeReceiver entity. The
 14 ClockTimeReceiver entity invokes the ClockTargetEventCapture.result function to return the time of the
 15 event relative to the current Grandmaster Clock or, if no PTP Instance is grandmaster-capable, the
 16 LocalClock. The ClockTargetEventCapture.result function also returns gmPresent, to indicate to the
 17 ClockTarget whether a Grandmaster PTP Instance is present.

18 **ClockTargetEventCapture.invoke parameters**

```
19 ClockTargetEventCapture.invoke {
20     domainNumber
21 }
```

22 **9.3.2.1 domainNumber (UInteger8)**

23 This parameter is the **domainNumber** of the ClockTimeReceiver entity that is requested to provide the
 24 synchronized time of the signaled event.

25 **ClockTargetEventCapture.result parameters**

```
26 ClockTargetEventCapture.result {
27     domainNumber,
28     timeReceiverTimeCallback,
29     gmPresent
30     isSynced,
31     grandmasterIdentity
32 }
```

33 **9.3.3.1 domainNumber (UInteger8)**

34 This parameter is the **domainNumber** of the ClockTimeReceiver entity that is providing the synchronized
 35 time of the signaled event.

1 **9.3.3.2 timeReceiverTimeCallback (ExtendedTimestamp)**

2 The value of timeReceiverTimeCallback is the time, relative to the Grandmaster Clock, that the
 3 corresponding ClockTargetEventCapture.invoke function is invoked.

4 NOTE—The invocation of the ClockTargetEventCapture.invoke function and the detection of this invocation by the
 5 ClockTimeReceiver entity are simultaneous in this abstract interface.

6 **9.3.3.3 gmPresent (Boolean)**

7 The value of gmPresent is set equal to the value of the global variable gmPresent (see 10.2.4.13). This
 8 parameter indicates to the ClockTarget whether a Grandmaster PTP Instance is present.

9 **9.3.3.4 isSynced (Boolean)**

10 The value of the variable isSynced (see 18.4.1.1). This parameter shall be present if the optional hot
 11 standby feature is implemented (see Clause 18).

12 **9.3.3.5 grandmasterIdentity (ClockIdentity)**

13 The value of grandmasterIdentity is the clockIdentity of the Grandmaster PTP Instance.

14 **9.4 ClockTargetTriggerGenerate interface**

15 **9.4.1 General**

16 This interface is used by the ClockTarget entity to request that the ClockTimeReceiver entity send a result at
 17 a specified time relative to the Grandmaster Clock. The ClockTarget entity invokes the
 18 ClockTargetTriggerGenerate.invoke function to indicate the synchronized time of the event. The
 19 ClockTimeReceiver entity invokes the ClockTargetTriggerGenerate.result function to either signal the event
 20 at the requested synchronized time or indicate an error condition.

21 **9.4.2 ClockTargetTriggerGenerate.invoke parameters**

```
22 ClockTargetTriggerGenerate.invoke {
23     domainNumber,
24     timeReceiverTimeCallback
25 }
```

26 **9.4.2.1 domainNumber (UInteger8)**

27 This parameter is the domainNumber of the ClockTimeReceiver entity that is requested to signal an event at
 28 the specified time.

29 **9.4.2.2 timeReceiverTimeCallback (ExtendedTimestamp)**

30 If timeReceiverTimeCallback is nonzero, its value is the synchronized time the corresponding
 31 ClockTargetTriggerGenerate.result function, i.e., the trigger, is to be invoked. If timeReceiverTimeCallback
 32 is zero, any previous ClockTargetTriggerGenerate.invoke function for which a
 33 ClockTargetTriggerGenerate.result function has not yet been issued is canceled.

1 **9.4.3 ClockTargetTriggerGenerate.result parameters**

```
2 ClockTargetTriggerGenerate.result {
3     domainNumber,
4     errorCondition,
5     gmPresent
6     isSynced,
7     grandmasterIdentity
8 }
```

9 **9.4.3.1 domainNumber (UInteger8)**

10 This parameter is the **domainNumber** of the ClockTimeReceiver entity that is triggering an event at the
11 specified time.

12 **9.4.3.2 errorCondition (Boolean)**

13 A value of FALSE indicates that the ClockTargetTriggerGenerate.result function was invoked at the time,
14 relative to the Grandmaster Clock, contained in the corresponding ClockTargetTriggerGenerate.invoke
15 function. A value of TRUE indicates that the ClockTargetTriggerGenerate.result function could not be
16 invoked at the synchronized time contained in the corresponding ClockTargetTriggerGenerate.invoke
17 function.

18 NOTE—For example, the ClockTargetTriggerGenerate.result function is invoked with errorCondition = TRUE if the
19 requested timeReceiverTimeCallback is a time prior to the synchronized time when the corresponding
20 ClockTargetTriggerGenerate.invoke function is invoked. As another example, the ClockTargetTriggerGenerate.result
21 function is invoked with errorCondition = TRUE if a discontinuity in the synchronized time causes the requested
22 timeReceiverTimeCallback to be skipped over.

23 **9.4.3.3 gmPresent (Boolean)**

24 As specified in 9.3.3.3.

25 **9.4.3.4 isSynced (Boolean)**

26 As specified in 9.3.3.4.

27 **9.4.3.5 grandmasterIdentity (ClockIdentity)**

28 As specified in 9.3.3.5.

29 **9.4.4 ClockTargetTriggerGenerate interface definition**

30 The invocation of the ClockTargetTriggerGenerate.invoke function causes the ClockTimeReceiver entity to
31 store the value of the timeReceiverTimeCallback parameter in an internal variable (replacing any previous
32 value of that variable) until the synchronized time, or LocalClock time if gmPresent is FALSE, equals the
33 value of that variable, at which time the ClockTargetTriggerGenerate.result function is invoked with
34 errorCondition = FALSE. If it is not possible to invoke the ClockTargetTriggerGenerate.result function at
35 timeReceiverTimeCallback, e.g., if timeReceiverTimeCallback is earlier than the synchronized time (or
36 LocalClock time if gmPresent is FALSE) when the ClockTargetTriggerGenerate.invoke function is invoked,
37 the ClockTargetTriggerGenerate.result function is invoked with errorCondition = TRUE. Invocation of the
38 ClockTargetTriggerGenerate.invoke function with timeReceiverTimeCallback = 0 (which is earlier than any
39 synchronized time) is used to cancel a pending request.

1 9.5 ClockTargetClockGenerator interface

2 9.5.1 General

3 This interface is used by the ClockTarget entity to request that the ClockTimeReceiver entity deliver a
 4 periodic clock signal of specified period and phase. The ClockTarget entity invokes the
 5 ClockTargetClockGenerator.invoke function to request that the ClockTimeReceiver entity generate the
 6 periodic clock signal. The ClockTimeReceiver entity invokes the ClockTargetClockGenerator.result
 7 function at significant instants of the desired clock signal.

8 9.5.2 ClockTargetClockGenerator.invoke parameters

```
9 ClockTargetClockGenerator.invoke {
10     domainNumber,
11     clockPeriod,
12     timeReceiverTimeCallbackPhase
13 }
```

14 9.5.2.1 domainNumber (UInteger8)

15 This parameter is the **domainNumber** of the ClockTimeReceiver entity that is requested to deliver a periodic
 16 clock signal.

17 9.5.2.2 clockPeriod (TimeInterval)

18 The value of **clockPeriod** is the period between successive invocations of the
 19 ClockTargetClockGenerator.result function. A value that is zero or negative causes any existing periodic
 20 clock signal generated via this application interface to be terminated.

21 9.5.2.3 timeReceiverTimeCallbackPhase (ExtendedTimestamp)

22 The value of **timeReceiverTimeCallbackPhase** describes phase of the generated clock signal by specifying a
 23 point on the timescale in use such that ClockTargetClockGenerator.result invocations occur at synchronized
 24 times that differ from **timeReceiverTimeCallbackPhase** by $n \times \text{clockPeriod}$, where n is an integer.

25 NOTE—The value of **timeReceiverTimeCallbackPhase** can be earlier or later than the synchronized time the
 26 ClockTargetClockGenerator.invoke function is invoked; use of a **timeReceiverTimeCallbackPhase** value in the future
 27 does not imply that the initiation of the periodic clock signal is suppressed until that synchronized time.

28 9.5.3 ClockTargetClockGenerator.result parameters

```
29 ClockTargetClockGenerator.result {
30     domainNumber,
31     timeReceiverTimeCallback
32 }
```

33 9.5.3.1 domainNumber (UInteger8)

34 This parameter is the **domainNumber** of the ClockTimeReceiver entity that is delivering a periodic clock
 35 signal.

36 9.5.3.2 gmPresent

37 As specified in 9.3.3.3.

1 **9.5.3.3 timeReceiverTimeCallback (ExtendedTimestamp)**

2 The value of timeReceiverTimeCallback is the synchronized time of this event.

3 **9.5.3.4 isSynced (Boolean)**

4 As specified in 9.3.3.4.

5 **9.5.3.5 grandmasterIdentity (ClockIdentity)**

6 As specified in 9.3.3.5.

7 **9.6 ClockTargetPhaseDiscontinuity interface**

8 **9.6.1 General**

9 This interface provides discontinuity information, sent from the Grandmaster PTP Instance, to an
10 application within an end station. It is used by the ClockTimeReceiver entity to supply sufficient
11 information to the ClockTarget entity to enable the ClockTarget entity to determine whether a phase or
12 frequency discontinuity has occurred. The ClockTimeReceiver invokes the
13 ClockTargetPhaseDiscontinuity.result function in the SEND_SYNC_INDICATION block of the
14 ClockTimeReceiverSync state machine (see 10.2.13 and Figure 10-9). The invocation occurs when a
15 PortSyncSync structure is received, after the needed information has been computed by the
16 ClockTimeReceiverSync state machine.

17 **9.6.2 ClockTargetPhaseDiscontinuity.result parameters**

18 ClockTargetPhaseDiscontinuity.result {
19 domainNumber,
20 grandmasterIdentity,
21 gmTimeBaseIndicator,
22 lastGmPhaseChange,
23 lastGmFreqChange
24 isSynced,
25 }

26 **9.6.2.1 domainNumber (UInteger8)**

27 This parameter is the `domainNumber` of the ClockTimeReceiver entity that is providing discontinuity
28 information.

29 **9.6.2.2 grandmasterIdentity (ClockIdentity)**

30 If gmPresent (see 10.2.4.13) is TRUE, the value of `grandmasterIdentity` is the ClockIdentity of the current
31 Grandmaster PTP Instance. If gmPresent is FALSE, the value of `grandmasterIdentity` is 0x0.

32 **9.6.2.3 gmTimeBaseIndicator (UInteger16)**

33 The value of `gmTimeBaseIndicator` is the timeBaseIndicator of the current Grandmaster PTP Instance.

1 **9.6.2.4 lastGmPhaseChange (ScaledNs)**

2 The value of the global lastGmPhaseChange parameter (see 10.2.4.16) received from the Grandmaster PTP
3 Instance.

4 **9.6.2.5 lastGmFreqChange (Float64)**

5 The value of lastGmFreqChange parameter (see 10.2.4.17) received from the Grandmaster PTP Instance.

6 **9.6.2.6 isSynced (Boolean)**

7 As specified in 9.3.3.4.

8 **9.6.2.7 grandmasterIdentity (ClockIdentity)**

9 As specified in 9.3.3.5.

10. Media-independent layer specification

2 10.1 Overview

3 10.1.1 General

4 Unless otherwise stated, information in this clause is per domain.

5 10.1.2 Model of operation

6 A PTP Instance contains a best timeTransmitter selection function and a synchronization function. These
 7 functions include PTP Port-specific aspects and aspects associated with the PTP Instance as a whole. The
 8 functions are distributed among a number of entities, which together describe the behavior of a compliant
 9 implementation. The functions are specified by a number of state machines.

10 The model for the media-independent layer of a PTP Instance is shown in Figure 10-1. It includes a single
 11 SiteSync entity, ClockTimeTransmitter entity, and ClockTimeReceiver entity for the PTP Instance as a
 12 whole, plus one PortSync for each PTP Port. The PTP Instance also includes one MD entity for each PTP
 13 Port, which is part of the media-dependent layer. The media-dependent functions performed by the MD
 14 entity are described in the clauses for the respective media. In addition to the entities, Figure 10-1 shows the
 15 information that flows between the entities via the PortSyncSync, MDSyncSend, and MDSyncReceive
 16 structures (see 10.2.2.3, 10.2.2.1, and 10.2.2.2, respectively).

17 The SiteSync, ClockTimeTransmitter, ClockTimeReceiver, and PortSync entities each contain a number of
 18 cooperating state machines, which are described later in this clause (the MD entity state machines are
 19 described in the respective media-dependent clauses). The ClockTimeTransmitter entity receives
 20 information from an external time source, known as a *ClockSource entity* (see 9.2), via an application
 21 interface, and provides the information to the SiteSync entity. The ClockTimeReceiver entity receives
 22 Grandmaster Clock time-synchronization and current Grandmaster PTP Instance information from the
 23 SiteSync entity, and makes the information available to an external application, known as a *clockTarget*
 24 *entity* (see 9.3 through 9.6), via one or more application service interfaces. The SiteSync entity executes the
 25 portion of best timeTransmitter clock selection associated with the PTP Instance as a whole, i.e., it uses the
 26 best timeTransmitter information received on each PTP Port to determine which PTP Port has received the
 27 best information, and updates the states of all the ports (see 10.3.1.1 for a discussion of PTP Port states). It
 28 also distributes synchronization information received on the TimeReceiverPort to all the ports whose state is
 29 TimeTransmitterPort (see 10.3.1.1). The PortSync entity for a TimeReceiverPort receives best
 30 timeTransmitter selection information from the PTP Instance at the other end of the associated link,
 31 compares this to the current best timeTransmitter information that it has, and forwards the result of the
 32 comparison to the Site Sync entity. The PortSync entity for a TimeReceiverPort also receives time-
 33 synchronization information from the MD entity associated with the PTP Port, and forwards it to the
 34 SiteSync entity. The PortSync entity for a TimeTransmitterPort sends best timeTransmitter selection and
 35 time-synchronization information to the MD entity for the PTP Port, which in turn sends the respective
 36 messages.

37 NOTE—This clause does not require a one-to-one correspondence between the PortSync entities of PTP Instances
 38 attached to the same gPTP communication path (see 3.11), i.e., more than two PTP Instances can be attached to a gPTP
 39 communication path that uses a shared medium and meet the requirements of this clause. However, it is possible for a
 40 media-dependent clause to have additional requirements that limit the gPTP communication paths to point-to-point links
 41 for that medium; in this case, each link has exactly two PortSync entities, which can be considered to be in one-to-one
 42 correspondence. One example of this is the full-duplex point-to-point media-dependent layer specified in Clause 11. In
 43 addition, one or more gPTP communication paths can be logically point-to-point but traverse the same shared medium.

1

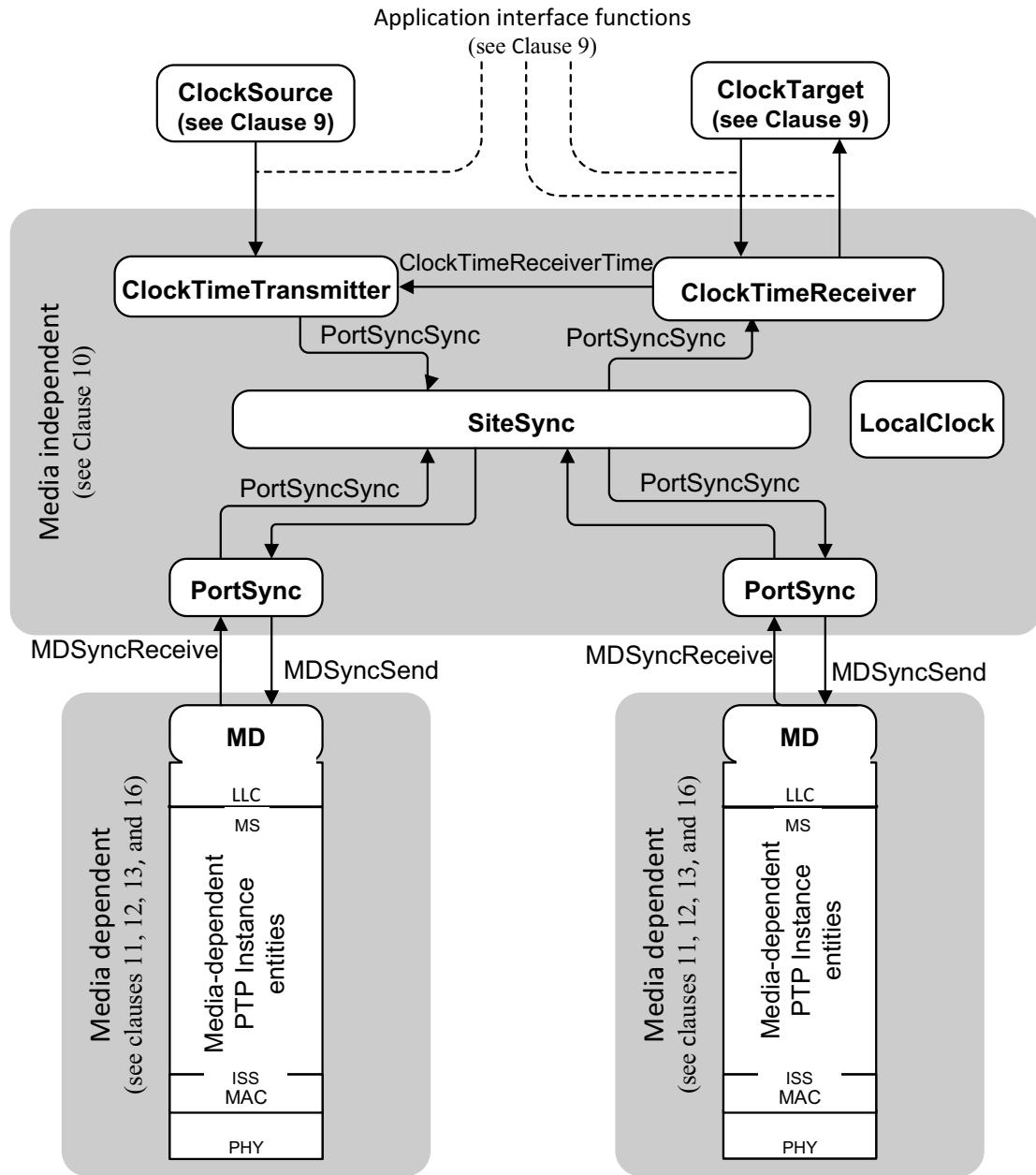


Figure 10-1—Model for media-independent layer of PTP Instance

2 The time-synchronization state machines are described in 10.2. The best timeTransmitter clock selection
3 state machines are described in 10.3. The attributes and format of the Announce message are described in
4 10.5 and 10.6. The timing characterization of the protocol is described in 10.7.

1 10.1.2.1 LocalClock entity

2 The LocalClock entity is a free-running local clock (see 3.16) that provides a common time to the PTP
3 Instance, relative to an arbitrary epoch. A PTP Instance contains a LocalClock entity. The requirements for
4 the LocalClock entity are specified in B.1. All timestamps are taken relative to the LocalClock entity (see
5 8.4.3). The LocalClock entity also provides the value of currentTime (see 10.2.4.12), which is used in the
6 state machines to specify the various timers.

7 NOTE—The epoch for the LocalClock entity can be the time that the PTP Instance is powered on.

8 10.1.3 Grandmaster-capable PTP Instance

9 A PTP Instance may be grandmaster-capable. An implementation may provide the ability to configure a PTP
10 Instance as grandmaster-capable via a management interface.

11 NOTE 1—The managed object gmCapable is read only (see Table 14-1). gmCapable is configured by setting the value
12 of the managed object priority1, which is read/write (see Table 14-1). If the value of priority1 is 255, then

- 13 a) gmCapable is set to FALSE (see 8.6.2.1), and
14 b) The value of the managed object clockClass, which is read only, is set to 255 (see 8.6.2.2).

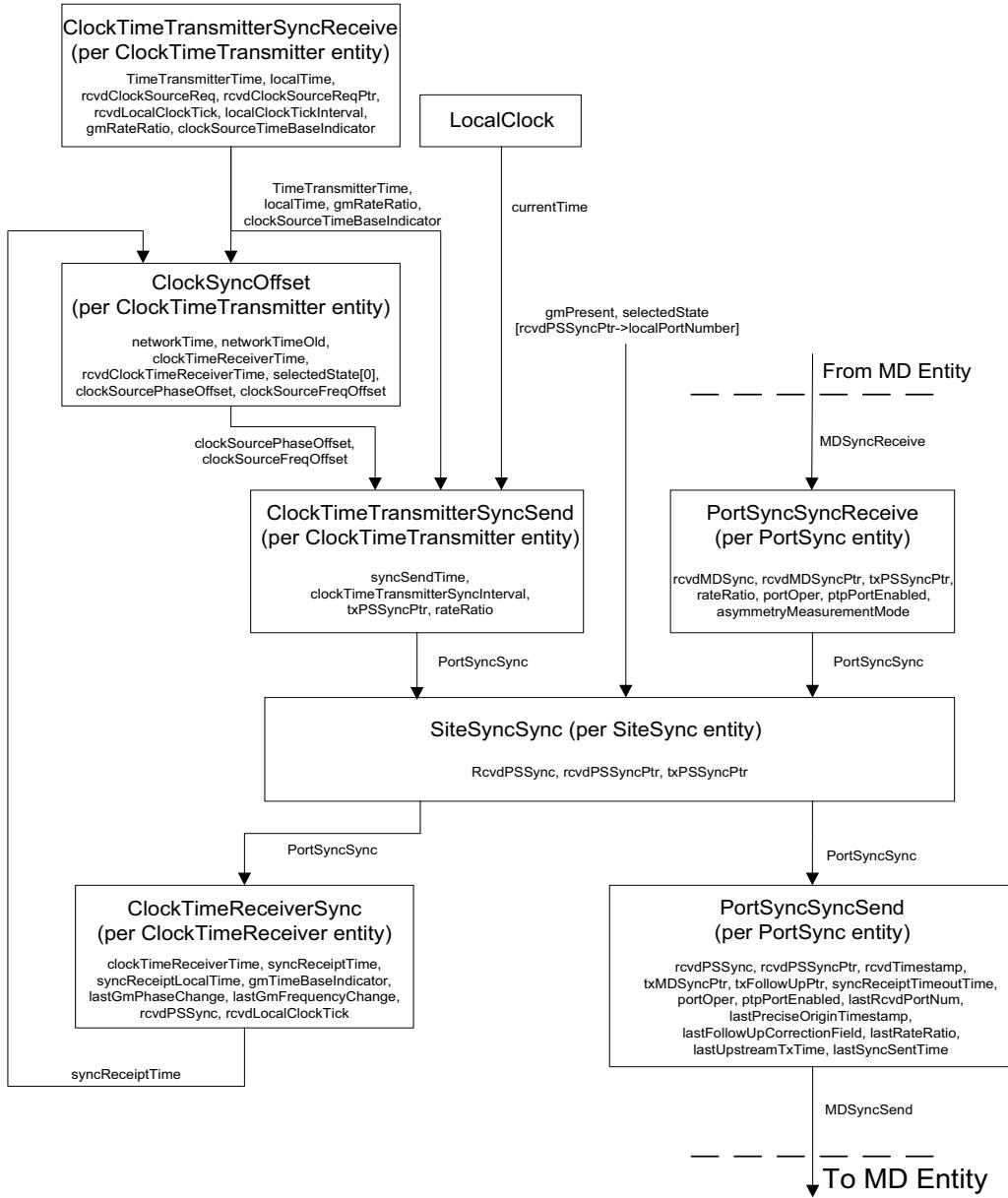
15 NOTE 2—While a PTP Instance that is not grandmaster-capable can never be the Grandmaster PTP Instance of the
16 gPTP domain, such a PTP Instance contains a best timeTransmitter selection function, invokes the best timeTransmitter
17 selection algorithm, and conveys synchronization information received from the current Grandmaster PTP Instance.

18 10.2 Time-synchronization state machines

19 10.2.1 Overview

20 The time-synchronization function in a PTP Instance is specified by a number of cooperating state
21 machines. Figure 10-2 illustrates these state machines, their local variables, their interrelationships, and the
22 global variables and structures used to communicate between them. The figure indicates the interaction
23 between the state machines and the media-dependent layer and LocalClock entity.

24 The ClockTimeTransmitterSyncReceive, *ClockSyncOffset*, and ClockTimeTransmitterSyncSend state
25 machines are optional for PTP Instances that are not grandmaster-capable (see 8.6.2.1 and 10.1.3). These
26 state machines may be present in a PTP Instance that is not grandmaster-capable; however, any information
27 supplied by them, via the ClockTimeTransmitterSyncSend state machine, to the SiteSyncSync state machine
28 is not used by the SiteSyncSync state machine if the PTP Instance is not grandmaster-capable.



Notes:

- a) selectedState for each port and gmPresent are set by Port State Selection state machine (see 10.3.12)
- b) currentTime is a global variable that is always equal to the current time relative to the local oscillator
- c) application interfaces to higher layers are not shown
- d) the ClockTimeTransmitterSyncReceive, ClockTimeTransmitterSyncSend, and ClockTimeTransmitterSyncOffset state machines are optional for PTP Instances that are not grandmaster-capable.

Figure 10-2—Time-synchronization state machines—overview and interrelationships

1 The media-independent layer state machines in Figure 10-2 are as follows:

- 2 a) ClockTimeTransmitterSyncReceive (one instance per PTP Instance): receives
3 ClockSourceTime.invoke functions from the ClockSource entity and notifications of LocalClock
4 entity ticks (see 10.2.4.18), updates timeTransmitterTime, and provides timeTransmitterTime to
5 ClockSyncOffset and ClockTimeTransmitterSyncSend state machines.
- 6 b) ClockSyncOffset (one instance per PTP Instance): receives syncReceiptTime from the
7 ClockTimeReceiver entity and timeTransmitterTime from the ClockTimeTransmitterSyncReceive
8 state machine, computes phase offset and frequency offset between timeTransmitterTime and
9 syncReceiptTime if the PTP Instance is not the Grandmaster PTP Instance, and provides the
10 frequency and phase offsets to the ClockTimeTransmitterSyncSend state machine.
- 11 c) ClockTimeTransmitterSyncSend (one instance per PTP Instance): receives timeTransmitterTime
12 from the ClockTimeTransmitterSyncReceive state machine, receives phase and frequency offset
13 between timeTransmitterTime and syncReceiptTime from the ClockSyncOffset state machine, and
14 provides timeTransmitterTime (i.e., synchronized time) and the phase and frequency offset to the
15 SiteSync entity using a PortSyncSync structure.
- 16 d) PortSyncSyncReceive (one instance per PTP Instance, per PTP Port): receives time-synchronization
17 information from the MD entity of the corresponding PTP Port, computes accumulated rateRatio,
18 computes syncReceiptTimeoutTime, and sends the information to the SiteSync entity.
- 19 e) SiteSyncSync (one instance per PTP Instance): receives time-synchronization information,
20 accumulated rateRatio, and syncReceiptTimeoutTime from the PortSync entity of the current
21 timeReceiver port or from the ClockTimeTransmitter entity; and sends the information to the
22 PortSync entities of all the ports and to the ClockTimeReceiver entity.
- 23 f) PortSyncSyncSend (one instance per PTP Instance, per PTP Port): receives time-synchronization
24 information from the SiteSync entity, requests that the MD entity of the corresponding PTP Port
25 send a time-synchronization event message, receives the syncEventEgressTimestamp for this event
26 message from the MD entity, uses the most recent time-synchronization information received from
27 the SiteSync entity and the timestamp to compute time-synchronization information that will be sent
28 by the MD entity in a general message (e.g., for full-duplex IEEE 802.3 media) or a subsequent
29 event message (e.g., for IEEE 802.11 media), and sends this latter information to the MD entity.
- 30 g) ClockTimeReceiverSync (one instance per PTP Instance): receives time-synchronization
31 information from the SiteSync entity; computes clockTimeReceiverTime and syncReceiptTime; sets
32 syncReceiptLocalTime, GmTimeBaseIndicator, lastGmPhaseChange, and lastGmFreqChange;
33 sends clockTimeReceiverTime to the ClockTimeTransmitter entity; and provides information to the
34 ClockTarget entity (via the ClockTargetPhaseDiscontinuity interface; see 9.6) to enable that entity to
35 determine if a phase or frequency discontinuity has occurred.

36 10.2.2 Data structures communicated between state machines

37 The following subclauses describe the data structures communicated between the time-synchronization state
38 machines.

39 10.2.2.1 MDSyncSend

40 10.2.2.1.1 General

41 This structure contains information that is sent by the PortSync entity of a PTP Port to the MD entity of that
42 PTP Port when requesting that the MD entity cause time-synchronization information to be sent. The
43 structure contains information that reflects the most recent time-synchronization information received by
44 this PTP Instance and is used to determine the contents of the time-synchronization event message and
45 possibly separate general message that will be sent by this PTP Port.

```

1 MDSyncSend {
2     domainNumber,
3     followUpCorrectionField,
4     sourcePortIdentity,
5     logMessageInterval,
6     preciseOriginTimestamp,
7     upstreamTxTime,
8     rateRatio,
9     gmTimeBaseIndicator,
10    lastGmPhaseChange,
11    lastGmFreqChange
12 }

```

13 The members of the structure are defined in the following subclauses.

14 **10.2.2.1.2 domainNumber (UInteger8)**

15 This parameter is the **domainNumber** of the gPTP domain in which this structure is sent.

16 NOTE—The **domainNumber** member is not essential because the state machines that send and receive this structure
17 are per domain, and each state machine implicitly knows the number of the domain in which it operates.

18 **10.2.2.1.3 followUpCorrectionField (ScaledNs)**

19 The followUpCorrectionField contains the accumulated time since the preciseOriginTimestamp was
20 captured by the Grandmaster PTP Instance. This is equal to the elapsed time, relative to the Grandmaster
21 Clock, between the time the Grandmaster PTP Instance sent the received time-synchronization event
22 message, truncated to the nearest nanosecond, and the time at which that event message was sent by the
23 upstream PTP Instance. The followUpCorrectionField is equal to the value of the followUpCorrectionField
24 member of the most recently received PortSyncSync structure from the PortSync entity of this PTP Port (see
25 10.2.2.3.5).

26 **10.2.2.1.4 sourcePortIdentity (PortIdentity)**

27 The sourcePortIdentity is the portIdentity of this PTP Port (see 8.5.2).

28 **10.2.2.1.5 logMessageInterval (Integer8)**

29 The logMessageInterval is the value of currentLogSyncInterval for this PTP Port (see 10.7.2.3).

30 **10.2.2.1.6 preciseOriginTimestamp (Timestamp)**

31 The preciseOriginTimestamp is the sourceTime of the ClockTimeTransmitter entity of the Grandmaster PTP
32 Instance, with any fractional nanoseconds truncated, when the received time-synchronization information
33 was sent by the Grandmaster PTP Instance. The preciseOriginTimestamp is the value of the
34 preciseOriginTimestamp member of the most recently received PortSyncSync structure from the PortSync
35 entity of this PTP Port (see 10.2.2.3.8).

36 **10.2.2.1.7 upstreamTxTime (UScaledNs)**

37 The upstreamTxTime is given by the following equation:

$$38 \quad \text{upstreamTxTime} = \text{syncEventIngressTimestamp} - \frac{\text{meanLinkDelay}}{\text{nrrPdelay}}$$

1 where

2 syncEventIngressTimestamp	corresponds to the receipt of the time-synchronization information at the timeReceiver port of this PTP Instance
3	is defined in 10.2.5.8
4 meanLinkDelay	
5 nrrPdelay	is defined in 11.2.13.13
6 upstreamTxTime	is the value of the upstreamTxTime member of the most recently received PortSyncSync structure from the PortSync entity of this PTP Port (see 10.2.2.3.9)
7	
8	

9 **10.2.2.1.8 rateRatio (Float64)**

10 The rateRatio is the value of the rateRatio member of the most recently received PortSyncSync structure
11 from the PortSync entity of this PTP Port (see 10.2.2.3.10). It is equal to the ratio of the frequency of the
12 Grandmaster Clock to the frequency of the LocalClock entity of this PTP Instance (see 10.2.8.1.4).

13 **10.2.2.1.9 gmTimeBaseIndicator (UInteger16)**

14 The gmTimeBaseIndicator is the timeBaseIndicator of the ClockSource entity of the current Grandmaster
15 PTP Instance. It is set equal to the gmTimeBaseIndicator of the received time-synchronization information.
16 The gmTimeBaseIndicator is the value of the gmTimeBaseIndicator member of the most recently received
17 PortSyncSync structure from the PortSync entity of this PTP Port (see 10.2.2.3.11).

18 **10.2.2.1.10 lastGmPhaseChange (ScaledNs)**

19 The lastGmPhaseChange is the time of the current Grandmaster Clock minus the time of the previous
20 Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the Grandmaster PTP
21 Instance, or the step change in the time of the current Grandmaster Clock at the time of the most recent
22 gmTimeBaseIndicator change. It is set equal to the lastGmPhaseChange of the received time-
23 synchronization information. The lastGmPhaseChange is the value of the lastGmPhaseChange member of
24 the most recently received PortSyncSync structure from the PortSync entity of this PTP Port
25 (see 10.2.2.3.12).

26 **10.2.2.1.11 lastGmFreqChange (Float64)**

27 The lastGmFreqChange is the fractional frequency offset of the current Grandmaster Clock relative to the
28 previous Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the
29 Grandmaster PTP Instance, or relative to itself prior to the last change in gmTimeBaseIndicator. It is set
30 equal to the lastGmFreqChange of the received time-synchronization information. The lastGmFreqChange
31 is the value of the lastGmFreqChange member of the most recently received PortSyncSync structure from
32 the PortSync entity of this PTP Port (see 10.2.2.3.13).

33 **10.2.2.2 MDSyncReceive**

34 **10.2.2.2.1 General**

35 This structure contains information that is sent by the MD entity of a PTP Port to the PortSync entity of that
36 PTP Port. It provides the PortSync entity with timeTransmitter clock timing information and timestamp of
37 receipt of a time-synchronization event message compensated for propagation time on the upstream link.
38 The information is sent to the PortSync entity upon receipt of time-synchronization information by the MD
39 entity of the PTP Port. The information is in turn provided by the PortSync entity to the SiteSync entity. The
40 information is used by the PortSyncSyncReceive state machine of the PortSync entity to compute the rate
41 ratio of the Grandmaster Clock relative to the local clock and is communicated to the SiteSync entity, and

1 then the SiteSync entity communicates it to the other PortSync entities for use in computing timeTransmitter
2 clock timing information.

```

3 MDSyncReceive {
4     domainNumber,
5     followUpCorrectionField,
6     sourcePortIdentity,
7     logMessageInterval,
8     preciseOriginTimestamp,
9     upstreamTxTime,
10    rateRatio,
11    gmTimeBaseIndicator,
12    lastGmPhaseChange,
13    lastGmFreqChange
14 }
```

15 The members of the structure are defined in the following subclauses.

16 **10.2.2.2 domainNumber (UInteger8)**

17 This parameter is the **domainNumber** of the gPTP domain in which this structure is sent.

18 NOTE—The **domainNumber** member is not essential because the state machines that send and receive this structure
19 are per domain, and each state machine implicitly knows the number of the domain in which it operates.

20 **10.2.2.3 followUpCorrectionField (ScaledNs)**

21 The followUpCorrectionField contains the elapsed time, relative to the Grandmaster Clock, between the
22 time the Grandmaster PTP Instance sent the received time-synchronization information, truncated to the
23 nearest nanosecond, and the time at which this information was sent by the upstream PTP Instance.

24 NOTE 1—The sum of followUpCorrectionField and preciseOriginTimestamp is the synchronized time that corresponds
25 to the time the most recently received time-synchronization event message was sent by the upstream PTP Instance.

26 NOTE 2—For a medium that uses separate event and general messages (for example, full-duplex point-to-point media
27 described in 11), the event message corresponding to the most recently received network synchronization information is
28 the event message that corresponds to the most recently received general message. For a medium that places
29 synchronization information based on the event message timestamp in the next event message (for example,
30 IEEE 802.11 media described in Clause 12), the event message corresponding to the most recently received network
31 synchronization information is the previous event message; in this case, the time-synchronization information in the
32 current event message refers to the previous event message.

33 **10.2.2.4 sourcePortIdentity (PortIdentity)**

34 The sourcePortIdentity is the value of the sourcePortIdentity of the time-synchronization event message
35 received by this PTP Port. It is the portIdentity of the upstream TimeTransmitterPort that sent the event
36 message.

37 **10.2.2.5 logMessageInterval (Integer8)**

38 The logMessageInterval is the value of the logMessageInterval of the time-synchronization event message
39 received by this PTP Port. It is the currentLogSyncInterval (see 10.7.2.3) of the upstream
40 TimeTransmitterPort that sent the event message.

1 10.2.2.6 preciseOriginTimestamp (Timestamp)

2 The preciseOriginTimestamp is the sourceTime of the ClockTimeTransmitter entity of the Grandmaster PTP
 3 Instance, with any fractional nanoseconds truncated, when the time-synchronization event message was sent
 4 by the Grandmaster PTP Instance.

5 10.2.2.7 upstreamTxTime (UScaledNs)

6 The upstreamTxTime is given by the following equation:

$$7 \quad \text{upstreamTxTime} = \text{syncEventIngressTimestamp} - \frac{\text{meanLinkDelay}}{\text{nrrPdelay}}$$

8 where

9 syncEventIngressTimestamp corresponds to the receipt of the time-synchronization information at the
 10 timeReceiver port of this PTP Instance (i.e., at this PTP Port)
 11 meanLinkDelay is defined in 10.2.5.8
 12 nrrPdelay is defined in 11.2.13.13

13 NOTE—Media-dependent modifications for increased accuracy might be needed.

14 10.2.2.8 rateRatio (Float64)

15 The rateRatio is the value of rateRatio of the received time-synchronization information. It is equal to the
 16 ratio of the frequency of the Grandmaster Clock to the frequency of the LocalClock entity of the PTP
 17 Instance at the other end of the link attached to this PTP Port, i.e., the PTP Instance that sent the most
 18 recently received time-synchronization event message (see 10.2.8.1.4).

19 10.2.2.9 gmTimeBaseIndicator (UInteger16)

20 The gmTimeBaseIndicator is the timeBaseIndicator of the ClockSource entity of the current Grandmaster
 21 PTP Instance. It is set equal to the gmTimeBaseIndicator of the received time-synchronization information.

22 10.2.2.10 lastGmPhaseChange (ScaledNs)

23 The lastGmPhaseChange is the time of the current Grandmaster Clock minus the time of the previous
 24 Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the Grandmaster PTP
 25 Instance, or the step change in the time of the current Grandmaster Clock at the time of the most recent
 26 gmTimeBaseIndicator change. It is set equal to the lastGmPhaseChange of the received time-
 27 synchronization information.

28 10.2.2.11 lastGmFreqChange (Float64)

29 The lastGmFreqChange is the fractional frequency offset of the current Grandmaster Clock relative to the
 30 previous Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the
 31 Grandmaster PTP Instance, or relative to itself prior to the last change in gmTimeBaseIndicator. It is set
 32 equal to the lastGmFreqChange of the received time-synchronization information.

1 10.2.2.3 PortSyncSync

2 10.2.2.3.1 General

3 This structure is sent by the PortSync and ClockTimeTransmitter entities to the SiteSync entity and also
4 from the SiteSync entity to the PortSync and ClockTimeReceiver entities.

5 When sent from the PortSync or ClockTimeTransmitter entity, it provides the SiteSync entity with
6 timeTransmitter clock timing information, timestamp of receipt of a time-synchronization event message
7 compensated for propagation time on the upstream link, and the time at which sync receipt timeout occurs if
8 a subsequent Sync message is not received by then. The information is used by the SiteSync entity to
9 compute the rate ratio of the Grandmaster Clock relative to the local clock and is communicated to the other
10 PortSync entities for use in computing timeTransmitter clock timing information.

11 When sent from the SiteSync entity to the PortSync or ClockTimeReceiver entity, the structure contains
12 information needed to compute the synchronization information that will be included in respective fields of
13 the time-synchronization event and general messages that will be sent and also to compute the synchronized
14 time that the ClockTimeReceiver entity will supply to the ClockTarget entity.

```
15 PortSyncSync {
16     domainNumber,
17     localPortNumber,
18     syncReceiptTimeoutTime,
19     followUpCorrectionField,
20     sourcePortIdentity,
21     logMessageInterval,
22     preciseOriginTimestamp,
23     upstreamTxTime,
24     rateRatio,
25     gmTimeBaseIndicator,
26     lastGmPhaseChange,
27     lastGmFreqChange
28 }
```

29 The parameters of the PortSyncSync structure are defined in the following subclauses for when the structure
30 is sent from the PortSync or ClockTimeTransmitter entity to the SiteSync entity. If the structure is sent from
31 the SiteSync entity to the PortSync or ClockTimeReceiver entity, the member values are copied from the
32 most recently received PortSyncSync structure where the PTP Port that received this structure has PTP Port
33 state of TimeReceiverPort.

34 10.2.2.3.2 domainNumber (UInteger8)

35 This parameter is the [domainNumber](#) of the gPTP domain in which this structure is sent.

36 NOTE—The [domainNumber](#) member is not essential because the state machines that send and receive this structure
37 are per domain, and each state machine implicitly knows the number of the domain in which it operates.

38 10.2.2.3.3 localPortNumber (UInteger16)

39 If the structure is sent by a PortSync entity, the localPortNumber is the port number of the PTP Port whose
40 PortSync entity sent this structure. If the structure is sent by a ClockTimeTransmitter entity, the
41 localPortNumber is zero.

1 10.2.2.3.4 syncReceiptTimeoutTime (UScaledNs)

2 If the structure is sent by a PortSync entity, the syncReceiptTimeoutTime is the value of the local time
 3 (i.e., the free-running, local clock time) at which sync receipt timeout occurs if a subsequent
 4 time-synchronization event message is not received by that time. If the structure is sent by a
 5 ClockTimeTransmitter entity, the syncReceiptTimeoutTime is FFFFFFFFFFFFFF₁₆ [see item h) in
 6 10.2.9.2.1].

7 10.2.2.3.5 followUpCorrectionField (ScaledNs)

8 If the structure is sent by a PortSync entity, the followUpCorrectionField is the value of the
 9 followUpCorrectionField member of the MDSyncReceive structure whose receipt caused the sending of this
 10 structure (see 10.2.2.2). If the structure is sent by a ClockTimeTransmitter entity, the
 11 followUpCorrectionField is the sub-nanosecond portion of the ClockTimeTransmitter time.

12 10.2.2.3.6 sourcePortIdentity (PortIdentity)

13 If the structure is sent by a PortSync entity, the sourcePortIdentity is the value of the sourcePortIdentity
 14 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.4).
 15 If the structure is sent by a ClockTimeTransmitter entity, the clockIdentity member of the sourcePortIdentity
 16 is the clockIdentity of this PTP Instance, and the portNumber member of the sourcePortIdentity is 0.

17 10.2.2.3.7 logMessageInterval (Integer8)

18 If the structure is sent by a PortSync entity, the logMessageInterval is the value of the logMessageInterval
 19 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.5).
 20 If the structure is sent by a ClockTimeTransmitter entity, the logMessageInterval is the value of
 21 clockTimeTransmitterLogSyncInterval (see 10.7.2.4).

22 10.2.2.3.8 preciseOriginTimestamp (Timestamp)

23 If the structure is sent by a PortSync entity, the preciseOriginTimestamp is the value of the preciseOriginTimestamp
 24 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.6). If the structure is sent by a ClockTimeTransmitter entity, the
 25 preciseOriginTimestamp is the ClockTimeTransmitter time truncated to the next lower nanosecond.

27 10.2.2.3.9 upstreamTxTime (UScaledNs)

28 If the structure is sent by a PortSync entity, the upstreamTxTime is the value of the upstreamTxTime
 29 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.7).
 30 If the structure is sent by a ClockTimeTransmitter entity, the upstreamTxTime is the local clock time
 31 corresponding to the ClockTimeTransmitter time.

32 10.2.2.3.10 rateRatio (Float64)

33 If the structure is sent by a PortSync entity, the rateRatio is the value of the rateRatio member of the
 34 MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.8). It is equal to
 35 the ratio of the frequency of the Grandmaster Clock to the frequency of the LocalClock entity of the PTP
 36 Instance at the other end of the link attached to this PTP Port, i.e., the PTP Instance that sent the most
 37 recently-received time-synchronization event message (see 10.2.8.1.4). If the structure is sent by a
 38 ClockTimeTransmitter entity, the rateRatio is equal to gmRateRatio (see 10.2.4.14).

1 10.2.2.3.11 gmTimeBaseIndicator (UInteger16)

2 If the structure is sent by a PortSync entity, the gmTimeBaseIndicator is the value of the
 3 gmTimeBaseIndicator member of the MDSyncReceive structure whose receipt caused the sending of this
 4 structure (see 10.2.2.9). If the structure is sent by a ClockTimeTransmitter entity, the
 5 gmTimeBaseIndicator is equal to clockSourceTimeBaseIndicator (see 10.2.4.8).

6 10.2.2.3.12 lastGmPhaseChange (ScaledNs)

7 If the structure is sent by a PortSync entity, the lastGmPhaseChange is the value of the lastGmPhaseChange
 8 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.9).
 9 If the structure is sent by a ClockTimeTransmitter entity, the lastGmPhaseChange is equal to
 10 clockSourcePhaseOffset (see 10.2.4.7).

11 10.2.2.3.13 lastGmFreqChange (Float64)

12 If the structure is sent by a PortSync entity, the lastGmFreqChange is the value of the lastGmFreqChange
 13 member of the MDSyncReceive structure whose receipt caused the sending of this structure (see 10.2.2.9).
 14 If the structure is sent by a ClockTimeTransmitter entity, the lastGmFreqChange is equal to
 15 clockSourceFreqOffset (see 10.2.4.6).

16 10.2.3 Overview of global variables used by time synchronization state machines

17 Subclauses 10.2.4 and 10.2.5 define global variables used by time synchronization state machines whose
 18 scopes are as follows:

- 19 — Per PTP Instance (i.e., per domain)
- 20 — Per PTP Instance, per PTP Port
- 21 — Instance used by the Common Mean Link Delay Service (CMLDS) (see 11.2.17) (i.e., variable is
 22 common across all [Link Ports](#))
- 23 — Instance used by CMLDS, per [Link Port](#)

24 Table 10-1 summarizes the scope of each global variable of 10.2.4 and 10.2.5.

Table 10-1—Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per LinkPort
BEGIN	10.2.4.1	Yes	No	Yes	No
clockTimeTransmitterSyncInterval	10.2.4.2	Yes	No	No	No
clockTimeReceiverTime	10.2.4.3	Yes	No	No	No
syncReceiptTime	10.2.4.4	Yes	No	No	No
syncReceiptLocalTime	10.2.4.5	Yes	No	No	No
clockSourceFreqOffset	10.2.4.6	Yes	No	No	No
clockSourcePhaseOffset	10.2.4.7	Yes	No	No	No

Table 10-1—Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5) (continued)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per LinkPort
clockSourceTimeBaseIndicator	10.2.4.8	Yes	No	No	No
clockSourceTimeBaseIndicatorOld	10.2.4.9	Yes	No	No	No
clockSourceLastGmPhaseChange	10.2.4.10	Yes	No	No	No
clockSourceLastGmFreqChange	10.2.4.11	Yes	No	No	No
currentTime	10.2.4.12	Yes	No	No	No
gmPresent	10.2.4.13	Yes	No	No	No
gmRateRatio	10.2.4.14	Yes	No	No	No
gmTimeBaseIndicator	10.2.4.15	Yes	No	No	No
lastGmPhaseChange	10.2.4.16	Yes	No	No	No
lastGmFreqChange	10.2.4.17	Yes	No	No	No
localClockTickInterval	10.2.4.18	Yes	No	No	No
localTime	10.2.4.19	Yes	No	No	No
selectedState	10.2.4.20	Yes	No	No	No
timeTransmitterTime	10.2.4.21	Yes	No	No	No
thisClock	10.2.4.22	Yes	No	Yes	No
parentLogSyncInterval	10.2.4.23	Yes	No	No	No
instanceEnable	10.2.4.24	Yes	No	No	No
syncReceiptTimeoutTime	10.2.4.25	Yes	No	No	No
asCapable	10.2.5.1	No	Yes	No	No
asymmetryMeasurementMode	10.2.5.2	No	Yes	No	Yes
syncReceiptTimeoutTimeInterval	10.2.5.3	No	Yes	No	No
currentLogSyncInterval	10.2.5.4	No	Yes	No	No
initialLogSyncInterval	10.2.5.5	No	Yes	No	No
syncInterval	10.2.5.6	No	Yes	No	No
neighborRateRatio	10.2.5.7	No	Yes ^a	No	No
meanLinkDelay	10.2.5.8	No	Yes ^a	No	Yes
delayAsymmetry	10.2.5.9	No	Yes ^a	No	Yes
computeNeighborRateRatio	10.2.5.10	No	Yes ^a	No	Yes
computeMeanLinkDelay	10.2.5.11	No	Yes ^a	No	Yes
portOper ^a	10.2.5.12	No	Yes	No	Yes
ptpPortEnabled	10.2.5.13	No	Yes	No	No
thisPort	10.2.5.14	No	Yes	No	Yes
syncLocked	10.2.5.15	No	Yes	No	No

Table 10-1—Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5) (continued)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per LinkPort
neighborGptpCapable	10.2.5.16	No	Yes	No	No
syncSlowdown	10.2.5.17	No	Yes	No	No
oldSyncInterval	10.2.5.18	No	Yes	No	No
gPtpCapableMessageSlowdown	10.2.5.19	No	Yes	No	No
gPtpCapableMessageInterval	10.2.5.20	No	Yes	No	No
oldGptpCapableMessageInterval	10.2.5.21	No	Yes	No	No
currentLogGptpCapableMessageInterval	10.2.5.22	No	Yes	No	No
initialLogGptpCapableMessageInterval	10.2.5.23	No	Yes	No	No
syncGrandmasterIdentity	10.2.4.26	Yes	No	No	No
syncStepsRemoved	10.2.4.27	Yes	No	No	No
driftTrackingTlvSupport	10.2.4.28	Yes	No	No	No
rcvdPSSyncCSS	10.2.4.29	Yes	No	No	No
rcvdLocalClockTickCSS	10.2.4.30	Yes	No	No	No
rateRatioDrift	10.2.4.31	Yes	No	No	No

^a There is one instance of this variable per physical port, which is accessible by all PTP Ports and Link Ports associated with the physical port.

1 10.2.4 Per PTP Instance global variables

2 10.2.4.1 **BEGIN:** A Boolean controlled by the system initialization. If BEGIN is true, all state machines, 3 including per-PTP Port state machines, continuously execute their initial state. See Annex E of 4 IEEE Std 802.1Q-2018.

5 10.2.4.2 **clockTimeTransmitterSyncInterval:** A variable containing the mean time interval between 6 successive messages providing time-synchronization information by the ClockTimeTransmitter entity to the 7 SiteSync entity. This value is given by $1000000000 \times 2^{\text{clockTimeTransmitterLogSyncInterval}}$ ns, where 8 clockTimeTransmitterLogSyncInterval is the logarithm to base 2 of the mean time between the successive 9 providing of time-synchronization information by the ClockTimeTransmitter entity (see 10.7.2.4). The data 10 type for clockTimeTransmitterSyncInterval is UScaledNs.

11 10.2.4.3 **ClockTimeReceiverTime:** The synchronized time maintained, at the timeReceiver, at the 12 granularity of the LocalClock entity [i.e., a new value is computed every localClockTickInterval (see 13 10.2.4.18) by the ClockTimeReceiver entity]. The data type for ClockTimeReceiverTime is 14 ExtendedTimestamp.

1 **10.2.4.4 syncReceiptTime:** The synchronized time computed by the ClockTimeReceiver entity at the
2 instant time-synchronization information, contained in a PortSyncSync structure, is received. The data type
3 for syncReceiptTime is ExtendedTimestamp.

4 **10.2.4.5 syncReceiptLocalTime:** The value of currentTime (i.e., the time relative to the LocalClock entity)
5 corresponding to syncReceiptTime. The data type for syncReceiptLocalTime is UScaledNs.

6 **10.2.4.6 clockSourceFreqOffset:** The fractional frequency offset of the ClockSource entity frequency
7 relative to the current Grandmaster Clock frequency. The data type for clockSourceFreqOffset is Float64.

8 **10.2.4.7 clockSourcePhaseOffset:** The time provided by the ClockSource entity, minus the synchronized
9 time. The data type for clockSourcePhaseOffset is ScaledNs.

10 **10.2.4.8 clockSourceTimeBaseIndicator:** A global variable that is set equal to the timeBaseIndicator
11 parameter of the ClockSourceTime.invoke application interface function (see 9.2.2.3), by the
12 ClockTimeTransmitter entity. The parameter timeBaseIndicator of ClockSourceTime.invoke is set by the
13 ClockSource entity and is changed by that entity whenever the time base changes. The data type for
14 clockSourceTimeBaseIndicator is UInteger16.

15 **10.2.4.9 clockSourceTimeBaseIndicatorOld:** A global variable that is set equal to the previous value of
16 clockSourceTimeBaseIndicator. The data type for clockSourceTimeBaseIndicatorOld is UInteger16.

17 **10.2.4.10 clockSourceLastGmPhaseChange:** A global variable that is set equal to the lastGmPhaseChange
18 parameter of the ClockSourceTime.invoke application interface function (see 9.2.2.4). That parameter is set
19 by the ClockSource entity and is changed by that entity whenever the time base changes. The data type for
20 clockSourceLastGmPhaseChange is ScaledNs.

21 **10.2.4.11 clockSourceLastGmFreqChange:** A global variable that is set equal to the lastGmFreqChange
22 parameter of the ClockSourceTime.invoke application interface function (see 9.2.2.5). That parameter is set
23 by the ClockSource entity and is changed by that entity whenever the time base changes. The data type for
24 clockSourceLastGmFreqChange is Float64.

25 **10.2.4.12 currentTime:** The current value of time relative to the LocalClock entity clock. The data type for
26 currentTime is UScaledNs.

27 **10.2.4.13 gmPresent:** A Boolean that indicates whether a grandmaster-capable PTP Instance is present in
28 the domain. If TRUE, a grandmaster-capable PTP Instance is present; if FALSE, a grandmaster-capable PTP
29 Instance is not present.

30 **10.2.4.14 gmRateRatio:** The measured ratio of the frequency of the ClockSource entity to the frequency of
31 the LocalClock entity. The data type for gmRateRatio is Float64.

32 **10.2.4.15 gmTimeBaseIndicator:** The most recent value of gmTimeBaseIndicator provided to the
33 ClockTimeReceiverSync state machine via a PortSyncSync structure. The data type for
34 gmTimeBaseIndicator is UInteger16.

35 **10.2.4.16 lastGmPhaseChange:** The most recent value of lastGmPhaseChange provided to the
36 ClockTimeReceiverSync state machine via a PortSyncSync structure. The data type for lastGmPhaseChange
37 is ScaledNs.

38 **10.2.4.17 lastGmFreqChange:** The most recent value of lastGmFreqChange provided to the
39 ClockTimeReceiverSync state machine via a PortSyncSync structure. The data type for lastGmFreqChange
40 is Float64.

1 **10.2.4.18 localClockTickInterval:** The time interval between two successive significant instants (i.e.,
 2 “ticks”) of the LocalClock entity. The data type for localClockTickInterval is TimeInterval.

3 **10.2.4.19 localTime:** The value of currentTime when the most recent ClockSourceTime.invoke function
 4 (see 9.2) was received from the ClockSource entity, or when the LocalClock entity most recently updated its
 5 time. The data type for localTime is UScaledNs.

6 **10.2.4.20 selectedState:** An Enumeration2 array of length numberPorts+1 (see 8.6.2.8). selectedState[j] is
 7 set equal to the PTP Port State (see Table 10-2) of the PTP Port whose portList index is j.

8 **10.2.4.21 timeTransmitterTime:** The time maintained by the ClockTimeTransmitter entity, based on
 9 information received from the ClockSource and LocalClock entities. The data type for timeTransmitterTime
 10 is ExtendedTimestamp.

11 **10.2.4.22 thisClock:** The clockIdentity of the current PTP Instance. The data type for thisClock is
 12 ClockIdentity.

13 **10.2.4.23 parentLogSyncInterval:** The most recent logMessageInterval value received on the
 14 timeReceiver port. If this PTP Instance is the Grandmaster PTP Instance, then this is the
 15 clockTimeTransmitterLogSyncInterval (see 10.7.2.4). The data type for parentLogSyncInterval is Integer8.

16 **10.2.4.24 instanceEnable:** A per-domain Boolean used to enable gPTP on all ports that are enabled for that
 17 domain (i.e., ports for which portOper and ptPPortEnabled are both TRUE). Setting instanceEnable to
 18 FALSE causes all per-domain state machines to go to the initial state.

19 NOTE—instanceEnable has no effect on the operation of the MDPdelayReq (see 11.2.19) and MDPdelayResp (see
 20 11.2.20) state machines because those state machines are not per domain (i.e., there is a single instance of each of those
 21 state machines, per link, for all domains).

22 **10.2.4.25 syncReceiptTimeoutTime:** The value of the syncReceiptTimeoutTime member of the most
 23 recently received PortSyncSync structure. The data type for syncReceiptTimeoutTime is UScaledNs.

24 **10.2.4.26 syncGrandmasterIdentity:** The clockIdentity carried in the syncGrandmassterIdentity field of
 25 the Drift Tracking TLV carried by the most recently received Sync message (twoStep flag FALSE) or
 26 Follow_Up message (twoStep flag TRUE). If the received Sync or Follow_Up message does not carry a
 27 Drift_Tracking TLV, syncGrandmasterIdentity is set to the null value 0xFFFF FFFF FFFF FFFF (see the
 28 section “Unassigned and NULL EUI values” of the IEEE Registration Authority tutorial “Guidelines for
 29 Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI), and Company ID
 30 (CID)” [B5a]). The data type for syncGrandmasterIdentity is ClockIdentity.

31 **10.2.4.27 syncStepsRemoved:** The value of the syncStepsRemoved field of the Drift_Tracking TLV carried
 32 by the most recently received Sync message (twoStep flag FALSE) or Follow_Up message (twoStep flag
 33 TRUE). If the received Sync or Follow_Up message does not carry a Drift_Tracking TLV,
 34 syncstepsRemoved is set to 0xFFFF. The data type for syncStepsRemoved is UInteger16.

35 **10.2.4.28 driftTrackingTlvSupport:** An indicator of whether the PTP Instance supports the Drift_Tracking
 36 TLV and the feature is enabled. The value is TRUE if the Drift_Tracking TLV is supported and the managed
 37 object driftTrackingTlvSupportEnabled (see 14.7b) is TRUE. The value is FALSE if the Drift_Tracking
 38 TLV is not supported, or if the TLV is supported and the managed object driftTrackingTlvSupportEnabled is
 39 FALSE. The data type for driftTrackingTlvSupport is Boolean.

40 NOTE—The Drift_Tracking TLV is transported only on full-duplex, point-to-point links as specified in Clause 11. Even
 41 if driftTrackingTlvSupport is TRUE, the Drift_Tracking TLV is not transported on links other than full-duplex,
 42 point-to-point links, and is not received by the PTP Ports at the other end of these links.

1 **10.2.4.29 revdPSSyncCSS:** A Boolean variable that is set to TRUE when a PortSyncSync structure is
 2 received from the SiteSyncSync state machine of the SiteSync entity by the ClockTimeReceiverSync state
 3 machine. This variable is reset by the ClockTimeReceiverSync state machine.

4 **10.2.4.30 revdLocalClockTickCSS:** A Boolean variable that is set to TRUE when the LocalClock entity
 5 updates its time. This variable is reset by the ClockTimeReceiverSync state machine.

6 **10.2.4.31 rateRatioDrift:** The value of the rateRatioDrift field of the Drift_Tracking TLV carried by the
 7 most recently received Sync message (twoStep flag FALSE) or Follow_Up message (twoStep flag TRUE).
 8 If the received Sync or Follow_Up message does not carry a Drift_Tracking TLV, rateRatioDrift is set to
 9 0xFFFFFFFF. The data type for rateRatioDrift is Integer32.

10 **10.2.5 Per-port global variables**

11 **10.2.5.1 asCapable:** A Boolean that is TRUE if and only if it is determined that this PTP Instance and the
 12 PTP Instance at the other end of the link attached to this PTP Port can interoperate with each other via the
 13 IEEE 802.1AS protocol. As a result,

- 14 a) This PTP Instance is capable of executing the IEEE 802.1AS protocol,
- 15 b) The PTP Instance at the other end of the link is capable of executing the IEEE 802.1AS protocol,
 16 and
- 17 c) There are no non-IEEE-802.1AS systems in between this PTP Instance and the PTP Instance at the
 18 other end of the link that introduce sufficient impairments that the end-to-end time-synchronization
 19 performance of B.3 cannot be met.

20 The determination of asCapable is different for each medium and is described in the respective media-
 21 dependent clauses.

22 There is one instance of this variable per PTP Instance (i.e., per domain), per PTP Port.

23 NOTE—The per-port global variable asCapableAcrossDomains (see 11.2.13.12) is common across, and accessible by,
 24 all the domains. It is computed by the MDPdelayReq state machine (see 11.2.19). For full-duplex point-to-point links
 25 (see 11), asCapableAcrossDomains is used when setting the instance of asCapable for each domain (for the link in
 26 question).

27 **10.2.5.2 asymmetryMeasurementMode:** A Boolean that contains the value of the managed object
 28 asymmetryMeasurementMode (see 14.8.45). For full-duplex IEEE 802.3 media, the value is TRUE if an
 29 asymmetry measurement is being performed for the link attached to this port and FALSE otherwise. For all
 30 other media, the value is FALSE. There is one instance of this variable for all the domains, i.e., all the PTP
 31 Instances (per port), **and also one instance of this variable for domain 0 if domain 0 is implemented**. The
 32 variable is accessible by all the domains.

33 **10.2.5.3 syncReceiptTimeoutTimeInterval:** The time interval after which sync receipt timeout occurs if
 34 time-synchronization information has not been received during the interval. The value of
 35 syncReceiptTimeoutTimeInterval is equal to syncReceiptTimeout (see 10.7.3.1) multiplied by the
 36 syncInterval (see 10.2.5.6) for the PTP Port at the other end of the link to which this PTP Port is attached.
 37 The value of syncInterval for the PTP Port at the other end of the link is computed from logMessageInterval
 38 of the received Sync message (see 10.6.2.2.14). The data type for syncReceiptTimeoutTimeInterval is
 39 UScaledNs.

40 **10.2.5.4 currentLogSyncInterval:** The current value of the logarithm to base 2 of the mean time interval, in
 41 seconds, between the sending of successive time-synchronization event messages (see 10.7.2.3). This value
 42 is set in the SyncIntervalSetting state machine (see 10.3.18). The data type for currentLogSyncInterval is
 43 Integer8.

1 **10.2.5.5 initialLogSyncInterval:** The initial value of the logarithm to base 2 of the mean time interval, in
 2 seconds, between the sending of successive time-synchronization event messages (see 10.7.2.3). The data
 3 type for initialLogSyncInterval is Integer.

4 **10.2.5.6 syncInterval:** A variable containing the mean time-synchronization event message transmission
 5 interval for the PTP Port. This value is set in the SyncIntervalSetting state machine (see 10.3.18). The data
 6 type for syncInterval is UScaledNs.

7 **10.2.5.7 neighborRateRatio:** The measured ratio of the frequency of the LocalClock entity of the time-
 8 aware system at the other end of the link attached to this port, to the frequency of the LocalClock entity of
 9 this time-aware system. The data type for neighborRateRatio is Float64. There is one instance of this
 10 variable for each domain, i.e., each PTP Instance (per port).

11 **10.2.5.8 meanLinkDelay:** The measured mean propagation delay (see 8.3) on the link attached to this port,
 12 relative to the LocalClock entity of the time-aware system at the other end of the link (i.e., expressed in the
 13 time base of the time-aware system at the other end of the link). The data type for meanLinkDelay is
 14 UScaledNs. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port),
 15 and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by
 16 all the domains.

17 NOTE—The variable meanLinkDelay was named neighborPropDelay in the 2011 edition of this standard.

18 **10.2.5.9 delayAsymmetry:** The asymmetry in the propagation delay on the link attached to this port. If
 19 propagation delay asymmetry is not modeled, then delayAsymmetry is zero. The data type for
 20 delayAsymmetry is ScaledNs. There is one instance of this variable for CMLDS (see 11.2.17), and there is
 21 also one instance of this variable for each domain. The instance of this variable for CMLDS is relative to the
 22 local clock. The instance of this variable for each domain is relative to the grandmaster time base for that
 23 domain. The instance of delayAsymmetry for CMLDS is used where needed in all computations done by the
 24 CMLDS, and also by computations using the syncEgressTimestamp field of the Drift_Tracking TLV (see
 25 11.4.4.2.1) if CMLDS is present. The instance of delayAsymmetry for a domain is used where needed in all
 26 computations done for that domain, and also by computations using the Drift_Tracking TLV if CMLDS is
 27 not present.

28 **10.2.5.10 computeNeighborRateRatio:** A Boolean, set by the LinkDelayIntervalSetting state machine (see
 29 11.2.21), that indicates whether nrrPdelay is computed for this port. There is one instance of this variable for
 30 each domain, i.e., each PTP Instance (per port), and one instance of this variable for CMLDS. If the instance
 31 of this variable for a domain is TRUE, nrrPdelay for that domain is computed.

32 **10.2.5.11 computeMeanLinkDelay:** A Boolean, set by the LinkDelayIntervalSetting state machine (see
 33 11.2.21), that indicates whether meanLinkDelay is to be computed by this port. There is one instance of this
 34 variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for
 35 domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

36 **10.2.5.12 portOper:** A Boolean that is TRUE if and only if the port is up and able to send and receive
 37 messages. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port). The
 38 variable is accessible by all the domains. The term *port* in this definition is a physical port.

39 NOTE 1—portOper is an indicator, and not a control, and reflects the operational status of the underlying medium. It is
 40 not administratively set by gPTP.

41 NOTE 2—The variable portOper corresponds to the variable portEnabled in the 2011 edition of this standard. The
 42 change is reflected in many state machines.

43 NOTE 3—portOper is the same as MAC_Operational (see IEEE Std 802.1AC-2016).

1 NOTE 4—When portOper is referenced for a logical port, the reference is to portOper for the physical port that
2 corresponds to the logical port.

3 **10.2.5.13 ptpPortEnabled:** A Boolean that is administratively set to TRUE if time-synchronization is to be
4 enabled on this PTP Port.

5 NOTE 1—It is expected that the value of ptpPortEnabled is set via the management interface (see 14.8.4). A physical
6 port can be enabled for data transport but not for synchronization transport.

7 NOTE 2—The variable ptpPortEnabled was named pttPortEnabled in the 2011 edition of this standard. Only the name
8 of this variable has changed; the definition and function of this variable are the same as in the 2011 edition. The name
9 change is reflected in many state machines.

10 **10.2.5.14 thisPort:** The portNumber of the current port. The data type for thisPort is UInteger16.

11 **10.2.5.15 syncLocked:** A Boolean, set by the PortSyncSyncSend state machine (see 10.2.12.3), that
12 indicates that this PTP Port, when operating as a timeTransmitter port, shall transmit a Sync as soon as
13 possible after the timeReceiver port received a Sync (ignoring syncInterval). If FALSE, the PTP Port shall
14 use the timing set by syncInterval.

15 **10.2.5.16 neighborGptpCapable:** A Boolean, set by the GptpCapableReceive state machine (see 10.4.2),
16 that indicates that the neighbor of this PTP Port (i.e., the PTP Port at the other end of the link attached to this
17 PTP Port) is capable of invoking gPTP.

18 **10.2.5.17 syncSlowdown:** A Boolean that is set to TRUE if the SyncIntervalSetting state machine (see
19 Figure 10-20 in 10.3.18.3) receives a TLV that requests a larger sync interval (see 10.7.2.3) and FALSE
20 otherwise. When syncSlowdown is set to TRUE, the PortSyncSyncSend state machine (see Figure 10-8)
21 continues to send time synchronization event messages (see 11.4.3, 12.1, 12.2, and 13.3.1) at the old (i.e.,
22 faster) rate until the number of time synchronization event messages equal to syncReceiptTimeout (see
23 10.7.3.1) have been sent, but with the respective time synchronization event message transmission interval
24 field (see 11.4.2.9, 12.7, Figure 12-8, and 13.3.1.2.10) of the time synchronization event message set equal
25 to the new sync interval (i.e., corresponding to the slower rate). After syncReceiptTimeout Sync messages
26 have been sent, subsequent time synchronization event messages are sent at the new (i.e., slower) rate and
27 with the respective time synchronization event message transmission interval field of the time
28 synchronization event message set to the new sync interval. When syncSlowdown is set to FALSE, the
29 PortSyncSyncSend state machine immediately sends time synchronization event messages at the new (i.e.,
30 faster or the same) rate.

31 NOTE—If a receiver of time synchronization event messages (see 11.4.3, 12.1, 12.2, and 13.3.1) requests a slower rate,
32 the receiver will continue to use the upstream sync interval value, which it obtains from the respective time
33 synchronization event message transmission interval field (see 11.4.2.9, 12.7, Figure 12-8, and 13.3.1.2.10) of the
34 received time synchronization event message, until it receives a time synchronization event message where that value has
35 changed. If, immediately after requesting a slower time synchronization event message rate, up to syncReceiptTimeout
36 consecutive time synchronization event messages sent to the receiver are lost, sync receipt timeout could occur if the
37 sender had changed to the slower rate immediately. Delaying the slowing down of the sending rate of time
38 synchronization event messages for syncReceiptTimeout messages prevents this timeout from happening.

39 **10.2.5.18 oldSyncInterval:** The saved value of the previous sync interval, when a new time-
40 synchronization event message transmission interval is requested via a Signaling message that contains a
41 message interval request TLV. The data type for oldSyncInterval is UScaledNs.

42 **10.2.5.19 gPtpCapableMessageSlowdown:** A Boolean that is set to TRUE if the
43 GptpCapableIntervalSetting state machine (see Figure 10-19 in 10.3.17.3) receives a TLV that requests a
44 larger gPTP-capable message interval (see 10.7.2.5) and FALSE otherwise. When
45 gPtpCapableMessageSlowdown is set to TRUE, the GptpCapableTransmit state machine (see Figure 10-21
46 in 10.4.1.3) continues to send Signaling messages containing the gPTP-capable TLV at the old (i.e., faster)
47 rate until a number of Signaling messages containing the gPTP-capable TLV, equal to

1 gPtpCapableReceiptTimeout (see 10.7.3.3), have been sent, but with the logGptpCapableMessageInterval
 2 field of the gPTP-capable TLV (see 10.6.4.5.6) set equal to the new gPTP-capable message interval (i.e.,
 3 corresponding to the slower rate). After gPtpCapableReceiptTimeout Signaling messages containing the
 4 gPTP-capable TLV have been sent, subsequent such Signaling messages are sent at the new (i.e., slower)
 5 rate and with the logGptpCapableMessageInterval field of the gPTP-capable TLV set to the new gPTP-
 6 capable message interval. When gPtpCapableSlowdown is set to FALSE, the GptpCapableTransmit state
 7 machine immediately sends Signaling messages containing the gPTP-capable TLV at the new (i.e., faster or
 8 the same) rate.

9 NOTE—If a receiver of Signaling messages containing the gPTP-capable TLV requests a slower rate, the receiver will
 10 continue to use the old gPTP-capable message interval value in determining, via the GptpCapableReceive state machine
 11 (see 10.4.2), if its neighbor is no longer capable of invoking gPTP, until it has received gPtpCapableReceiptTimeout
 12 such Signaling messages. If, immediately after requesting a slower rate, up to gPtpCapableReceiptTimeout consecutive
 13 Signaling messages, containing the gPTP-capable TLV, sent to the receiver are lost, a declaration that the sender is no
 14 longer capable of invoking gPTP could occur if the sender had changed to the slower rate immediately. Delaying the
 15 slowing down of the sending rate of Signaling messages containing the gPTP-capable TLV for
 16 gPtpCapableReceiptTimeout messages prevents this timeout from happening.

17 **10.2.5.20 gPtpCapableMessageInterval:** A variable whose value is the mean time, in seconds, between the
 18 sending of successive Signaling messages that carry the gPTP-capable TLV (see 10.7.2.5 and 10.6.4.4). The
 19 data type for gPtpCapableMessageInterval is UScaledNs.

20 **10.2.5.21 oldGptpCapableMessageInterval:** The saved value of the previous interval between the sending
 21 of successive Signaling messages that carry the gPTP-capable TLV (see 10.2.5.20), when a new such
 22 interval is requested via a Signaling message that contains a gPTP-capable message interval request TLV.
 23 The data type for oldGptpCapableMessageInterval is UScaledNs.

24 **10.2.5.22 currentLogGptpCapableMessageInterval:** The current value of the logarithm to base 2 of the
 25 mean time interval, in seconds, between the sending of successive Signaling messages that carry the gPTP-
 26 capable TLV (see 10.6.4.4). This value is set in the GptpCapableIntervalSetting state machine (see 10.4.3).
 27 The data type for currentGptpCapableMessageInterval is Integer8.

28 **10.2.5.23 initialLogGptpCapableMessageInterval:** The initial value of the logarithm to base 2 of the
 29 mean time interval, in seconds, between the sending of successive Signaling messages that carry the gPTP-
 30 capable TLV (see 10.6.4.4). The data type for initialLogGptpCapableMessageInterval is Integer8.

31 **10.2.6 Function used by multiple state machines**

32 **10.2.6.1 random():** Returns a uniformly-distributed pseudo-random number whose data type is UIInteger16
 33 (i.e., the function returns a uniformly distributed, pseudo-random integer in the range $[0, 2^{16} - 1]$).

34 **10.2.7 SiteSyncSync state machine**

35 **10.2.7.1 State machine variables**

36 The following variables are used in the state diagram in Figure 10-3 (in 10.2.7.3):

37 **10.2.7.1.1 rcvdPSSyncSSS:** A Boolean variable that notifies the current state machine when a
 38 PortSyncSync structure (see 10.2.2.3) is received from the PortSyncSyncReceive state machine of a
 39 PortSync entity or from the ClockTimeTransmitterSyncSend state machine of the ClockTimeTransmitter
 40 entity. This variable is reset by this state machine.

41 **10.2.7.1.2 rcvdPSSyncPtrSSS:** A pointer to the received PortSyncSync structure indicated by
 42 rcvdPSSyncSSS.

43 **10.2.7.1.3 txPSSyncPtrSSS:** A pointer to the PortSyncSync structure transmitted by the state machine.

1 10.2.7.2 State machine functions

2 **10.2.7.2.1 setPSSyncSend (rcvdPSSyncPtrSSS):** Creates a PortSyncSync structure to be transmitted, and
 3 returns a pointer to this structure. The members are copied from the received PortSyncSync structure
 4 pointed to by rcvdPSSyncPtrSSS.

5 **10.2.7.2.2 txPSSync (txPSSyncPtrSSS):** Transmits a copy of the PortSyncSync structure pointed to by
 6 txPSSyncPtrSSS to the PortSyncSyncSend state machine of each PortSync entity and the
 7 ClockTimeReceiverSync state machine of the ClockTimeReceiver entity of this PTP Instance.

8 10.2.7.3 State diagram

9 The SiteSyncSync state machine shall implement the function specified by the state diagram in Figure 10-3,
 10 the local variables specified in 10.2.7.1, the functions specified in 10.2.7.2, the structure specified in
 11 10.2.2.3, and the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state
 12 machine receives time-synchronization information, accumulated rateRatio, and syncReceiptTimeoutTime
 13 from the PortSync entity (PortSyncSyncReceive state machine) of the current timeReceiver port or from the
 14 ClockTimeTransmitter entity (ClockTimeTransmitterSyncSend state machine). If the information was sent
 15 by a PortSync entity, the state machine also receives the portIdentity of the PTP Port on the upstream PTP
 16 Instance that sent the information to this PTP Instance (if the information was sent by the
 17 ClockTimeTransmitter entity, the portIdentity is that of the ClockTimeTransmitter entity, i.e., it has
 18 clockIdentity equal to the clockIdentity of this PTP Instance and portNumber 0). The state machine sends a
 19 PortSyncSync structure to the PortSync entities of all the ports and to the ClockTimeReceiver entity.

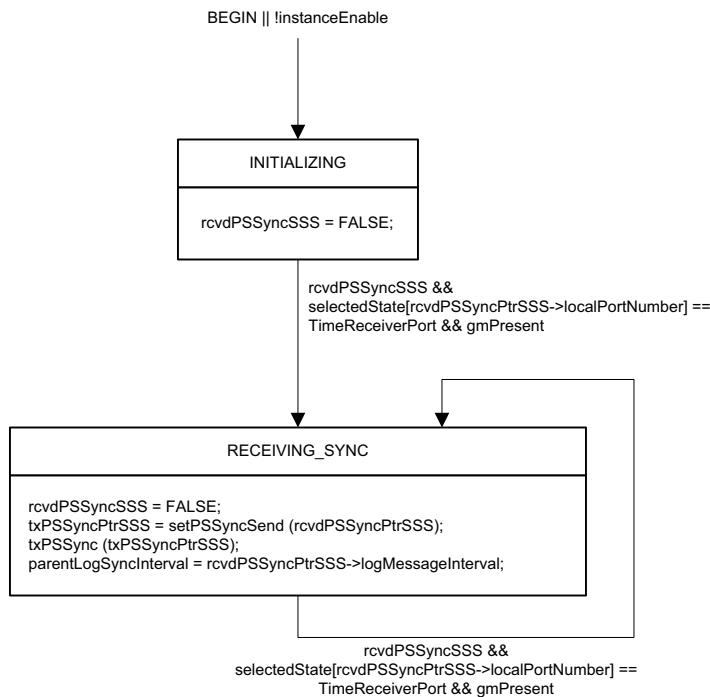


Figure 10-3—SiteSyncSync state machine

1 10.2.8 PortSyncSyncReceive state machine

2 10.2.8.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-4 (in 10.2.8.3):

4 **10.2.8.1.1 rcvdMDSyncPSSR:** A Boolean variable that notifies the current state machine when an
5 MDSyncReceive structure is received from an MD entity of the same PTP Port (see 10.2.2.1). This variable
6 is reset by this state machine.

7 **10.2.8.1.2 rcvdMDSyncPtrPSSR:** A pointer to the received MDSyncReceive structure indicated by
8 rcvdMDSyncPSSR.

9 **10.2.8.1.3 txPSSyncPtrPSSR:** A pointer to the PortSyncSync structure transmitted by the state machine.

10 **10.2.8.1.4 rateRatio:** A Float64 variable that holds the ratio of the frequency of the Grandmaster Clock to
11 the frequency of the LocalClock entity. This frequency ratio is computed by:

- 12 a) Measuring the ratio of the Grandmaster Clock frequency to the LocalClock frequency at the
13 Grandmaster PTP Instance and initializing rateRatio to this value in the
14 ClockTimeTransmitterSyncSend state machine of the Grandmaster PTP Instance and
- 15 b) Accumulating, in the PortSyncSyncReceive state machine of each PTP Instance, the frequency
16 offset of the LocalClock entity of the PTP Instance at the remote end of the link attached to that PTP
17 Port to the frequency of the LocalClock entity of this PTP Instance.

18 **10.2.8.1.5 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 10-4). The
19 data type for TEMP is Integer16.

20 10.2.8.2 State machine functions

21 **10.2.8.2.1 setPSSyncPSSR (rcvdMDSyncPtrPSSR syncReceiptTimeoutTimeInterval, rateRatio):**
22 Creates a PortSyncSync structure to be transmitted, and returns a pointer to this structure. The members are
23 set as follows:

- 24 a) localPortNumber is set equal to thisPort.
- 25 b) domainNumber, followUpCorrectionField, sourcePortIdentity, logMessageInterval,
26 preciseOriginTimestamp, and upstreamTxTime are copied from the received MDSyncReceive
27 structure pointed to by rcvdMDSyncPtrPSSR.
- 28 c) syncReceiptTimeoutTime is set equal to currentTime plus syncReceiptTimeoutTimeInterval (see
29 10.2.5.3).
- 30 d) The function argument rateRatio is set equal to the local variable rateRatio (computed just prior to
31 invoking setPSSyncPSSR (see Figure 10-4)). The rateRatio member of the PortSyncSync structure
32 is then set equal to the function argument rateRatio.

33 **10.2.8.2.2 txPSSyncPSSR (txPSSyncPtrPSSR):** Transmits a copy of the PortSyncSync structure pointed to
34 by txPSSyncPtrPSSR to the SiteSyncSync state machine of this PTP Instance.

10.2.8.3 State diagram

2 The PortSyncSyncReceive state machine shall implement the function specified by the state diagram in
 3 Figure 10-4, the local variables specified in 10.2.8.1, the functions specified in 10.2.8.2, the structures
 4 specified in 10.2.2.1 and 10.2.2.3, and the relevant global variables and functions specified in 10.2.4 through
 5 10.2.6. The state machine receives time-synchronization information, accumulated rateRatio, and
 6 syncReceiptTimeoutTime from the MD entity of the same PTP Port. The state machine adds, to rateRatio,
 7 the fractional frequency offset of the LocalClock entity relative to the LocalClock entity of the upstream
 8 PTP Instance at the remote end of the link attached to this PTP Port. The state machine computes
 9 syncReceiptTimeoutTime. The state machine sends this information to the SiteSync entity (SiteSyncSync
 10 state machine).

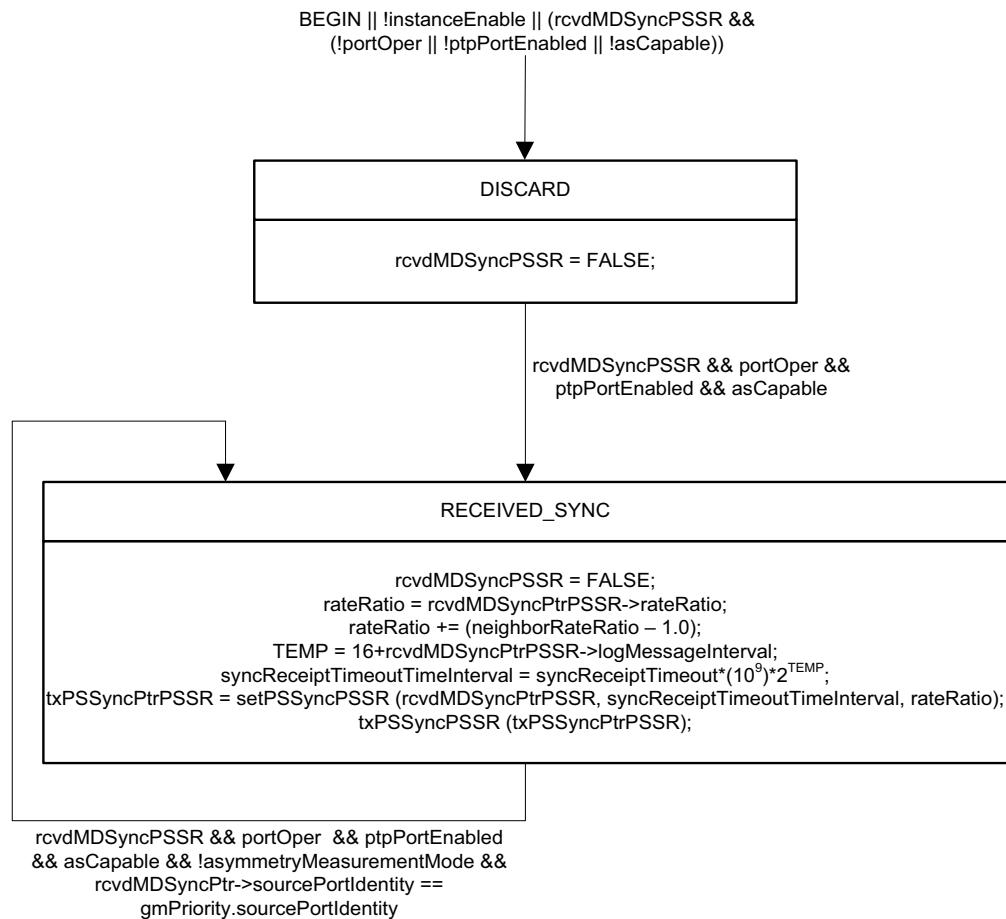


Figure 10-4—PortSyncSyncReceive state machine

1 10.2.9 ClockTimeTransmitterSyncSend state machine

2 10.2.9.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-5 (in 10.2.9.3):

4 **10.2.9.1.1 syncSendTime:** The time in seconds, relative to the LocalClock entity, when synchronization
5 information will next be sent to the SiteSync entity, via a PortSyncSync structure. The data type for
6 syncSendTime is UScaledNs.

7 **10.2.9.1.2 txPSSyncPtrCMSS:** A pointer to the PortSyncSync structure transmitted by the state machine.

8 10.2.9.2 State machine functions

9 **10.2.9.2.1 setPSSyncCMSS (gmRateRatio):** Creates a PortSyncSync structure to be transmitted, and
10 returns a pointer to this structure. The members are set as follows:

- 11 a) localPortNumber is set to 0.
- 12 b) preciseOriginTimestamp is set equal to the timeTransmitterTime, with any fractional nanoseconds
13 truncated.
- 14 c) followUpCorrectionField is set equal to the sum of
 - 15 1) The fractional nanoseconds portion of timeTransmitterTime.fractionalNanoseconds and
 - 16 2) The quantity gmRateRatio × (currentTime – localTime).

17 **NOTE 1**—Both localTime (10.2.4.19) and timeTransmitterTime (10.2.4.21) are updated by the
18 ClockTimeTransmitterSyncReceive state machine (10.2.11). The updates occur both when sourceTime (9.2.2.2) is
19 received from the ClockSource (9.2) - indicated by the variable revdClockSourcReq (10.2.11.1.1) being TRUE—and
20 when the LocalClock entity (10.1.2.1) itself updates by one tick—indicated by the variable revdLocalClockTickCMSR
21 (10.2.11.1.3) being TRUE. timeTransmitterTime is updated by updateTimeTransmitterTime() (see 10.2.11.2.2).
22 localTime is updated by setting it equal to currentTime (Figure 10-7). The result is that localTime is equal to the value of
23 currentTime when timeTransmitterTime was most recently updated. localTime, currentTime, and timeTransmitterTime
24 are per PTP Instance global variables. When they are updated, their values are known to all state machines.

25 **NOTE 2**—It is possible that currentTime (10.2.4.12) and localTime (10.2.4.19) are not equal, despite the statement
26 localTime = currentTime in the RECEIVE_SOURCE_TIME state of the ClockTimeTransmitterSyncReceive state
27 machine (10.2.11). For example, if the ClockTimeTransmitterSyncReceive state machine is implemented as a different
28 module than the ClockTimeTransmitterSyncSend state machine (10.2.9), with timeTransmitterTime (10.2.4.21) and
29 localTime being sent from the ClockTimeTransmitterSyncReceive state machine to the ClockTimeTransmitterSyncSend
30 state machine, then currentTime and localTime will not be equal. In this case, currentTime in c) 2) is the value of the
31 LocalClock entity (10.1.2.1) when the ClockTimeTransmitterSyncSend state machine receives timeTransmitterTime and
32 localTime from the ClockTimeTransmitterSyncReceive state machine.

- 33 d) The clockIdentity member of sourcePortIdentity is set equal to the clockIdentity of this PTP
34 Instance.
- 35 e) The portNumber member of the sourcePortIdentity is set to 0.

36 **NOTE 3**—This quantity and localPortNumber are redundant; both are retained so that the SiteSync entity can process
37 PortSyncSync structures received from a PortSync entity or the ClockTimeTransmitter entity in the same manner.

- 38 f) logMessageInterval is set to clockTimeTransmitterLogSyncInterval.
- 39 g) upstreamTxTime is set equal to localTime.
- 40 h) syncReceiptTimeoutTime is set equal to FFFFFFFFFFFFFF₁₆, which indicates that there is no
41 sync receipt timeout.

42 **NOTE 4**—A ClockTimeTransmitter entity does not receive Sync messages, and there is no notion of sync receipt
43 timeout.

- 1 i) rateRatio is set equal to gmRateRatio.
- 2 j) gmTimeBaseIndicator is set equal to clockSourceTimeBaseIndicator.
- 3 k) lastGmPhaseChange is set equal to clockSourcePhaseOffset.
- 4 l) lastGmFreqChange is set equal to clockSourceFreqOffset.
- 5 m) domainNumber is set equal to the **domainNumber** of this gPTP domain.

6 **10.2.9.2.2 txPSSyncCMSS (txPSSyncPtrCMSS):** Transmits a copy of the PortSyncSync structure pointed
7 to by txPSSyncPtrCMSS to the SiteSync state machine.

8 **10.2.9.2.3 computeClockTimeTransmitterSyncInterval():** Computes the value of
9 clockTimeTransmitterSyncInterval (see 10.2.4.2) as $1000000000 \times 2^{clockTimeTransmitterLogSyncInterval}$ ns,
10 where clockTimeTransmitterLogSyncInterval is the minimum currentLogSyncInterval value, taken over all
11 the PTP Ports of the PTP Instance (see 10.7.2.4).

12 **10.2.9.3 State diagram**

13 The ClockTimeTransmitterSyncSend state machine shall implement the function specified by the state
14 diagram in Figure 10-5, the local variables specified in 10.2.9.1, the functions specified in 10.2.9.2, the
15 structure specified in 10.2.2.3, and the relevant global variables and functions specified in 10.2.4 through
16 10.2.6. The state machine receives timeTransmitterTime and clockSourceTimeBaseIndicator from the
17 ClockTimeTransmitterSyncReceive state machine, and phase and frequency offset between
18 timeTransmitterTime and syncReceiptTime from the **ClockSyncOffset** state machine. It provides
19 timeTransmitterTime (i.e., synchronized time) and the phase and frequency offset to the SiteSync entity via
20 a PortSyncSync structure.

21 The ClockTimeTransmitterSyncSend state machine is optional for PTP Instances that are not grandmaster-
22 capable (see 8.6.2.1, 10.1.3, and 10.2.1). This state machine may be present in a PTP Instance that is not
23 grandmaster-capable; however, any information supplied by it to the SiteSyncSync state machine is not used
24 by the SiteSyncSync state machine if the PTP Instance is not grandmaster-capable.

1

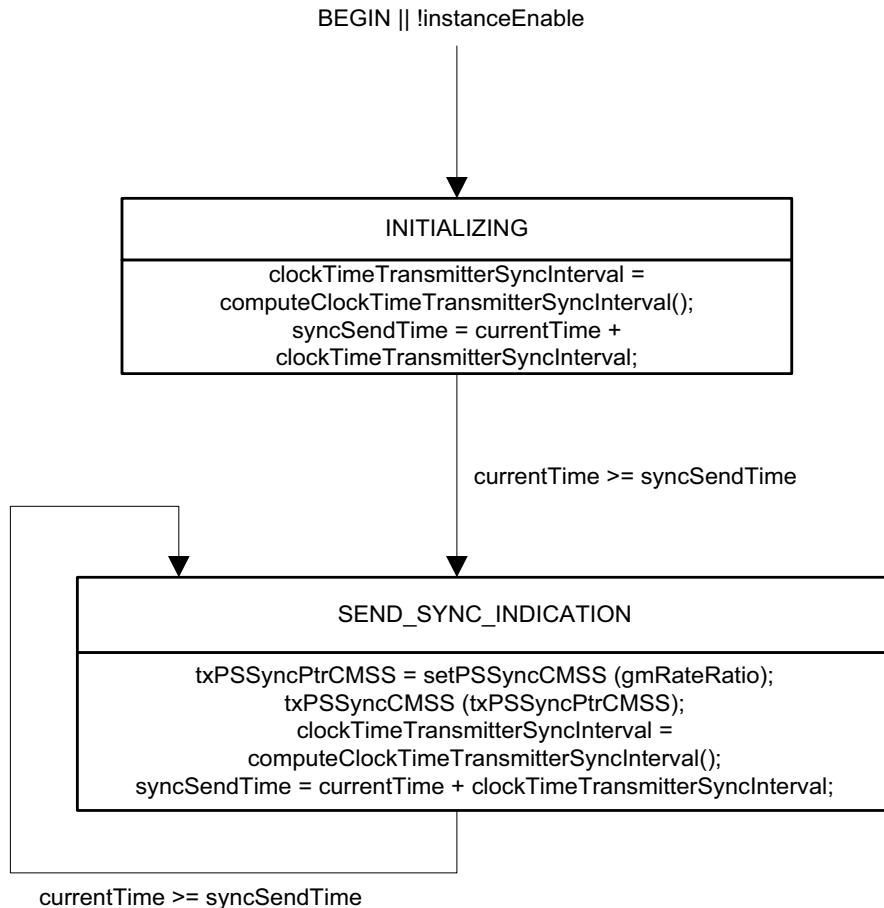


Figure 10-5—ClockTimeTransmitterSyncSend state machine

2 10.2.10 ClockSyncOffset state machine

3 10.2.10.1 State machine variables

4 The following variable is used in the state diagram in Figure 10-6 (in 10.2.10.3):

5 **10.2.10.1.1 rcvdSyncReceiptTime:** A Boolean variable that notifies the current state machine that 6 syncReceiptTime has been updated by the ClockTimeReceiver entity. This variable is reset by this state 7 machine.

8 10.2.10.2 State machine functions

9 **10.2.10.2.1 computeClockSourceFreqOffset():** Computes and returns `clockSourceFreqOffset` (see 10 10.2.4.6), using successive values of `timeTransmitterTime` computed by the 11 `ClockTimeTransmitterSyncReceive` state machine (see 10.2.11) and successive values of `syncReceiptTime` 12 computed by the `ClockTimeReceiverSync` state machine (see 10.2.13). The data type for the returned value

1 is Float64. Any scheme that uses this information to compute clockSourceFreqOffset is acceptable as long
2 as the performance requirements specified in B.2.4 are met.

3 NOTE—As one example, clockSourceFreqOffset can be estimated as the ratio of the duration of a time interval
4 measured by the ClockSource entity to the duration of the same time interval computed from ClockTimeReceiverTime
5 values, minus 1.

6 10.2.10.3 State diagram

7 The **ClockSyncOffset** state machine shall implement the function specified by the state diagram in
8 Figure 10-6, the local variable specified in 10.2.10.1, the function specified in 10.2.10.2, and the relevant
9 global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives
10 syncReceiptTime from the ClockTimeReceiverSync state machine and timeTransmitterTime from the
11 ClockTimeTransmitterSyncReceive state machine. It computes clockSourcePhaseOffset and
12 clockSourceFrequency offset if this PTP Instance is not currently the Grandmaster PTP Instance, i.e., if
13 selectedState[0] is equal to PassivePort.

14 The **ClockSyncOffset** state machine is optional for PTP Instances that are not grandmaster-capable (see
15 8.6.2.1, 10.1.3, and 10.2.1). This state machine may be present in a PTP Instance that is not grandmaster-
16 capable; however, any information supplied by it, via the ClockTimeTransmitterSyncSend state machine, to
17 the SiteSyncSync state machine is not used by the SiteSyncSync state machine if the PTP Instance is not
18 grandmaster-capable.

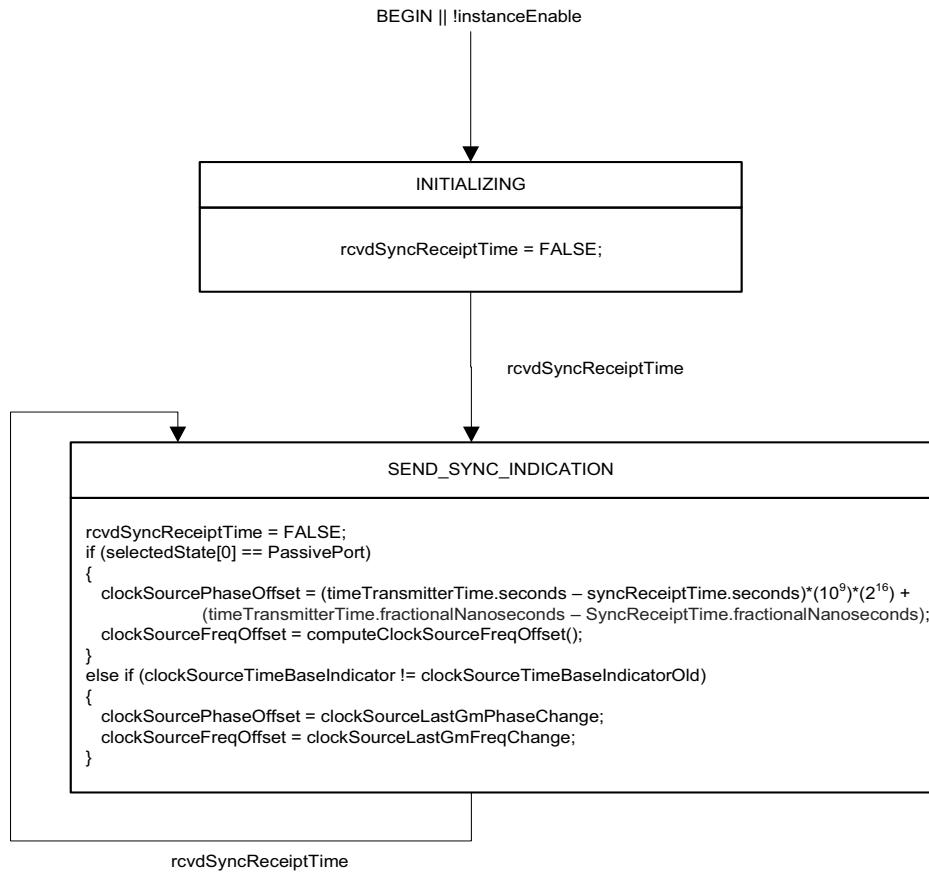


Figure 10-6—ClockSyncOffset state machine

2 10.2.11 ClockTimeTransmitterSyncReceive state machine

3 10.2.11.1 State machine variables

4 The following variables are used in the state diagram in Figure 10-7 (in 10.2.11.3):

5 **10.2.11.1.1 rcvdClockSourceReq:** A Boolean variable that notifies the current state machine when 6 sourceTime is received from the ClockSource entity, due to the `ClockSourceTime.invoke` primitive having 7 been invoked at that entity. This variable is reset by this state machine.

8 **10.2.11.1.2 rcvdClockSourceReqPtr:** A pointer to the received `ClockSourceTime.invoke` function 9 parameters.

10 **10.2.11.1.3 rcvdLocalClockTickCMSR:** A Boolean variable that notifies the current state machine when 11 the LocalClock entity updates its time. This variable is reset by this state machine.

1 10.2.11.2 State machine functions

2 **10.2.11.2.1 computeGmRateRatio():** Computes gmRateRatio (see 10.2.4.14), using values of sourceTime
 3 conveyed by successive ClockSourceTime.invoke functions (see 9.2.2.1), and corresponding values of
 4 localTime (see 10.2.4.19). Any scheme that uses this information, along with any other information
 5 conveyed by the successive ClockSourceTime.invoke functions and corresponding values of localTime, to
 6 compute gmRateRatio is acceptable as long as the performance requirements specified in B.2.4 are met.

7 NOTE—As one example, gmRateRatio can be estimated as the ratio of the elapsed time of the ClockSource entity that
 8 supplies time to this PTP Instance, to the elapsed time of the LocalClock entity of this PTP Instance. This ratio can be
 9 computed for the time interval between a received ClockSourceTime.invoke function and a second received
 10 ClockSourceTime.invoke function some number of ClockSourceTime.invoke functions later, i.e.,

$$11 \quad \frac{\text{ClockSource.invoke.sourceTime}_N - \text{ClockSource.invoke.sourceTime}_0}{\text{localTime}_N - \text{localTime}_0}$$

12 where the successive received ClockSourceTime.invoke functions are indexed from 0 to N , with the first such function
 13 indexed as 0, and localTime_j is the value of localTime when the ClockSourceTime.invoke function whose index is j is
 14 received.

15 **10.2.11.2.2 updateTimeTransmitterTime():** Updates the global variable timeTransmitterTime (see
 16 10.2.4.21), based on information received from the ClockSource and LocalClock entities. It is the
 17 responsibility of the application to filter timeTransmitter times appropriately. As one example,
 18 timeTransmitterTime can be set equal to the sourceTime member of the ClockSourceTime.invoke function
 19 when this function is invoked at the ClockSource entity and can be incremented by localClockTickInterval
 20 (see 10.2.4.18) multiplied by gmRateRatio (see 10.2.4.14) when rcvdLocalClockTickCMSR is TRUE.

21 10.2.11.3 State diagram

22 The ClockTimeTransmitterSyncReceive state machine shall implement the function specified by the state
 23 diagram in Figure 10-7, the local variables specified in 10.2.11.1, the functions specified in 10.2.11.2, and
 24 the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state machine updates the
 25 global variable timeTransmitterTime with information received from the ClockSource entity via the
 26 ClockSourceTime.invoke function and information received from the LocalClock entity. It also computes
 27 gmRateRatio, i.e., the ratio of the ClockSource entity frequency and the LocalClock entity frequency.

28 The ClockTimeTransmitterSyncReceive state machine is optional for PTP Instances that are not
 29 grandmaster-capable (see 8.6.2.1, 10.1.3, and 10.2.1). This state machine may be present in a PTP Instance
 30 that is not grandmaster-capable; however, any information supplied by it, via the
 31 ClockTimeTransmitterSyncSend state machine, to the SiteSyncSync state machine is not used by the
 32 SiteSyncSync state machine if the PTP Instance is not grandmaster-capable.

1

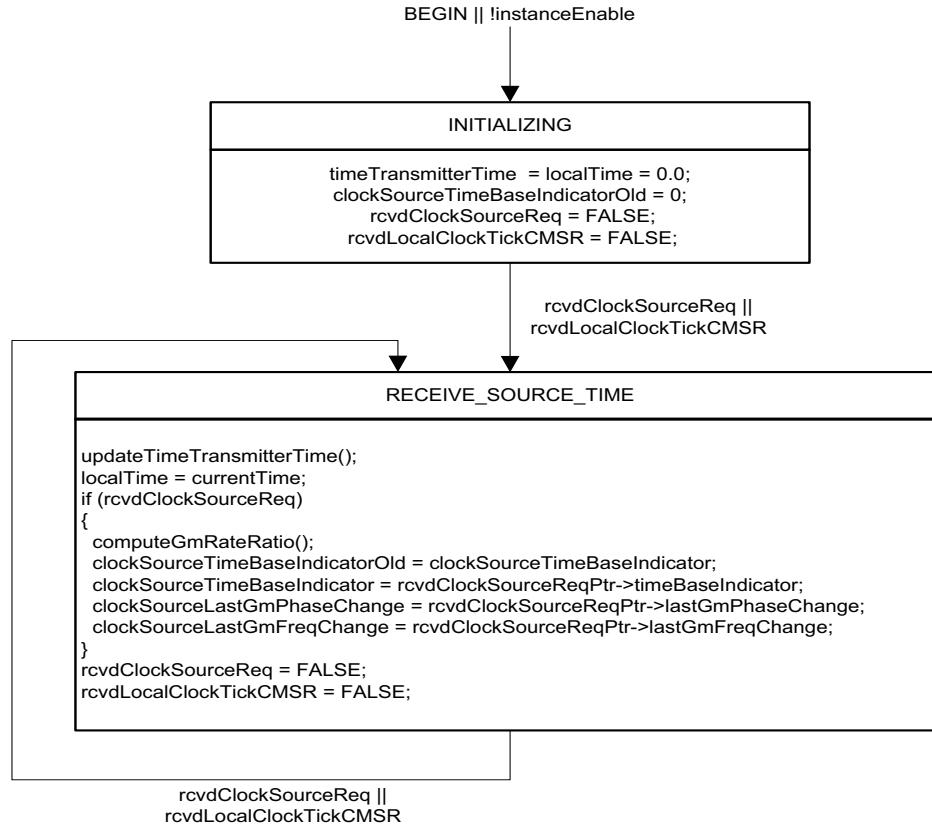


Figure 10-7—ClockTimeTransmitterSyncReceive state machine

2 10.2.12 PortSyncSyncSend state machine

3 10.2.12.1 State machine variables

4 The following variables are used in the state diagram in Figure 10-8 (in 10.2.12.3):

5 **10.2.12.1.1 rcvdPSSyncPSSS:** A Boolean variable that notifies the current state machine when a 6 PortSyncSync structure is received from the SiteSyncSync state machine of the SiteSync entity of the PTP 7 Instance (see 10.2.2.3). This variable is reset by this state machine.

8 **10.2.12.1.2 rcvdPSSyncPtrPSSS:** A pointer to the received PortSyncSync structure indicated by 9 rcvdPSSyncPSSS.

10 **10.2.12.1.3 lastPreciseOriginTimestamp:** The preciseOriginTimestamp member of the most recently 11 received PortSyncSync structure. The data type for lastPreciseOriginTimestamp is Timestamp.

12 **10.2.12.1.4 lastFollowUpCorrectionField:** The followUpCorrectionField member of the most recently 13 received PortSyncSync structure. The data type for lastFollowUpCorrectionField is ScaledNs.

1 **10.2.12.1.5 lastRateRatio:** The rateRatio member of the most recently received PortSyncSync structure.
 2 The data type for lastRateRatio is Float64.

3 **10.2.12.1.6 lastUpstreamTxTime:** The upstreamTxTime member of the most recently received
 4 PortSyncSync structure. The data type for lastUpstreamTxTime is UScaledNs.

5 **10.2.12.1.7 lastSyncSentTime:** The value of currentTime (i.e., the time relative to the LocalClock entity)
 6 when the most recent MDSyncSend structure was sent. The data type for lastSyncSentTime is UScaledNs.

7 NOTE—lastSyncSentTime is the time the abstract MDSyncSend structure was sent, NOT the time the corresponding
 8 Sync message (or equivalent) was sent on a physical link.

9 **10.2.12.1.8 lastRcvdPortNum:** The portNumber of the PTP Port on which time-synchronization
 10 information was most recently received. The data type for lastRcvdPortNum is UInteger16.

11 **10.2.12.1.9 lastGmTimeBaseIndicator:** The gmTimeBaseIndicator of the most recently received
 12 PortSyncSync structure. The data type for lastGmTimeBaseIndicator is UInteger16.

13 **10.2.12.1.10 lastGmPhaseChangePSSS:** The lastGmPhaseChange of the most recently received
 14 PortSyncSync structure. The data type for lastGmPhaseChange is ScaledNs.

15 **10.2.12.1.11 lastGmFreqChangePSSS:** The lastGmFreqChange of the most recently received
 16 PortSyncSync structure. The data type for lastGmFreqChange is Float64.

17 **10.2.12.1.12 txMDSyncPtr:** A pointer to the MDSyncSend structure sent to the MD entity of this PTP Port.

18 **10.2.12.1.13 numberSyncTransmissions:** A count of the number of consecutive Sync message
 19 transmissions after the SyncIntervalSetting state machine (see Figure 10-20) has set syncSlowdown
 20 (see 10.2.5.17) to TRUE. The data type for numberSyncTransmissions is UInteger8.

21 **10.2.12.1.14 interval1:** A local variable that holds either syncInterval or oldSyncInterval. The data type for
 22 interval1 is UScaledNs.

23 **10.2.12.2 State machine functions**

24 **10.2.12.2.1 setMDSync():** Creates an MDSyncSend structure, and returns a pointer to this structure. The
 25 members are set as follows:

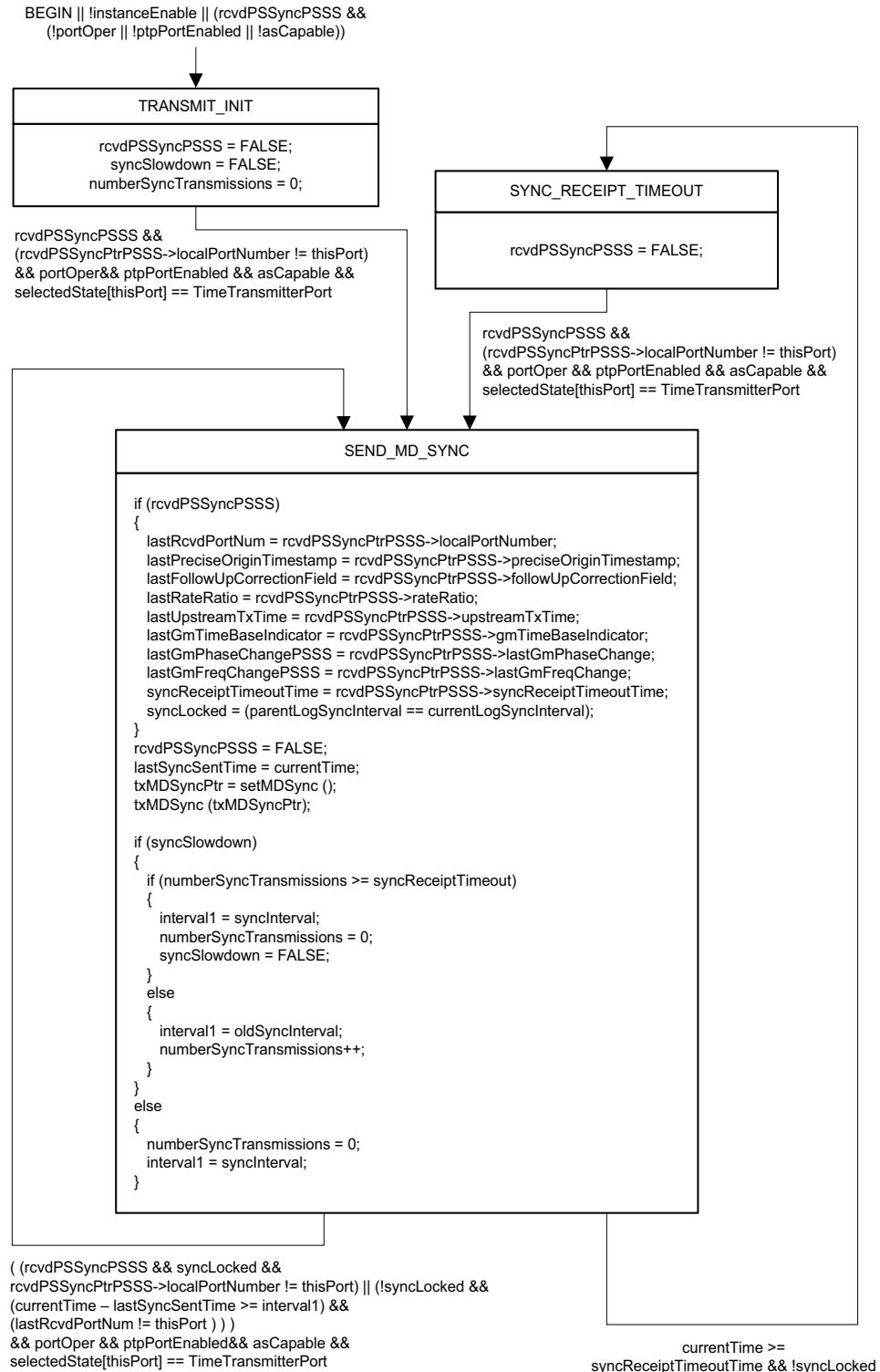
- 26 a) sourcePortIdentity is set to the portIdentity of this PTP Port (see 8.5.2).
- 27 b) logMessageInterval is set equal to the value of currentLogSyncInterval for this PTP Port
 (see 10.7.2.3).
- 29 c) preciseOriginTimestamp is set equal to lastPreciseOriginTimestamp (see 10.2.12.1.3).
- 30 d) rateRatio is set equal to lastRateRatio (see 10.2.12.1.5).
- 31 e) followUpCorrectionField is set equal to lastFollowUpCorrectionField (see 10.2.12.1.4).
- 32 f) upstreamTxTime is set equal to lastUpstreamTxTime (see 10.2.12.1.6).
- 33 g) gmTimeBaseIndicator is set to lastGmTimeBaseIndicator (see 10.2.12.1.9).
- 34 h) lastGmPhaseChange (structure member) is set to lastGmPhaseChangePSSS (see 10.2.12.1.10).
- 35 i) lastGmFreqChange (structure member) is set to lastGmFreqChangePSSS (see 10.2.12.1.11).
- 36 j) domainNumber is set equal to the **domainNumber** of this gPTP domain (see 8.1).

37 **10.2.12.2.2 txMDSync(txMDSyncPtr):** Transmits the MDSyncSend structure pointed to by txMDSyncPtr,
 38 to the MD entity of this PTP Port.

10.2.12.3 State diagram

2 The PortSyncSyncSend state machine shall implement the function specified by the state diagram in
3 Figure 10-8, the local variables specified in 10.2.12.1, the functions specified in 10.2.12.2, the structures
4 specified in 10.2.2.1 through 10.2.2.3, and the relevant global variables and functions specified in 10.2.4
5 through 10.2.6. The state machine receives time-synchronization information from the SiteSyncSync state
6 machine, corresponding to the receipt of the most recent synchronization information on either the
7 timeReceiver port, if this PTP Instance is not the Grandmaster PTP Instance, or from the
8 ClockTimeTransmitterSyncSend state machine, if this PTP Instance is the Grandmaster PTP Instance. The
9 state machine causes time-synchronization information to be sent to the MD entity if this PTP Port is a
10 TimeTransmitterPort.

1

**Figure 10-8—PortSyncSyncSend state machine**

1 10.2.13 ClockTimeReceiverSync state machine

2 10.2.13.1 State machine variables

3 The following **variable is** used in the state diagram in Figure 10-9 (in 10.2.13.3):

4 **10.2.13.1.1 rcvdPSSyncPtrCSS:** A pointer to the received PortSyncSync structure.

5 10.2.13.2 State machine functions

6 **10.2.13.2.1 updateTimeReceiverTime():** Updates the global variable `clockTimeReceiverTime` (see
7 10.2.4.3), based on information received from the SiteSync and LocalClock entities. It is the responsibility
8 of the application to filter timeReceiver times appropriately (see B.3 and B.4 for examples). As one
9 example, `clockTimeReceiverTime` can be:

10 a) Set to `syncReceiptTime` at every LocalClock update immediately after a PortSyncSync structure is
11 received, and

12 b) Incremented by `localClockTickInterval` (see 10.2.4.18) multiplied by the `rateRatio` member of the
13 previously received PortSyncSync structure during all other LocalClock updates.

14 If no PTP Instance is grandmaster-capable, i.e., `gmPresent` is FALSE, then `clockTimeReceiverTime` is set to
15 the time provided by the LocalClock. This function is invoked when `rcvdLocalClockTickCSS` is TRUE.

16 **10.2.13.2.2 invokeApplicationInterfaceFunction (functionName):** Invokes the application interface
17 function whose name is `functionName`. For the ClockTimeReceiverSync state machine, `functionName` is
18 `clockTargetPhaseDiscontinuity.result` (see 9.6.2).

19 10.2.13.3 State diagram

20 The ClockTimeReceiverSync state machine shall implement the function specified by the state diagram in
21 Figure 10-9, the local variables specified in 10.2.13.1, the functions specified in 10.2.13.2, and the relevant
22 global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives a
23 PortSyncSync structure from the SiteSyncSync state machine. It computes `syncReceiptTime` and
24 `clockTimeReceiverTime`, and sets `syncReceiptLocalTime` (i.e., the time relative to the LocalClock entity
25 corresponding to `syncReceiptTime`), `GmTimeBaseIndicator`, `lastGmPhaseChange`, and `lastGmFreqChange`.
26 It provides `clockTimeReceiverTime` to the `ClockSyncOffset` state machine, and provides information to the
27 `ClockTarget` entity (via the `ClockTargetPhaseDiscontinuity` interface; see 9.6) to enable that entity to
28 determine if a phase or frequency discontinuity has occurred.

29 The per-PTP Port global variables used in the ClockTimeReceiverSync state machine are determined based
30 on `rcvdPSSyncPtrCSS->localPortNumber`, as follows:

31 a) If `rcvdPSSyncPtrCSS->localPortNumber > 0`, the per-PTP Port global variables of PTP Port number
32 `rcvdPSSyncPtrCSS->localPortNumber` are used.

33 b) If `rcvdPSSyncPtrCSS->localPortNumber == 0`, the values of the used per-PTP Port global variables
34 are fixed as follows:

35 1) `meanLinkDelay = 0`

36 2) `delayAsymmetry = 0`

37 3) `nrrPdelay = 1.0`

1

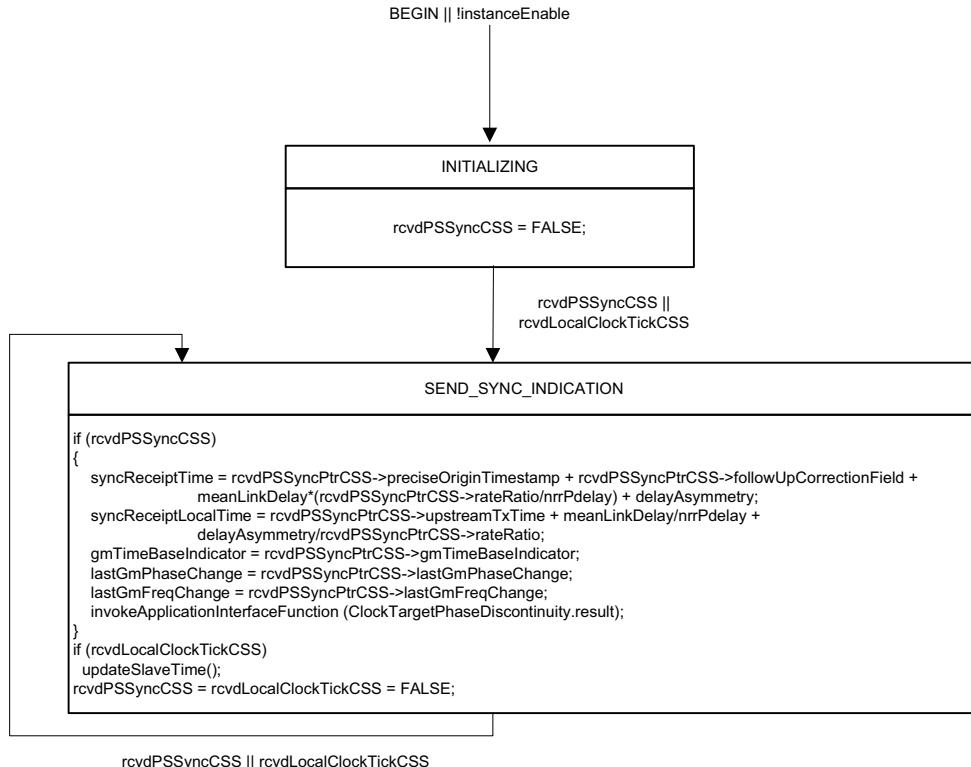


Figure 10-9—ClockTimeReceiverSync state machine

**2 10.3 Best timeTransmitter clock selection, external port configuration, and
 3 announce interval setting state machines**

4 10.3.1 Best timeTransmitter clock selection and external port configuration overview

5 10.3.1.1 General

6 There are two methods for setting the Grandmaster PTP Instance and time-synchronization spanning tree for
 7 a gPTP domain:

- 8 a) The BTCA is used to determine the Grandmaster PTP Instance for a gPTP domain and construct the
 9 time-synchronization spanning tree with that Grandmaster PTP Instance as the root. In this case, the
 10 network is configured automatically, i.e., the PTP Port states are set, using the results of the BTCA.
- 11 b) The PTP Port states are configured to force a desired Grandmaster PTP Instance and to construct a
 12 desired time-synchronization spanning tree with the Grandmaster PTP Instance as the root.

13 The PTP Port state definitions are given in Table 10-2.

- 14 The per PTP Instance global variable `externalPortConfigurationEnabled` indicates whether method a) or b)
 15 is used; a value of TRUE indicates method b), and a value of FALSE indicates method a) (see 10.3.9.24).
 16 The data type of `externalPortConfigurationEnabled` is Boolean. Method a) is implemented and is the default

1 mode of operation (i.e., externalPortConfigurationEnabled is FALSE) on domain 0 to maintain backward
2 compatibility. For domains other than domain 0, the following statements apply:

- 3 c) At least one of the possibilities [method a) or b)] is implemented.
- 4 d) Both possibilities can be implemented.
- 5 e) If both possibilities are implemented, the default value of externalPortConfigurationEnabled is
6 FALSE.

7 Once an Announce message is transmitted by a PTP Port, subsequent timing information (see 7.4)
8 transmitted by that PTP Port is derived from the Grandmaster PTP Instance indicated in that Announce
9 message.

Table 10-2—PTP Port state definitions

PTP Port state	Description
TimeTransmitterPort	Any PTP Port, P, of the PTP Instance that is closer to the root than any other PTP Port of the gPTP communication path connected to P.
TimeReceiverPort	The one PTP Port of the PTP Instance that is closest to the root PTP Instance. If the root is grandmaster-capable, the TimeReceiverPort is also closest to the Grandmaster PTP Instance. The PTP Instance does not transmit Sync or Announce messages on the TimeReceiverPort.
PassivePort	Any PTP Port of the PTP Instance whose PTP Port state is not TimeTransmitterPort, TimeReceiverPort, or DisabledPort.
DisabledPort	Any PTP Port of the PTP Instance for which the variables portOper, ptPPortEnabled, and asCapable are not all TRUE.
NOTE—PTP Port states are per PTP Port and per domain (i.e., per PTP Instance; see 8.1).	

10 NOTE—Information contained in Sync and associated Follow_Up messages received on PTP Ports whose PTP Port
11 state is PassivePort is discarded; the SiteSyncSync state machine (see 10.2.7) uses only information received from a PTP
12 Port whose PTP Port state is TimeReceiverPort.

13 An example timeTransmitter/timeReceiver hierarchy of PTP Instances is shown in Figure 10-10. The
14 Grandmaster PTP Instance ports all have PTP Port state of TimeTransmitterPort. All the other PTP Instances
15 have exactly one timeReceiver port. The time-synchronization spanning tree is composed of the PTP
16 Instances and the links that do not have an endpoint PTP Port whose PTP Port state is PassivePort.

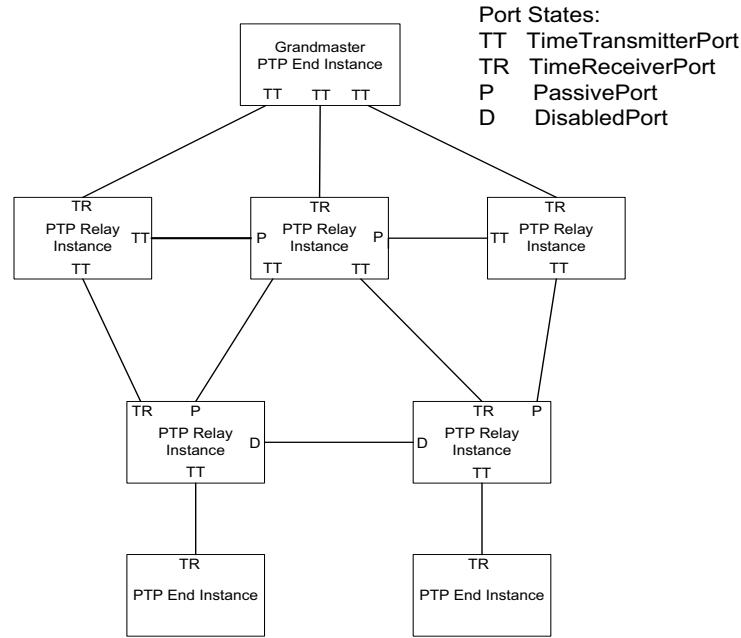


Figure 10-10—Example timeTransmitter/timeReceiver hierarchy of PTP Instances

2 10.3.1.2 Best timeTransmitter clock algorithm overview

3 In the BTCA (i.e., method a) of 10.3.1.1), best timeTransmitter selection information is exchanged between
 4 PTP Instances of time-aware systems via Announce messages (see 10.5 and 10.6). Each Announce message
 5 contains time-synchronization spanning tree vector information that identifies one PTP Instance as the root
 6 of the time-synchronization spanning tree and, if the PTP Instance is grandmaster-capable, the Grandmaster
 7 PTP Instance. Each PTP Instance in turn uses the information contained in the Announce messages it
 8 receives, along with its knowledge of itself, to compute which of the PTP Instances that it has knowledge of
 9 ought to be the root of the spanning tree and, if grandmaster-capable, the Grandmaster PTP Instance. As part
 10 of constructing the time-synchronization spanning tree, each PTP Port of each PTP Instance is assigned a
 11 PTP Port state from Table 10-2 by state machines associated with the ports and with the PTP Instance as a
 12 whole.

13 NOTE—The BTCA described in this standard is the default BTCA according to the specifications of 9.3
 14 of IEEE Std 1588-2019. It is also equivalent to a subset of the Rapid Spanning Tree Protocol (RSTP) described in
 15 IEEE Std 802.1Q-2018 (though the full RSTP described in IEEE Std 802.1Q-2018 is not equivalent to the full BTCA
 16 described in IEEE Std 1588-2019). The BTCA description here uses the formalism of the RSTP description
 17 in IEEE Std 802.1Q-2018.

18 10.3.1.3 External port configuration overview

19 In external port configuration (i.e., method b) of 10.3.1.1), an external entity determines the synchronization
 20 spanning tree and sets the PTP Port states accordingly. The method used by the external entity to determine
 21 the synchronization spanning tree is outside the scope of this standard. However, as with the BTCA,
 22 Announce messages are used to transport information on the time-synchronization spanning tree and
 23 Grandmaster PTP Instance time properties information from one PTP Instance to the next in the tree. The
 24 external entity sets the state of a PTP Port by setting the value of
 25 externalPortConfigurationPortDS.desiredState to the desired state.

1 In the case of external port configuration, the time-synchronization spanning tree and the desired PTP Port
 2 states are controlled by an external entity, but the Grandmaster PTP Instance might change. If the
 3 timeReceiver port of a PTP Instance that is gmCapable (priority1 < 255, see 8.6.2.1) is no longer asCapable,
 4 the state of this PTP Port changes from TimeReceiverPort to DisabledPort (see 10.3.6.2) and the PTP
 5 Instance becomes grandmaster for the time synchronization (sub-)tree where it is the root (10.3.15.2.2 f) 3)).
 6 The original time-synchronization spanning tree can be split into disjunct subtrees with different, mutually
 7 unsynchronized, Grandmaster PTP Instances. If the port becomes asCapable again, its PTP Port state is
 8 again set to the desired state. If the timeReceiver port of a PTP Instance that is not gmCapable is no longer
 9 asCapable, the state of this PTP Port changes from TimeReceiverPort to DisabledPort (see 10.3.6.2).
 10 However, in this case gmPresent is set to FALSE (see 10.3.15.2.2 f) 3)), and the PTP Instance does not send
 11 Sync messages on any of its ports.

12 In external port configuration, there is no supervision of announce receipt timeout (see 10.7.3.2).

13 10.3.2 systemIdentity

14 The systemIdentity attribute of a PTP Instance is a UInteger112 (i.e., a 14-byte, unsigned integer) formed by
 15 concatenating the following attributes, in the following order, from most significant to least significant octet:

- 16 a) priority1 (1 octet; see 8.6.2.1)
- 17 b) clockClass (1 octet; see 8.6.2.2 and 6.4.3.8)
- 18 c) clockAccuracy (1 octet; see 8.6.2.3 and 6.4.3.8)
- 19 d) offsetScaledLogVariance (2 octets; see 8.6.2.4 and 6.4.3.8)
- 20 e) priority2 (1 octet; see 8.6.2.5)
- 21 f) clockIdentity (8 octets; see 8.5.2.2 and 6.4.3.6)

22 The systemIdentity attribute is defined for convenience when comparing two PTP Instances to determine,
 23 when using the BTCA (i.e., method a) of 10.3.1.1), which is a better candidate for root and if the PTP
 24 Instance is grandmaster-capable (i.e., the value of priority1 is less than 255; see 8.6.2.1). Two PTP Instances
 25 are compared as follows. Let the systemIdentity of PTP Instance A be S_A and the systemIdentity of PTP
 26 Instance B be S_B . Let the clockIdentity of A be C_A and the clockIdentity of B be C_B . Then, if $C_A \neq C_B$, i.e.,
 27 A and B represent different PTP Instances,

- 28 g) A is better than B if and only if $S_A < S_B$, and
- 29 h) B is better than A if and only if $S_B < S_A$.

30 If $C_A = C_B$, i.e., A and B represent the same PTP Instance,

- 31 i) $S_A < S_B$ means that A represents an upgrading of the PTP Instance compared to B or, equivalently, B
 represents a downgrading of the PTP Instance compared to A,
- 33 j) $S_B < S_A$ means that B represents an upgrading of the PTP Instance compared to A or, equivalently,
 A represents a downgrading of the PTP Instance compared to B, and
- 35 k) $S_A = S_B$ means that A and B represent the same PTP Instance that has not changed.

36 Comparisons g) and h) in this subclause imply that, with the ordering of attributes in the systemIdentity, the
 37 clockIdentity is a tie-breaker when two different PTP Instances that have identical attributes a) through e)
 38 are compared.

39 Comparisons g) and h) also imply that a PTP Instance that is grandmaster-capable is always better than
 40 another PTP Instance that is not grandmaster-capable because the priority1 is less than 255 if the PTP
 41 Instance is grandmaster-capable and is equal to 255 if it is not grandmaster-capable (see 8.6.2.1).

1 The cases where A and B represent different PTP Instances and represent the same PTP Instance are handled
 2 separately in the BTCA. When comparing two different PTP Instances, the better PTP Instance is selected as
 3 the Grandmaster PTP Instance candidate. However, if A and B represent the same PTP Instance with
 4 attributes that have changed, the PTP Instance is considered as having the most recent attributes when doing
 5 subsequent comparisons with other PTP Instances.

6 **10.3.3 stepsRemoved**

7 Every PTP Instance has a stepsRemoved associated with it. For the root PTP Instance, and therefore the
 8 Grandmaster PTP Instance when the root is grandmaster-capable, it is zero. For all other PTP Instances, it is
 9 the number of gPTP communication paths in the path from the root to the respective PTP Instance.

10 NOTE—For example, stepsRemoved for a timeReceiver port on the same gPTP communication path as the
 11 Grandmaster PTP Instance will have a value of 1, indicating that a single path was traversed.

12 The stepsRemoved attributes of different ports of a PTP Instance are compared after comparisons of other
 13 attributes that take precedence (i.e., priority1, clockClass, clockAccuracy, offsetScaledLogVariance,
 14 priority2) do not result in one PTP Port being declared better than the other. Among the ports whose
 15 stepsRemoved attributes are compared, the PTP Port on the PTP Instance with the lowest stepsRemoved is
 16 assigned the state of TimeReceiverPort for that PTP Instance (the root PTP Instance does not have a
 17 TimeReceiverPort). This lowest stepsRemoved is also considered the stepsRemoved for the PTP Instance. If
 18 a PTP Instance has two or more ports with the same stepsRemoved, then the PTP Port with the smallest
 19 portNumber is selected as the TimeReceiverPort.

20 **10.3.4 time-synchronization spanning tree priority vectors**

21 PTP Instances send best timeTransmitter selection information to each other in Announce messages. The
 22 information is structured in a time-synchronization spanning tree priority vector. Time-synchronization
 23 spanning tree priority vectors provide the basis for a concise specification of the BTCA's determination of
 24 the time-synchronization spanning tree and Grandmaster PTP Instance. A priority vector is formed by
 25 concatenating the following attributes, in the following order, from most significant to least significant octet:

- 26 a) rootSystemIdentity (14 octets; see 10.3.2)
- 27 b) stepsRemoved (2 octets; see 10.3.3)
- 28 c) sourcePortIdentity (i.e., portIdentity of the transmitting PTP Instance; 10 octets; see 8.5.2 and
 10.6.2)
- 29 d) portNumber of the receiving PTP Port (2 octets; see 8.5.2.3)

31 The first two components of a priority vector are significant throughout the gPTP domain; they are
 32 propagated via Announce messages and updated through invocation of BTCA state machines. The next
 33 component is assigned hop-by-hop for each gPTP communication path or PTP Instance and thus is of local
 34 significance only. It is used as a tie-breaker in decisions between time-synchronization spanning tree priority
 35 vectors that are otherwise equal. The fourth component is not conveyed in Announce messages, but is used
 36 as a tie-breaker within a PTP Instance.

37 The set of all time-synchronization spanning tree priority vectors is totally ordered. For all components, a
 38 lesser numerical value is better, and earlier components in the preceding list are more significant. In
 39 addition, as mentioned earlier, a priority vector that reflects a root PTP Instance that is grandmaster-capable
 40 is always better than a priority vector that reflects a root PTP Instance that is not grandmaster-capable. As
 41 each PTP Port receives a priority vector, via an Announce message, from ports closer to the root, additions
 42 are made to one or more components to yield a worse priority vector. This process of receiving information,
 43 adding to it, and passing it on, can be described in terms of the message priority vector received and a set of
 44 priority vectors used to facilitate the computation of a priority vector for each PTP Port, to be transmitted in
 45 further Announce Messages to PTP Instances further from the root.

10.3.5 Priority vector calculations

2 The portPriorityVector is the time-synchronization spanning tree priority vector held for the PTP Port when
 3 the reception of Announce messages and any pending update of information has been completed:

4 portPriorityVector = {rootSystemIdentity : stepsRemoved : sourcePortIdentity : portNumber}

5 A messagePriorityVector is the time-synchronization spanning tree priority vector conveyed in a received
 6 Announce Message. For a PTP Instance S receiving an Announce Message on PTP Port P_S with
 7 portNumber PN_S , from a TimeTransmitterPort with portIdentity P_M on PTP Instance M claiming a
 8 rootSystemIdentity of R_M and a stepsRemoved of SR_M :

9 messagePriorityVector = { R_M : SR_M : P_M : PN_S }

10 This messagePriorityVector is superior to the portPriorityVector and will replace it if, and only if, the
 11 messagePriorityVector is better than the portPriorityVector, or the Announce message has been transmitted
 12 from the same timeTransmitter PTP Instance and TimeTransmitterPort as the portPriorityVector, i.e., if the
 13 following is true:

14 $((R_M < \text{rootSystemIdentity})) \parallel$

15 $((R_M == \text{rootSystemIdentity}) \&\& (SR_M < \text{stepsRemoved})) \parallel$

16 $((R_M == \text{rootSystemIdentity}) \&\& (SR_M == \text{stepsRemoved}) \&\& (P_M < \text{sourcePortIdentity} \text{ (of current})$
 17 $\text{timeTransmitter PTP Instance)})) \parallel$

18 $((R_M == \text{rootSystemIdentity}) \&\& (SR_M == \text{stepsRemoved})$

19 $\&\& (P_M == \text{sourcePortIdentity} \text{ (of current timeTransmitter PTP Instance)}) \&\& (PN_S < \text{portNumber})) \parallel$

20 $((P_M.\text{clockIdentity} == \text{sourcePortIdentity.clockIdentity} \text{ (of current timeTransmitter PTP Instance)}) \&\&$
 21 $(P_M.\text{portNumber} == \text{sourcePortIdentity.PortNumber} \text{ (of the current timeTransmitter PTP Instance)}))$

22 A gmPathPriorityVector can be calculated from a received portPriorityVector by adding one to the
 23 stepsRemoved component:

24 gmPathPriorityVector = { R_M : $SR_M + 1$: P_M : PN_S }

25 The systemPriorityVector for a PTP Instance S with systemIdentity S_S and clockIdentity C_S is the priority
 26 vector that would, with the portIdentity of the TimeReceiverPort set equal to the portIdentity of the
 27 transmitting PTP Port, be used as the message priority vector in Announce Messages transmitted on S's
 28 ports whose state is TimeTransmitterPort if S was selected as the root:

29 systemPriorityVector = { S_S : 0 : { C_S : 0} : 0}

1 The gmPriorityVector for S is the best of the set comprising the systemPriorityVector vector plus every
 2 gmPathPriorityVector for which the clockIdentity of the timeTransmitter PTP Instance portIdentity is not the
 3 clockIdentity of S. If the systemPriorityVector is best, S has been selected as the root. When the best
 4 gmPathPriorityVector is that of PTP Port PN_S above, then:

5 gmPriorityVector = {S_S : 0 : {C_S : 0} : 0} if S is better than R_M, or

6 gmPriorityVector = {R_M : SR_M + 1 : P_M : PN_S} if S is worse than R_M.

7 The timeTransmitterPriorityVector for a PTP Port Q on PTP Instance S is the gmPriorityVector with S's
 8 clockIdentity C_S substituted for the clockIdentity of the timeTransmitter portIdentity, and Q's portNumber
 9 PN_Q substituted for the portNumber of the timeTransmitter portIdentity and for the portNumber of the
 10 receiving PTP Port:

11 timeTransmitterPriorityVector = {S_S : 0 : {C_S : PN_Q} : PN_Q} if S is better than R_M, or

12 timeTransmitterPriorityVector = {R_M : SR_M + 1 : {C_S : PN_Q} : PN_Q} if S is worse than R_M.

13 If the timeTransmitterPriorityVector is better than the portPriorityVector, the PTP Port will be the
 14 TimeTransmitterPort for the attached gPTP communication path and the portPriorityVector will be updated.
 15 The messagePriorityVector information in Announce messages transmitted by a PTP Port always includes
 16 the first three components of the timeTransmitterPriorityVector of the PTP Port.

17 NOTE—The consistent use of lower numerical values to indicate better information is deliberate as the
 18 TimeTransmitterPort that is closest to the root, i.e., has a numerically lowest path cost component, is selected from
 19 amongst potential alternatives for any given gPTP communication path. Adopting the conventions that lower numerical
 20 values indicate better information, that where possible more significant priority components are encoded earlier in the
 21 octet sequence of an Announce message, and that earlier octets in the encoding of individual components are more
 22 significant allows concatenated octets that compose a priority vector to be compared as if they were a multiple octet
 23 encoding of a single number, without regard to the boundaries between the encoded components. To reduce the
 24 confusion that naturally arises from having the lesser of two numerical values represent the better of the two, i.e., the one
 25 to be chosen all other factors being equal, this clause uses the following consistent terminology. Relative numeric values
 26 are described as “least,” “lesser,” “equal,” and “greater,” and their comparisons as “less than,” “equal to,” or “greater
 27 than,” while relative time-synchronization spanning tree priorities are described as “best,” “better,” “the same,”
 28 “different,” and “worse” and their comparisons as “better than,” “the same as,” “different from,” and “worse than.” The
 29 operators “<” and “==” represent less than and equal to, respectively. The terms “superior” and “inferior” are used for
 30 comparisons that are not simply based on priority, but can include the fact that the priority vector of a
 31 TimeTransmitterPort can replace an earlier vector transmitted in an Announce message by the same PTP Port.

32 **10.3.6 PTP Port state assignments**

33 **10.3.6.1 PTP Port state assignments when the BTCA is used**

34 The BTCA assigns one of the following PTP Port states to each PTP Port: TimeTransmitterPort,
 35 TimeReceiverPort, PassivePort, or DisabledPort.

36 The DisabledPort state is assigned if portOper is FALSE (see 10.2.5.12), ptpPortEnabled is FALSE
 37 (see 10.2.5.13), or asCapable is FALSE (see 10.2.5.1).

38 A PTP Port for which portOper, ptpPortEnabled, and asCapable are all TRUE has its PTP Port state assigned
 39 according to the source and relative priority of the time-synchronization spanning tree portPriorityVector
 40 (see 10.3.4 and 10.3.5) as follows:

41 a) If the PTP Instance is not the root, the source of the gmPriorityVector is the TimeReceiverPort.

- 1 b) Each PTP Port whose portPriorityVector is its timeTransmitterPriorityVector is a
2 TimeTransmitterPort.
- 3 c) Each PTP Port, other than the TimeReceiverPort, whose portPriorityVector has been received from
4 another PTP Instance or another PTP Port on this PTP Instance is a PassivePort.

5 10.3.6.2 PTP Port state assignments when external port configuration is used

6 If external port configuration is used, one of the states TimeTransmitterPort, TimeReceiverPort, PassivePort,
7 or DisabledPort is assigned to each PTP Port by an external entity, as described in this subclause.

8 The DisabledPort state is assigned if portOper is FALSE (see 10.2.5.12), ptpPortEnabled is FALSE (see
9 10.2.5.13), or asCapable is FALSE (see 10.2.5.1).

10 The member externalPortConfigurationPortDS.desiredState (see 14.12.2) is used by an external entity to set
11 the state of the respective PTP Port to TimeTransmitterPort, TimeReceiverPort, or PassivePort. When this
12 member is set, its value is copied to the per PTP Port local variable portStateInd (see 10.3.15.1.5). If
13 portOper, ptpPortEnabled, and asCapable are all TRUE for this PTP Port, the PTP Port state is set equal to
14 the value of externalPortConfigurationPortDS.desiredState by copying the value of this member to the
15 element of the selectedState array (see 10.2.4.20) for this PTP Port.

16 10.3.7 Overview of best timeTransmitter clock selection, external port configuration, and 17 announce interval setting state machines

18 10.3.7.1 Best timeTransmitter clock selection state machines overview

19 The best timeTransmitter clock selection function in a PTP Instance is specified by a number of cooperating
20 state machines. Figure 10-11 is not itself a state machine, but illustrates the machines, their local variables,
21 their interrelationships, their performance parameters, and the global variables and structures used to
22 communicate between them.

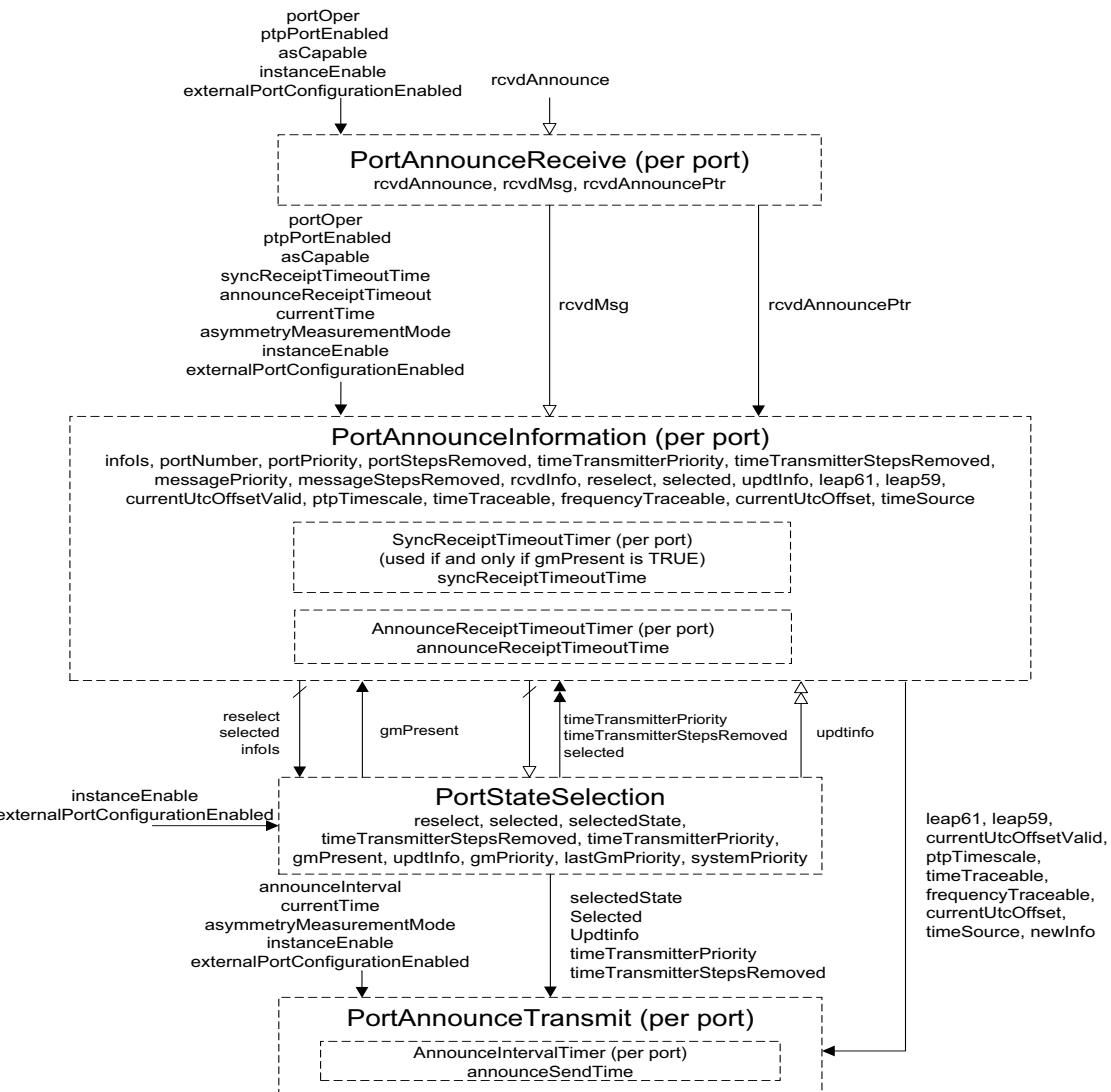
23 NOTE—The BTCA state machines are all invoked by the media-independent layer, i.e., by the SiteSync and PortSync
24 entities. The media-dependent layer, i.e., the MD entity, simply takes an Announce message received from the PortSync
25 entity of the same PTP Port and gives it to the next lower layer (e.g., IEEE 802.3, IEEE 802.11). It is the PortSync entity
26 that generates and consumes Announce messages.

27 The following media-independent layer state machines are in Figure 10-11:

- 28 a) PortAnnounceReceive (one instance per PTP Instance, per PTP Port): receives Announce
29 information from the MD entity of the same PTP Port, determines if the Announce message is
30 qualified and, if so, sets the rcvdMsg variable. This state machine is invoked by the PortSync entity
31 of the PTP Port.
- 32 b) PortAnnounceInformation (one instance per PTP Instance, per PTP Port): receives new qualified
33 Announce information from the PortAnnounceReceive state machine, determines if the Announce
34 information is better than the current best master information it knows about, and updates the current
35 best timeTransmitter information when it receives updated PTP Port state information from the
36 PortStateSelection state machine and when announce receipt timeout or, when gmPresent is TRUE,
37 sync receipt timeout occurs. This state machine is invoked by the PortSync entity of the PTP Port.
- 38 c) PortStateSelection (one instance per PTP Instance): updates the gmPathPriority vector for each PTP
39 Port of the PTP Instance, the gmPriorityVector for the PTP Instance, and the
40 timeTransmitterPriorityVector for each PTP Port of the PTP Instance; determines the PTP Port state
41 for each PTP Port; and updates gmPresent. This state machine is invoked by the SiteSync entity of
42 the PTP Instance.

- 1 d) PortAnnounceTransmit (one instance per PTP Instance, per PTP Port): if the PTP Port state is
2 TimeTransmitterPort, transmits Announce information to the MD entity when an announce interval
3 has elapsed, PTP Port states have been updated, and portPriority and portStepsRemoved information
4 has been updated with newly determined timeTransmitterPriority and
5 timeTransmitterStepsRemoved information. This state machine is invoked by the PortSync entity of
6 the PTP Port and is also used when external port configuration is used.

1

**Notation:**

Variables are shown both within the machine where they are principally used and between machines where they are used to communicate information. In the latter case a variety of arrow styles, running from one machine to another, show how each is typically used:

- Not changed by the target machine. Where the machines are both per port, this variable communicates between instances for the same port
- Set (or cleared) by the originating machine, cleared (or set) by the target machine. Where the machines are both per port, this communicates between instances for the same port.
- As above, except that the originating per-port machine instance communicates with multiple port machine instances (by setting or clearing variables owned by those ports).
- As above, except that multiple per-port instances communicate with (an)other instance(s) (by setting or clearing variables owned by the originating ports).

Figure 10-11—Best timeTransmitter clock selection state machines—overview and interrelationships

10.3.7.2 External port configuration state machines overview

2 The external port configuration function in a PTP Instance is specified by a number of cooperating state
3 machines. Figure 10-12 is not itself a state machine, but illustrates the machines, their local variables, their
4 interrelationships, their performance parameters, and the global variables and structures used to
5 communicate between them.

6 NOTE—The external port configuration state machines are all invoked by the media-independent layer and are per PTP
7 Port, i.e., they are invoked by the PortSync entity for the respective PTP Port. The media-dependent layer, i.e., the MD
8 entity, simply takes an Announce message received from the PortSync entity of the same PTP Port and gives it to the
9 next lower layer (e.g., IEEE 802.3, IEEE 802.11). It is the PortSync entity that generates and consumes Announce
10 messages.

11 The following media-independent layer state machines are in Figure 10-12:

- 12 a) PortAnnounceInformationExt (one instance per PTP Instance, per PTP Port): Receives and stores
13 new Announce information received in Announce messages.
- 14 b) PortStateSettingExt (one instance per PTP Instance): Copies the desired PTP Port state for the PTP
15 Port to the respective selectedState array element, updates gmPresent, computes
16 timeTransmitterStepsRemoved, stores the time properties information in the respective global
17 variables, and computes the gmPriorityVector and timeTransmitterPriorityVector.
- 18 c) PortAnnounceTransmit (one instance per PTP Instance, per PTP Port): If the PTP Port state is
19 TimeTransmitterPort, transmits Announce information to the MD entity when an announce interval
20 has elapsed, PTP Port states have been updated, and portPriority and portStepsRemoved information
21 has been updated with newly determined timeTransmitterPriority and
22 timeTransmitterStepsRemoved information. This state machine is invoked by the PortSync entity of
23 the PTP Port and is also used when the BTCA is used.

1

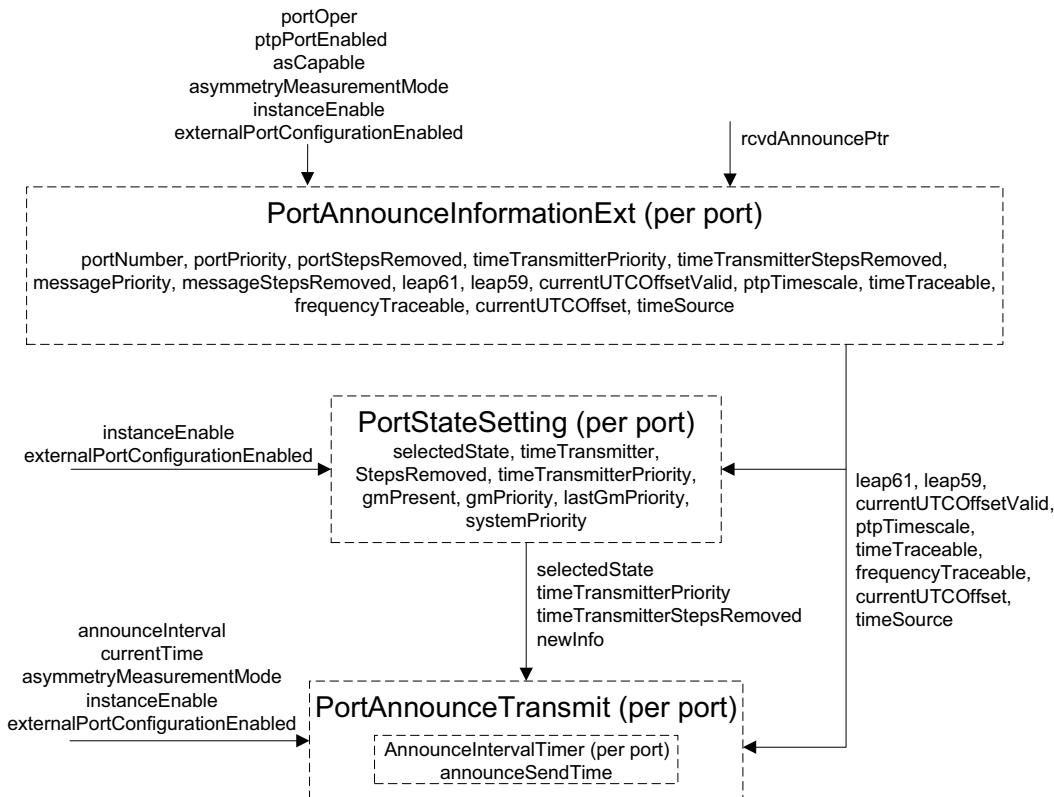


Figure 10-12—External port configuration state machines—overview and interrelationships

2 10.3.7.3 AnnounceIntervalSetting state machine overview

3 An additional state machine, the AnnounceIntervalSetting state machine, receives a Signaling message that
 4 contains a Message Interval Request TLV (see 10.6.4.3) and sets the global variables that give the duration
 5 of the mean interval between successive Announce messages.

6 10.3.8 Overview of global variables used by best timeTransmitter clock selection, external 7 port configuration, and announce interval setting state machines

8 10.3.9 and 10.3.10 define global variables used by best timeTransmitter clock selection, external port
 9 configuration, and announce interval setting state machines whose scopes are as follows:

- 10 — Per PTP Instance (i.e., per domain)
- 11 — Per PTP Instance, per PTP Port
- 12 — Instance used by CMLDS (see 11.2.17) (i.e., variable is common across all [Link Ports](#))
- 13 — Instance used by CMLDS, per [Link Port](#)

14 Table 10-3 summarizes the scope of each global variable of 10.3.9 and 10.3.10.

Table 10-3—Summary of scope of global variables used by best timeTransmitter clock selection, external port configuration, and announce interval setting state machines (see 10.3.9 and 10.3.10)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per Link Port
reselect	10.3.9.1	Yes	No	No	No
selected	10.3.9.2	Yes	No	No	No
timeTransmitterStepsRemoved	10.3.9.3	Yes	No	No	No
leap61	10.3.9.4	Yes	No	No	No
leap59	10.3.9.5	Yes	No	No	No
currentUtcOffsetValid	10.3.9.6	Yes	No	No	No
ptpTimescale	10.3.9.7	Yes	No	No	No
timeTraceable	10.3.9.8	Yes	No	No	No
frequencyTraceable	10.3.9.9	Yes	No	No	No
currentUtcOffset	10.3.9.10	Yes	No	No	No
timeSource	10.3.9.11	Yes	No	No	No
sysLeap61	10.3.9.12	Yes	No	No	No
sysLeap59	10.3.9.13	Yes	No	No	No
sysCurrentUtcOffsetValid	10.3.9.14	Yes	No	No	No
sysPtpTimescale	10.3.9.15	Yes	No	No	No
sysTimeTraceable	10.3.9.16	Yes	No	No	No
sysFrequencyTraceable	10.3.9.17	Yes	No	No	No
sysCurrentUtcOffset	10.3.9.18	Yes	No	No	No
sysTimeSource	10.3.9.19	Yes	No	No	No
systemPriority	10.3.9.20	Yes	No	No	No
gmPriority	10.3.9.21	Yes	No	No	No
lastGmPriority	10.3.9.22	Yes	No	No	No
pathTrace	10.3.9.23	Yes	No	No	No
externalPortConfigurationEnabled	10.3.9.24	Yes	No	No	No
lastAnnouncePort	10.3.9.25	Yes	No	No	No
announceReceiptTimeoutTimeInterval	10.3.10.1	No	Yes	No	No
announceSlowdown	10.3.10.2	No	Yes	No	No
oldAnnounceInterval	10.3.10.3	No	Yes	No	No

Table 10-3—Summary of scope of global variables used by best timeTransmitter clock selection, external port configuration, and announce interval setting state machines (see 10.3.9 and 10.3.10) (continued)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per Link Port
infoIs	10.3.10.4	No	Yes	No	No
timeTransmitterPriority	10.3.10.5	No	Yes	No	No
currentLogAnnounceInterval	10.3.10.6	No	Yes	No	No
initialLogAnnounceInterval	10.3.10.7	No	Yes	No	No
announceInterval	10.3.10.8	No	Yes	No	No
messageStepsRemoved	10.3.10.9	No	Yes	No	No
newInfo	10.3.10.10	No	Yes	No	No
portPriority	10.3.10.11	No	Yes	No	No
portStepsRemoved	10.3.10.12	No	Yes	No	No
rcvdAnnouncePtr	10.3.10.13	No	Yes	No	No
rcvdMsg	10.3.10.14	No	Yes	No	No
updInfo	10.3.10.15	No	Yes	No	No
annLeap61	10.3.10.16	No	Yes	No	No
annLeap59	10.3.10.17	No	Yes	No	No
annCurrentUtcOffsetValid	10.3.10.18	No	Yes	No	No
annPtpTimescale	10.3.10.19	No	Yes	No	No
annTimeTraceable	10.3.10.20	No	Yes	No	No
annFrequencyTraceable	10.3.10.21	No	Yes	No	No
annCurrentUtcOffset	10.3.10.22	No	Yes	No	No
annTimeSource	10.3.10.23	No	Yes	No	No
receivedPathTrace	10.3.10.24	No	Yes	No	No

1 10.3.9 Per PTP Instance global variables

2 10.3.9.1 reselect: A Boolean array of length numberPorts+1 (see 8.6.2.8). Setting reselect[j], where $0 \leq j \leq$ 3 numberPorts, to TRUE causes the STATE_SELECTION block of the PortStateSelection state machine (see 4 10.3.13) to be re-entered, which in turn causes the PTP Port state of each PTP Port of the PTP Instance to be 5 updated (via the function updStatesTree(); see 10.3.13.2.4). This variable is used only by the BTCA, i.e., 6 not by the **external port** configuration option.

7 10.3.9.2 selected: A Boolean array of length numberPorts+1 (see 8.6.2.8). selected[j], where $0 \leq j \leq$ 8 numberPorts, is set to TRUE immediately after the PTP Port states of all the ports are updated. This value 9 indicates to the PortAnnounceInformation state machine (see 10.3.12) that it can update the

1 portPriorityVector and other variables for each PTP Port. This variable is used by both the BTCA and the
2 **external port** configuration option; however, its value does not impact the **external port** configuration option
3 (see the NOTE in 10.3.16.3).

4 NOTE—Array elements 0 of the reselect and selected arrays are not used, except that the function clearReselectTree()
5 sets reselect[0] to FALSE when it sets the entire array to zero and the function setSelectedTree() sets selected[0] to
6 TRUE when it sets the entire array to TRUE. This action is taken only for convenience, so that array element j can
7 correspond to PTP Port j. Note also that, in contrast, selectedState[0] is used (see 10.2.4.20).

8 **10.3.9.3 timeTransmitterStepsRemoved:** The value of stepsRemoved for the PTP Instance, after the PTP
9 Port states of all the ports have been updated (see 10.3.13.2.4 for details on the computation of
10 timeTransmitterStepsRemoved). The data type for timeTransmitterStepsRemoved is UInteger16. This
11 variable is used by both the BTCA and the **external port** configuration option.

12 **10.3.9.4 leap61:** A Boolean variable whose value is TRUE if the last minute of the current UTC day, relative
13 to the current Grandmaster Clock, contains 61 s and FALSE if the last minute of the current UTC day does
14 not contain 61 s. This variable is used by both the BTCA and the **external port** configuration option.

15 **10.3.9.5 leap59:** A Boolean variable whose value is TRUE if the last minute of the current UTC day, relative
16 to the current Grandmaster Clock, contains 59 s and FALSE if the last minute of the current UTC day does
17 not contain 59 s. This variable is used by both the BTCA and the **external port** configuration option.

18 **10.3.9.6 currentUtcOffsetValid:** A Boolean variable whose value is TRUE if currentUtcOffset (see
19 10.3.9.10), relative to the current Grandmaster Clock, is known to be correct and FALSE if currentUtcOffset
20 is not known to be correct. This variable is used by both the BTCA and the **external port** configuration
21 option.

22 **10.3.9.7 ptpTimescale:** A Boolean variable whose value is TRUE if the timescale of the current
23 Grandmaster Clock is PTP (see 8.2.1) and FALSE if the timescale is ARB. This variable is used by both the
24 BTCA and the **external port** configuration option.

25 **10.3.9.8 timeTraceable:** A Boolean variable whose value is TRUE if both ClockTimeReceiverTime [i.e.,
26 the synchronized time maintained at the timeReceiver (see 10.2.4.3)] and currentUtcOffset (see 10.3.9.10),
27 relative to the current Grandmaster Clock, are traceable to a primary reference and FALSE if one or both are
28 not traceable to a primary reference. This variable is used by both the BTCA and the **external port**
29 configuration option.

30 **10.3.9.9 frequencyTraceable:** A Boolean variable whose value is TRUE if the frequency that determines
31 ClockTimeReceiverTime, i.e., the frequency of the LocalClockEntity multiplied by the most recently
32 computed rateRatio by the PortSyncSyncReceive state machine (see 10.2.8.1.4), is traceable to a primary
33 reference and FALSE if this frequency is not traceable to a primary reference. This variable is used by both
34 the BTCA and the **external port** configuration option.

35 **10.3.9.10 currentUtcOffset:** The difference between TAI time and UTC time, i.e., TAI time minus UTC
36 time, in seconds, and relative to the current Grandmaster Clock, when known. Otherwise, the value has no
37 meaning (see 10.3.9.6). The data type for currentUtcOffset is Integer16. This variable is used by both the
38 BTCA and the **external port** configuration option.

1 NOTE—For example, 2006-01-01 00:00:00 UTC and 2006-01-01 00:00:33 TAI represent the same instant of time. At
2 this time, currentUtcOffset was equal to 33 s.¹⁷

3 **10.3.9.11 timeSource:** The value of the timeSource attribute of the current Grandmaster PTP Instance. The
4 data type for timeSource is TimeSource (see 8.6.2.7). This variable is used by both the BTCA and the
5 **external port** configuration option.

6 **10.3.9.12 sysLeap61:** A Boolean variable whose value is TRUE if the last minute of the current UTC day,
7 relative to the ClockTimeTransmitter entity of this PTP Instance, contains 61 s and FALSE if the last minute
8 of the current UTC day does not contain 61 s. This variable is used by both the BTCA and the **external port**
9 configuration option.

10 **10.3.9.13 sysLeap59:** A Boolean variable whose value is TRUE if the last minute of the current UTC day,
11 relative to the ClockTimeTransmitter entity of this PTP Instance, contains 59 s and FALSE if the last minute
12 of the current UTC day does not contain 59 s. This variable is used by both the BTCA and the **external port**
13 configuration option.

14 **10.3.9.14 sysCurrentUtcOffsetValid:** A Boolean variable whose value is TRUE if currentUtcOffset (see
15 10.3.9.10), relative to the ClockTimeTransmitter entity of this PTP Instance, is known to be correct and
16 FALSE if currentUtcOffset is not known to be correct. This variable is used by both the BTCA and the
17 **external port** configuration option.

18 **10.3.9.15 sysPtpTimescale:** A Boolean variable whose value is TRUE if the timescale of the
19 ClockTimeTransmitter entity of this PTP Instance is PTP (see 8.2.1) and FALSE if the timescale of the
20 ClockTimeTransmitter entity of this PTP Instance is ARB. This variable is used by both the BTCA and the
21 **external port** configuration option.

22 **10.3.9.16 sysTimeTraceable:** A Boolean variable whose value is TRUE if both timeTransmitterTime [i.e.,
23 the time maintained by the ClockTimeTransmitter entity of this PTP Instance (see 10.2.4.21)] and
24 currentUtcOffset (see 10.3.9.10), relative to the ClockTimeTransmitter entity of this PTP Instance, are
25 traceable to a primary reference and FALSE if one or both are not traceable to a primary reference. This
26 variable is used by both the BTCA and the **external port** configuration option.

27 **10.3.9.17 sysFrequencyTraceable:** A Boolean variable whose value is TRUE if the frequency that
28 determines timeTransmitterTime of the ClockTimeTransmitter entity of this PTP Instance, i.e., the
29 frequency of the LocalClockEntity multiplied by the most recently computed gmRateRatio by the
30 ClockTimeTransmitterSyncReceive state machine (see 10.2.4.14 and 10.2.11), is traceable to a primary
31 reference and FALSE if this frequency is not traceable to a primary reference. This variable is used by both
32 the BTCA and the **external port** configuration option.

33 **10.3.9.18 sysCurrentUtcOffset:** The difference between TAI time and UTC time, i.e., TAI time minus UTC
34 time, in seconds, and relative to the ClockTimeTransmitter entity of this PTP Instance, when known.
35 Otherwise, the value has no meaning (see 10.3.9.14). The data type for sysCurrentUtcOffset is Integer16.
36 This variable is used by both the BTCA and the **external port** configuration option.

37 NOTE—See the NOTE in 10.3.9.10 for more detail on the sign convention.

38 **10.3.9.19 sysTimeSource:** The value of the timeSource attribute of the ClockTimeTransmitter entity of this
39 PTP Instance (see 8.6.2.7). The data type for sysTimeSource is TimeSource.

40 **10.3.9.20 systemPriority:** The systemPriority vector for this PTP Instance. The data type for systemPriority
41 is UIInteger224 (see 10.3.5).

¹⁷ Note also that a leap second was not added at the end of the last UTC minute of 2005-12-31.

1 **10.3.9.21 gmPriority:** The current gmPriorityVector for the PTP Instance. The data type for gmPriority is
2 UInteger224 (see 10.3.5).

3 **10.3.9.22 lastGmPriority:** The previous gmPriorityVector for the PTP Instance, prior to the most recent
4 invocation of the PortStateSelection state machine. The data type for lastGmPriority is UInteger224 (see
5 10.3.4). lastGmPriority is used only by the BTCA, i.e., not by the **external port** configuration option.

6 **10.3.9.23 pathTrace:** An array that contains the clockIdentities of the successive PTP Instances that receive,
7 process, and send Announce messages. The data type for pathTrace is ClockIdentity[N], where N is the
8 number of PTP Instances, including the Grandmaster PTP Instance, that the Announce information has
9 traversed. This variable is used by both the BTCA and the **external port** configuration option.

10 NOTE 1—N is equal to stepsRemoved+1 (see 10.6.3.2.6). The size of the pathTrace array can change after each
11 reception of an Announce message, up to the maximum size for the respective medium. For example, the maximum
12 value of N for a full-duplex IEEE 802.3 medium is 179. This is obtained from the fact that the number of PTP octets in
13 an Announce message is $68 + 8N$, where N is the number of entries in the pathTrace array (see 10.6.3.1 and Table 10-
14 11), and the maximum payload size for full-duplex IEEE 802.3 media is 1500 octets. Setting $68 + 8N = 1500$, and
15 solving for N gives N = 179.

16 NOTE 2—The current behavior for the path trace feature is documented in 10.3.11.2.1 and 10.3.16.2.1 and is as follows:

- 17 — Item c) of 10.3.11.2.1, the description of the qualifyAnnounce() function of the PortAnnounceReceive state
18 machine, indicates that if a path trace TLV is present and one of the elements of the pathSequence array field is
19 equal to the clockIdentity of the clock where the TLV is being processed, the Announce message is not
20 qualified.
- 21 — Item d) of 10.3.11.2.1 (qualifyAnnounce() function) indicates that if the Announce message is qualified and a
22 path trace TLV is present, the pathSequence array of the TLV is copied to the pathTrace array (described in this
23 subclause) and the clockIdentity of the PTP Instance that processes the Announce message is appended to the
24 array. However, if a path trace TLV is not present, the path trace array is empty.
- 25 — Item f) of 10.3.16.2.1, the description of the txAnnounce() function of the PortAnnounceTransmit state
26 machine, indicates that a path trace TLV is constructed and appended to an Announce message just before the
27 Announce message is transmitted only if the pathTrace array is not empty and appending the TLV does not
28 cause the media-dependent layer frame to exceed any respective maximum size. If appending the TLV does
29 cause a respective maximum frame size to be exceeded or if the pathTrace array is empty, the TLV is not
30 appended.
- 31 — As a result of the behaviors of the qualifyAnnounce() and txAnnounce() functions described in this note, the
32 path trace feature is **no longer** used, i.e., a path trace TLV is not appended to an Announce message and the
33 pathTrace array is empty, **when** appending a clockIdentity to the TLV would cause the frame carrying the
34 Announce message to exceed its maximum size.

35 NOTE 3—Once the value of stepsRemoved of an Announce message reaches 255, the Announce message is not
36 qualified [see item b) of 10.3.11.2.1].

37 **10.3.9.24 externalPortConfigurationEnabled:** A variable whose value indicates whether PTP Port states
38 are externally configured or determined by the BTCA. The data type shall be Boolean. The value TRUE
39 indicates that the PTP Port states are externally configured; the value FALSE indicates that the PTP Port
40 states are determined by the BTCA. This variable is used by both the BTCA and the **external port**
41 configuration option.

42 **10.3.9.25 lastAnnouncePort:** The PTP Port number of the PTP Port on which the most recent Announce
43 message was received. This variable is used by the PortAnnounceInformationExt and PortStateSettingExt
44 state machines for the external port configuration option. This variable is not used by the BTCA. The data
45 type for this variable is UInteger16.

1 10.3.10 Per-port global variables

2 **10.3.10.1 announceReceiptTimeoutTimeInterval:** The time interval after which announce receipt timeout
 3 occurs if an Announce message has not been received during the interval. The value of
 4 announceReceiptTimeoutTimeInterval is equal to announceReceiptTimeout (see 10.7.3.2) multiplied by the
 5 announceInterval (see 10.3.10.8) for the PTP Port at the other end of the link to which this PTP Port is
 6 attached. The value of announceInterval for the PTP Port at the other end of the link is computed from
 7 logMessageInterval of the received Announce message (see 10.6.2.2.14). The data type for
 8 announceReceiptTimeoutTimeInterval is UScaledNs. This variable is used only by the BTCA, i.e., not by
 9 the **external port** configuration option.

10 **10.3.10.2 announceSlowdown:** A Boolean that is set to TRUE if the AnnounceIntervalSetting state
 11 machine (see Figure 10-19 in item 10.3.17.3) receives a TLV that requests a larger Announce message
 12 transmission interval (see 10.7.2.2) and FALSE otherwise. When announceSlowdown is set to TRUE, the
 13 PortAnnounceTransmit state machine (see Figure 10-18) continues to send Announce messages at the old
 14 (i.e., faster) rate until a number of Announce messages equal to announceReceiptTimeout (see 10.7.3.2)
 15 have been sent, but with the logMessageInterval field of the PTP common header set equal to the new
 16 announce interval (i.e., corresponding to the slower rate). After announceReceiptTimeout Announce
 17 messages have been sent, subsequent Announce messages are sent at the new (i.e., slower) rate and with the
 18 logMessageInterval field of the PTP common header set to the new announce interval. This variable is used
 19 by both the BTCA and the **external port** configuration option. When announceSlowdown is set to FALSE,
 20 the PortAnnounceTransmit state machine immediately sends Announce messages at the new (i.e., faster or
 21 the same) rate.

22 NOTE—If a receiver of Announce messages requests a slower rate, the receiver **continues** to use the upstream
 23 announceInterval value, which it obtains from the logMessageInterval field of received Announce messages, until it
 24 receives an Announce message where that value has changed. If, immediately after requesting a slower Announce
 25 message rate, up to announceReceiptTimeout minus one consecutive Announce messages sent to the receiver are lost,
 26 announce receipt timeout could occur if the sender had changed to the slower rate immediately. Delaying the slowing
 27 down of the sending rate of Announce messages for announceReceiptTimeout messages prevents announce receipt
 28 timeout from occurring until at least announceReceiptTimeout Announce messages have been lost. Note that networks
 29 with high packet loss can still experience announce receipt timeout under high-packet-loss conditions; however,
 30 the announce receipt timeout condition occurs only after at least announceReceiptTimeout Announce messages have
 31 been lost.

32 **10.3.10.3 oldAnnounceInterval:** The saved value of the previous announce interval, when a new announce
 33 interval is requested via a Signaling message that contains a message interval request TLV. The data type for
 34 oldAnnounceInterval is UScaledNs. This variable is used by both the BTCA and the **external port**
 35 configuration option.

36 **10.3.10.4 infoIs:** An Enumeration2 that takes the values Received, Mine, Aged, or Disabled to indicate the
 37 origin and state of the PTP Port's time-synchronization spanning tree information:

- 38 a) If infoIs is Received, the PTP Port has received current information (i.e., announce receipt timeout
 39 has not occurred and, if gmPresent is TRUE, sync receipt timeout also has not occurred) from the
 40 timeTransmitter PTP Instance for the attached gPTP communication path.
- 41 b) If infoIs is Mine, information for the PTP Port has been derived from the TimeReceiverPort for the
 42 PTP Instance (with the addition of TimeReceiverPort stepsRemoved). This includes the possibility
 43 that the TimeReceiverPort is the PTP Port whose portNumber is 0, i.e., the PTP Instance is the root
 44 of the gPTP domain.
- 45 c) If infoIs is Aged, announce receipt timeout or, when gmPresent is TRUE, sync receipt timeout has
 46 occurred.
- 47 d) If portOper, ptpNetEnabled, and asCapable are not all TRUE, infoIs is Disabled.

48 The variable infoIs is used only by the BTCA, i.e., not by the **external port** configuration option.

1 **10.3.10.5 timeTransmitterPriority:** The timeTransmitterPriorityVector for the PTP Port. The data type for
 2 timeTransmitterPriority is UIInteger224 (see 10.3.4). This variable is used by both the BTCA and the
 3 **external port** configuration option.

4 **10.3.10.6 currentLogAnnounceInterval:** The current value of the logarithm to base 2 of the mean time
 5 interval, in seconds, between the sending of successive Announce messages (see 10.7.2.2). This value is set
 6 in the AnnounceIntervalSetting state machine (see 10.3.17). The data type for currentLogAnnounceInterval
 7 is Integer8. This variable is used by both the BTCA and the **external port** configuration option.

8 **10.3.10.7 initialLogAnnounceInterval:** The initial value of the logarithm to base 2 of the mean time
 9 interval, in seconds, between the sending of successive Announce messages (see 10.7.2.2). The data type for
 10 initialLogAnnounceInterval is Integer8. This variable is used by both the BTCA and the **external port**
 11 configuration option.

12 **10.3.10.8 announceInterval:** A variable containing the mean Announce message transmission interval for
 13 the PTP Port. This value is set in the AnnounceIntervalSetting state machine (see 10.3.17). The data type for
 14 announceInterval is UScaledNs. This variable is used by both the BTCA and the **external port** configuration
 15 option.

16 **10.3.10.9 messageStepsRemoved:** The value of stepsRemoved contained in the received Announce
 17 information. The data type for messageStepsRemoved is UIInteger16. This variable is used by both the
 18 BTCA and the **external port** configuration option.

19 **10.3.10.10 newInfo:** A Boolean variable that is set to cause a PTP Port to transmit Announce information;
 20 specifically, it is set when an announce interval has elapsed (see Figure 10-18), PTP Port states have been
 21 updated, and portPriority and portStepsRemoved information has been updated with newly determined
 22 timeTransmitterPriority and timeTransmitterStepsRemoved information. This variable is used by both the
 23 BTCA and the **external port** configuration option.

24 **10.3.10.11 portPriority:** The portPriorityVector for the PTP Port. The data type for portPriority is
 25 UIInteger224 (see 10.3.4). This variable is used only by the BTCA, i.e., not by the **external port** configuration
 26 option.

27 **10.3.10.12 portStepsRemoved:** The value of stepsRemoved for the PTP Port. portStepsRemoved is set
 28 equal to timeTransmitterStepsRemoved (see 10.3.9.3) after timeTransmitterStepsRemoved is updated. The
 29 data type for portStepsRemoved is UIInteger16. This variable is used by both the BTCA and the **external port**
 30 configuration option.

31 **10.3.10.13 rcvdAnnouncePtr:** A pointer to a structure that contains the fields of a received Announce
 32 message. This variable is used by both the BTCA and the **external port** configuration option.

33 **10.3.10.14 rcvdMsg:** A Boolean variable that is TRUE if a received Announce message is qualified and
 34 FALSE if it is not qualified. This variable is used only by the BTCA, i.e., not by the **external port**
 35 configuration option.

36 **10.3.10.15 updtInfo:** A Boolean variable that is set to TRUE to indicate that the PortAnnounceInformation
 37 state machine (see 10.3.12) should copy the newly determined timeTransmitterPriority and
 38 timeTransmitterStepsRemoved to portPriority and portStepsRemoved, respectively. This variable is used by
 39 both the BTCA and the **external port** configuration option; however, its value does not impact the **external**
 40 **port** configuration option (see the NOTE in 10.3.16.3).

41 **10.3.10.16 annLeap61:** A global variable in which the leap61 flag (see 10.6.2.2.8) of a received Announce
 42 message is saved. The data type for annLeap61 is Boolean. This variable is used by both the BTCA and the
 43 **external port** configuration option.

1 **10.3.10.17 annLeap59:** A global variable in which the leap59 flag (see 10.6.2.2.8) of a received Announce
2 message is saved. The data type for annLeap59 is Boolean. This variable is used by both the BTCA and the
3 **external port** configuration option.

4 **10.3.10.18 annCurrentUtcOffsetValid:** A global variable in which the currentUtcOffsetValid flag (see
5 10.6.2.2.8) of a received Announce message is saved. The data type for annCurrentUtcOffsetValid is
6 Boolean. This variable is used by both the BTCA and the **external port** configuration option.

7 **10.3.10.19 annPtpTimescale:** A global variable in which the ptptimescale flag (see 10.6.2.2.8) of a
8 received Announce message is saved. The data type for annPtpTimescale is Boolean. This variable is used
9 by both the BTCA and the **external port** configuration option.

10 **10.3.10.20 annTimeTraceable:** A global variable in which the timeTraceable flag (see 10.6.2.2.8) of a
11 received Announce message is saved. The data type for annTimeTraceable is Boolean. This variable is used
12 by both the BTCA and the **external port** configuration option.

13 **10.3.10.21 annFrequencyTraceable:** A global variable in which the frequencyTraceable flag (see
14 10.6.2.2.8) of a received Announce message is saved. The data type for annFrequencyTraceable is Boolean.
15 This variable is used by both the BTCA and the **external port** configuration option.

16 **10.3.10.22 annCurrentUtcOffset:** A global variable in which the currentUtcOffset field (see 10.6.3.2.1) of
17 a received Announce message is saved. The data type for annCurrentUtcOffset is Integer16. This variable is
18 used by both the BTCA and the **external port** configuration option.

19 **10.3.10.23 annTimeSource:** A global variable in which the timeSource field (see 10.6.3.2.1) of a received
20 Announce message is saved. The data type for annTimeSource is TimeSource (see 8.6.2.7). This variable is
21 used by both the BTCA and the **external port** configuration option.

22 **10.3.10.24 receivedPathTrace:** An array in which the pathSequence array field of the path trace TLV of the
23 most recently received Announce message is saved. The data type for receivedPathTrace is
24 clockIdentity[N], where N is the number of entries in the pathSequence array field.

25 **10.3.11 PortAnnounceReceive state machine**

26 **10.3.11.1 State machine variables**

27 The following variable is used in the state diagram in Figure 10-13 (in 10.3.11.3):

28 **10.3.11.1.1 rcvdAnnouncePAR:** A Boolean variable that notifies the current state machine when Announce
29 message information is received from the MD entity of the same PTP Port. This variable is reset by this state
30 machine.

31 **10.3.11.2 State machine functions**

32 **10.3.11.2.1 qualifyAnnounce (rcvdAnnouncePtr):** Qualifies the received Announce message pointed to
33 by rcvdAnnouncePtr as follows:

34 a) If the Announce message was sent by the current PTP Instance, i.e., if
35 sourcePortIdentity.clockIdentity (see 10.6.2.2.11 and 8.5.2) is equal to thisClock (see 10.2.4.22), the
36 Announce message is not qualified, and FALSE is returned;

37 b) If the stepsRemoved field is greater than or equal to 255, the Announce message is not qualified, and
38 FALSE is returned;

- 1 c) If a path trace TLV is present and one of the elements of the pathSequence array field of the path
- 2 trace TLV is equal to thisClock (i.e., the clockIdentity of the current PTP Instance; see 10.2.4.22),
- 3 the Announce message is not qualified, and FALSE is returned;
- 4 d) Otherwise, the Announce message is qualified, and TRUE is returned. If a path trace TLV is present,
- 5 it is saved in the per port global variable receivedPathTrace. If a path trace TLV is not present, the
- 6 per port global variable receivedPathTrace is set to the empty array.

7 10.3.11.3 State diagram

8 The PortAnnounceReceive state machine shall implement the function specified by the state diagram in
 9 Figure 10-13, the local variable specified in 10.3.11.1, the function specified in 10.3.11.2, and the relevant
 10 global variables specified in 10.2.4, 10.2.5, 10.3.9, and 10.3.10. The state machine is not used if
 11 externalPortConfigurationEnabled is TRUE. The machine receives Announce information from the MD
 12 entity of the same PTP Port, determines if the Announce message is qualified, and if so, sets the rcvdMsg
 13 variable.

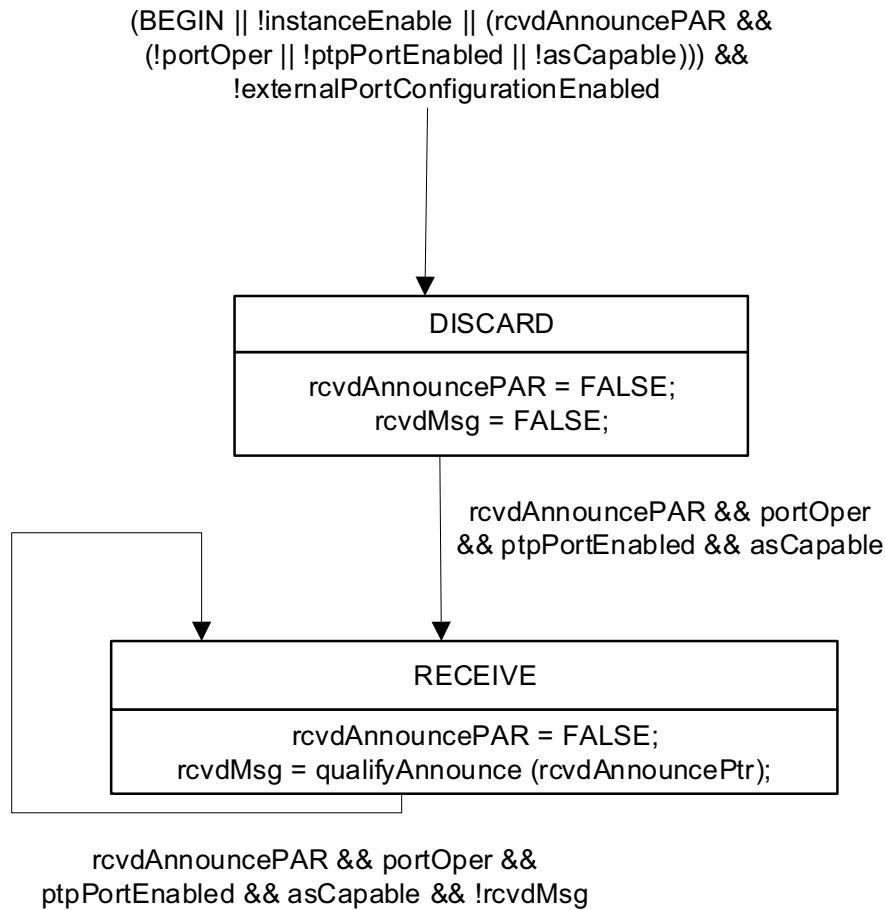


Figure 10-13—PortAnnounceReceive state machine

1 10.3.12 PortAnnounceInformation state machine

2 10.3.12.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-14 (in 10.3.12.3):

4 **10.3.12.1.1 announceReceiptTimeoutTime:** A variable used to save the time at which announce receipt
5 timeout occurs. The data type for announceReceiptTimeoutTime is UScaledNs.

6 **10.3.12.1.2 messagePriorityPAI:** The messagePriorityVector corresponding to the received Announce
7 information. The data type for messagePriorityPAI is UInteger224 (see 10.3.4).

8 **10.3.12.1.3 rcvInfo:** An Enumeration2 that holds the value returned by rcvInfo() (see 10.3.12.2.1).

9 **10.3.12.1.4 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 10-14). The
10 data type for TEMP is Integer16.

11 10.3.12.2 State machine functions

12 **10.3.12.2.1 rcvInfo (rcvdAnnouncePtr):** Decodes the messagePriorityVector (see 10.3.4 and 10.3.5) and
13 stepsRemoved (10.6.3.2.6) field from the Announce information pointed to by rcvdAnnouncePtr
14 (see 10.3.10.13), and then:

- 15 a) Stores the messagePriorityVector and stepsRemoved field value in messagePriorityPAI and
16 messageStepsRemoved, respectively, and then:
 - 17 1) If the received message conveys the PTP Port state TimeTransmitterPort and the
18 messagePriorityVector is the same as the portPriorityVector of the PTP Port, returns
19 RepeatedTimeTransmitterInfo; else
 - 20 2) If the received message conveys the PTP Port state TimeTransmitterPort and the
21 messagePriorityVector is superior to the portPriorityVector of the PTP Port, returns
22 SuperiorTimeTransmitterInfo; else
 - 23 3) If the received message conveys the PTP Port state TimeTransmitterPort, and the
24 messagePriorityVector is worse than the portPriorityVector of the PTP Port, returns
25 InferiorTimeTransmitterInfo; else
 - 26 4) Returns OtherInfo.

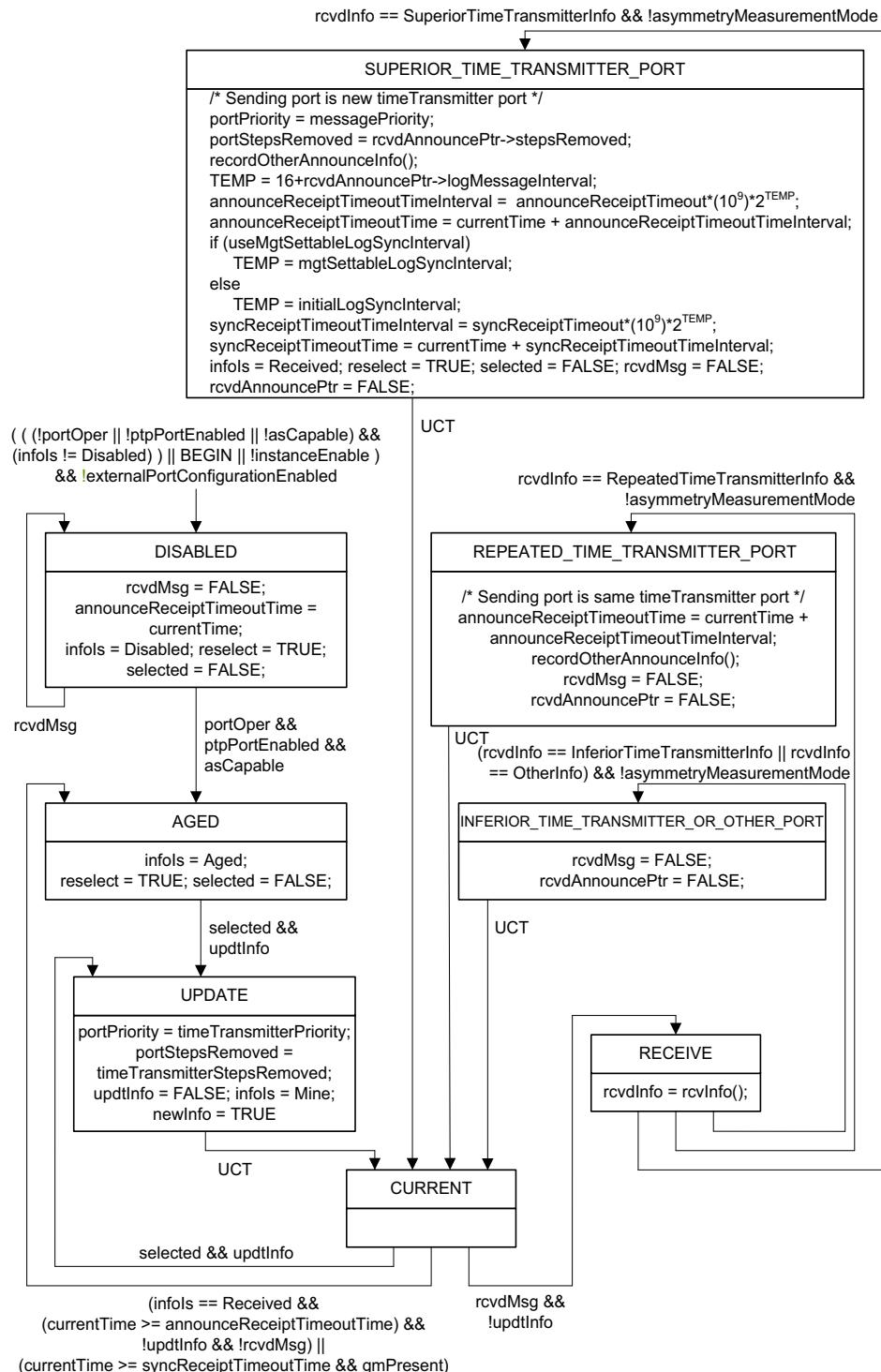
27 NOTE—In accordance with 10.3.5, the messagePriorityVector is superior to the portPriorityVector of the PTP Port if,
28 and only if, the messagePriorityVector is better than the portPriorityVector, or the Announce message has been
29 transmitted from the same timeTransmitter PTP Instance and TimeTransmitterPort as the portPriorityVector. In
30 steps a) 1) to a) 4) in this subclause, rcvInfo() first checks whether the messagePriorityVector and portPriorityVector are
31 the same (and the received message conveys the PTP Port state TimeTransmitterPort), before checking whether the
32 messagePriorityVector is superior to the portPriorityVector. The reason for this sequence is that
33 RepeatedTimeTransmitterInfo needs to be returned if the messagePriorityVector and portPriorityVector are the same,
34 while SuperiorTimeTransmitterInfo needs to be returned in other instances where the Announce message has been
35 transmitted from the same timeTransmitter PTP Instance and TimeTransmitterPort as the portPriorityVector (if the test
36 for SuperiorTimeTransmitterInfo were done before the test for RepeatedTimeTransmitterInfo,
37 SuperiorTimeTransmitterInfo would be returned when RepeatedTimeTransmitterInfo is desired).

38 **10.3.12.2.2 recordOtherAnnounceInfo():** Saves the flags leap61, leap59, currentUtcOffsetValid,
39 ptptimescale, timeTraceable, and frequencyTraceable, and the fields currentUtcOffset and timeSource, of
40 the received Announce message for this PTP Port. The values are saved in the per-PTP Port global variables
41 annLeap61, annLeap59, annCurrentUtcOffsetValid, annPtptimescale, annTimeTraceable,
42 annFrequencyTraceable, annCurrentUtcOffset, and annTimeSource (see 10.3.10.16 through 10.3.10.23).

10.3.12.3 State diagram

2 The PortAnnounceInformation state machine shall implement the function specified by the state diagram in
3 Figure 10-14, the local variables specified in 10.3.12.1, the functions specified in 10.3.12.2, and the relevant
4 global variables specified in 10.2.4, 10.2.5, 10.3.9, and 10.3.10. This state machine is used only if
5 externalPortConfigurationEnabled is FALSE (if this variable is TRUE, the PortAnnounceInformationExt
6 state machine of 10.3.14.3 is used instead). The state machine receives new qualified Announce information
7 from the PortAnnounceReceive state machine (see 10.3.11) of the same PTP Port and determines if the
8 Announce information is better than the current best timeTransmitter information it knows. The state
9 machine also updates the current best timeTransmitter information when it receives updated PTP Port state
10 information from the PortStateSelection state machine (see 10.3.13) and when announce receipt timeout or,
11 when gmPresent is TRUE, sync receipt timeout occurs.

1

**Figure 10-14—PortAnnounceInformation state machine**

1 10.3.13 PortStateSelection state machine

2 10.3.13.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-15 (in 10.3.13.3):

4 **10.3.13.1.1 systemIdentityChange:** A Boolean variable that notifies the current state machine when the systemIdentity (see 10.3.2) of this PTP Instance has changed. This variable is reset by this state machine.

6 The systemIdentity changes when at least one of the attributes priority1, clockClass, clockAccuracy, offsetScaledLogVariance, and priority2 changes (e.g., due to management action, degradation or loss of the ClockSource). The systemIdentity also includes the attribute clockIdentity, but this attribute does not change.

10 **10.3.13.1.2 asymmetryMeasurementModeChange:** A Boolean variable that notifies the current state machine when the per-port variable asymmetryMeasurementMode (see 10.2.5.2) changes on any port. This variable is reset by this state machine. There is one instance of asymmetryMeasurementModeChange for all the domains (per port). The variable is accessible by all the domains.

14 10.3.13.2 State machine functions

15 **10.3.13.2.1 updStateDisabledTree():** Sets all the elements of the selectedState array (see 10.2.4.20) to DisabledPort. Sets lastGmPriority to all ones. Sets the pathTrace array (see 10.3.9.23) to contain the single element thisClock (see 10.2.4.22).

18 **10.3.13.2.2 clearReselectTree():** Sets all the elements of the reselect array (see 10.3.9.1) to FALSE.

19 **10.3.13.2.3 setSelectedTree():** Sets all the elements of the selected array (see 10.3.9.2) to TRUE.

20 **10.3.13.2.4 updStatesTree():** Performs the following operations (see 10.3.4 and 10.3.5 for details on the priority vectors):

22 a) Computes the gmPathPriorityVector for each PTP Port that has a portPriorityVector and for which neither announce receipt timeout nor, if gmPresent is TRUE, sync receipt timeout have occurred,

24 b) Saves gmPriority (see 10.3.9.21) in lastGmPriority (see 10.3.9.22), computes the gmPriorityVector for the PTP Instance and saves it in gmPriority, chosen as the best of the set consisting of the systemPriorityVector (for this PTP Instance) and the gmPathPriorityVector for each PTP Port for which the clockIdentity of the timeTransmitter port is not equal to thisClock (see 10.2.4.22),

28 c) Sets the per PTP Instance global variables leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource as follows:

30 1) If the gmPriorityVector was set to the gmPathPriorityVector of one of the ports, then leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource are set to annLeap61, annLeap59, annCurrentUtcOffsetValid, annPtpTimescale, annTimeTraceable, annFrequencyTraceable, annCurrentUtcOffset, and annTimeSource, respectively, for that PTP Port.

35 2) If the gmPriorityVector was set to the systemPriorityVector, then leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource are set to sysLeap61, sysLeap59, sysCurrentUtcOffsetValid, sysPtpTimescale, sysTimeTraceable, sysFrequencyTraceable, sysCurrentUtcOffset, and sysTimeSource, respectively.

40 d) Updates the timePropertiesDS members leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource with the values of the corresponding global variables computed in item c) above.

- 1 e) Updates the parentDS members grandmasterIdentity, grandmasterClockQuality.clockClass, grandmasterClockQuality.clockAccuracy, grandmasterClockQuality.offsetScaledLogVariance, grandmasterPriority1, grandmasterPriority2 with the clockIdentity, clockClass, clockAccuracy, offsetScaledLogVariance, priority1, and priority2 attributes, respectively, of the systemIdentity component of the gmPriority computed in item b) above.
- 2 f) Updates the parentDS member parentPortIdentity with the sourcePortIdentity component of the gmPriority computed in item b) above.
- 3 g) Computes the timeTransmitterPriorityVector for each PTP Port.
- 4 h) Computes timeTransmitterStepsRemoved, which is equal to one of the following:
 - 10 1) messageStepsRemoved (see 10.3.10.9) for the PTP Port associated with the gmPriorityVector, incremented by 1, if the gmPriorityVector is not the systemPriorityVector, or
 - 11 2) 0 if the gmPriorityVector is the systemPriorityVector.
- 13 i) Sets currentDS.stepsRemoved equal to timeTransmitterStepsRemoved.
- 14 j) Assigns the PTP Port state for PTP Port j, and sets selectedState[j] equal to this PTP Port state, as follows, for j = 1, 2, ..., numberPorts:
 - 16 1) If the PTP Port is disabled (infoIs == Disabled), then selectedState[j] is set to DisabledPort.
 - 17 2) If asymmetryMeasurementMode is TRUE, then selectedState[j] is set to PassivePort, and updInfo is set to FALSE.
 - 19 3) If announce receipt timeout, or sync receipt timeout with gmPresent set to TRUE, has occurred (infoIs = Aged), then selectedState[j] is set to TimeTransmitterPort, and updInfo is set to TRUE.
 - 22 4) If the portPriorityVector was derived from another PTP Port on the PTP Instance or from the PTP Instance itself as the root (infoIs == Mine), then selectedState[j] is set to TimeTransmitterPort. In addition, updInfo is set to TRUE if the portPriorityVector differs from the timeTransmitterPriorityVector or portStepsRemoved differs from timeTransmitterStepsRemoved.
 - 27 5) If the portPriorityVector was received in an Announce message, announce receipt timeout, or sync receipt timeout with gmPresent TRUE, has not occurred (infoIs == Received), and the gmPriorityVector is now derived from the portPriorityVector, then selectedState[j] is set to TimeReceiverPort, and updInfo is set to FALSE. The per port global variable receivedPathTrace, for this port, is copied to the per PTP Instance global array pathTrace, and, if it is not empty, thisClock is appended to pathTrace.
 - 33 6) If the portPriorityVector was received in an Announce message, announce receipt timeout, or sync receipt timeout with gmPresent TRUE, has not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, the timeTransmitterPriorityVector is not better than the portPriorityVector, and the sourcePortIdentity component of the portPriorityVector *does not* reflect another PTP Port on the PTP Instance, then selectedState[j] is set to PassivePort, and updInfo is set to FALSE.
 - 39 7) If the portPriorityVector was received in an Announce message, announce receipt timeout, or sync receipt timeout with gmPresent TRUE, has not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, the timeTransmitterPriorityVector is not better than the portPriorityVector, and the sourcePortIdentity component of the portPriorityVector *does* reflect another PTP Port on the PTP Instance, then selectedState[j] set to PassivePort, and updInfo is set to FALSE.
 - 45 8) If the portPriorityVector was received in an Announce message, announce receipt timeout, or sync receipt timeout with gmPresent TRUE, has not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, and the timeTransmitterPriorityVector is better than the portPriorityVector, then selectedState[j] is set to TimeTransmitterPort, and updInfo is set to TRUE.
- 50 k) Updates gmPresent as follows:

- 1 1) gmPresent is set to TRUE if the priority1 field of the rootSystemIdentity of the
2 gmPriorityVector is less than 255.
- 3 2) gmPresent is set to FALSE if the priority1 field of the rootSystemIdentity of the
4 gmPriorityVector is equal to 255.
- 5 l) Assigns the PTP Port state for PTP Port 0 (see 8.5.2.3), and sets selectedState[0] as follows:
 - 6 1) if selectedState[j] is set to TimeReceiverPort for any PTP Port with portNumber j, j = 1, 2, ...,
7 numberPorts, selectedState[0] is set to PassivePort.
 - 8 2) if selectedState[j] is *not* set to TimeReceiverPort for any PTP Port with portNumber j, j = 1, 2,
9 ..., numberPorts, selectedState[0] is set to TimeReceiverPort.
- 10 m) If the clockIdentity member of the systemIdentity (see 10.3.2) member of gmPriority (see 10.3.9.21)
11 is equal to thisClock (see 10.2.4.22), i.e., if the current PTP Instance is the Grandmaster PTP
12 Instance, the pathTrace array is set to contain the single element thisClock (see 10.2.4.22).

13 **10.3.13.3 State diagram**

14 The PortStateSelection state machine shall implement the function specified by the state diagram in
15 Figure 10-15, the functions specified in 10.3.13.1, and the relevant global variables specified in 10.2.4,
16 10.2.5, 10.3.9, and 10.3.10. This state machine is used only if externalPortConfigurationEnabled is FALSE
17 (if this variable is TRUE, the PortStateSettingExt state machine is used instead). The state machine updates
18 the gmPathPriority vector for each PTP Port of the PTP Instance, the gmPriorityVector for the PTP Instance,
19 and the timeTransmitterPriorityVector for each PTP Port of the PTP Instance. The state machine determines
20 the PTP Port state for each PTP Port and updates gmPresent.

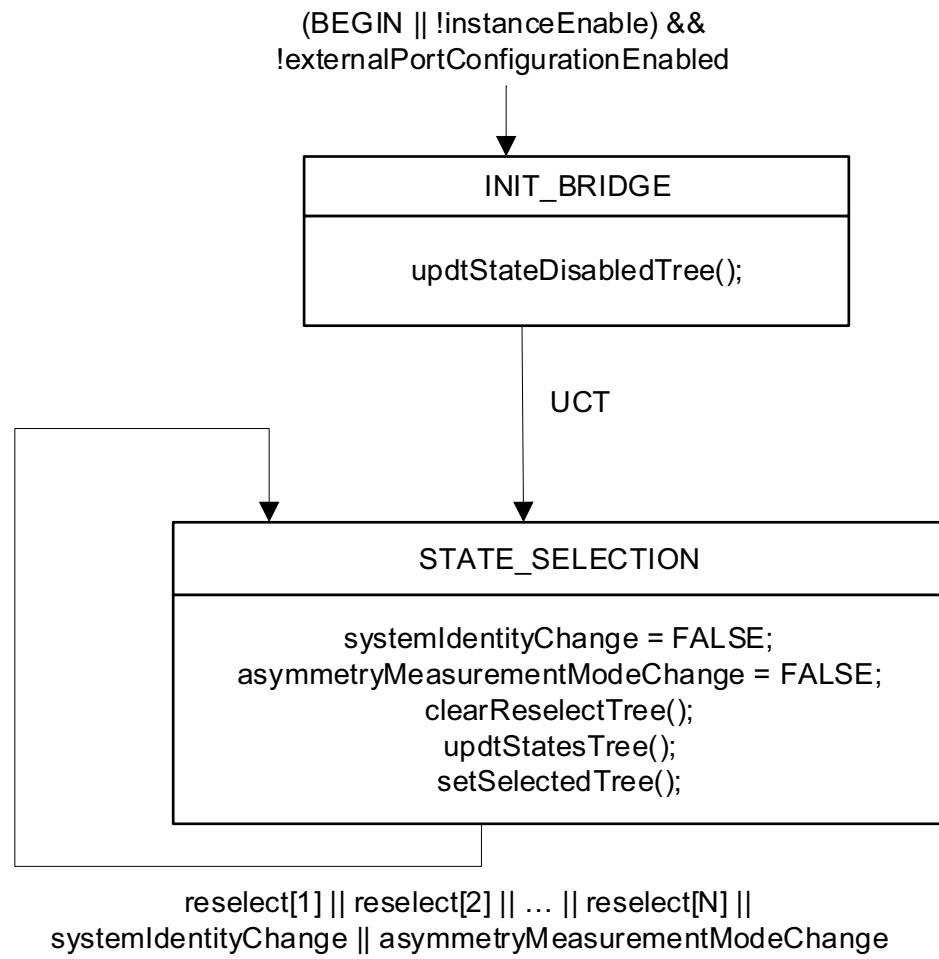


Figure 10-15—PortStateSelection state machine

1 **10.3.14 PortAnnounceInformationExt state machine**

2 **10.3.14.1 State machine variables**

3 The following variables are used in the state diagram in Figure 10-16 (in 10.3.14.3):

4 **10.3.14.1.1 rcvAnnouncePAIE:** A Boolean variable that notifies the current state machine when
5 Announce message information is received from the MD entity of the same PTP Port. This variable is reset
6 by this state machine.

7 **10.3.14.1.2 messagePriorityPAIE:** The messagePriorityVector corresponding to the received Announce
8 information. The data type for messagePriorityPAIE is UIInteger224 (see 10.3.4).

9 **10.3.14.2 State machine functions**

10 **10.3.14.2.1 rcvInfoExt (rcvdAnnouncePtr):** Decodes the messagePriorityVector (see 10.3.4 and 10.3.5)
11 and stepsRemoved (see 10.6.3.2.6) field from the Announce information pointed to by rcvdAnnouncePtr
12 (see 10.3.10.13), and then stores the messagePriorityVector and stepsRemoved field value in
13 messagePriorityPAIE and messageStepsRemoved, respectively. If a path trace TLV is present in the
14 Announce message and the portState of the PTP Port is TimeReceiverPort, the pathSequence array field of
15 the TLV is copied to the global array pathTrace, and thisClock is appended to pathTrace (i.e., is added to the
16 end of the array).

17 **10.3.14.2.2 recordOtherAnnounceInfo():** Saves the flags leap61, leap59, currentUtcOffsetValid,
18 ptpTimescale, timeTraceable, and frequencyTraceable, and the fields currentUtcOffset and timeSource, of
19 the received Announce message for this PTP Port. The values are saved in the per-PTP Port global variables
20 annLeap61, annLeap59, annCurrentUtcOffsetValid, annPtpTimescale, annTimeTraceable,
21 annFrequencyTraceable, annCurrentUtcOffset, and annTimeSource (see 10.3.10.16 through 10.3.10.23).

22 **10.3.14.3 State diagram**

23 The PortAnnounceInformationExt state machine shall implement the function specified by the state diagram
24 in Figure 10-16, the local variables specified in 10.3.14.1, the functions specified in 10.3.14.2, and the
25 relevant global variables specified in 10.2.4, 10.2.5, 10.3.9, and 10.3.10. This state machine is used only if
26 externalPortConfigurationEnabled is TRUE (if this variable is FALSE, the PortAnnounceInformation state
27 machine of 10.3.12.3 is used instead). The state machine receives Announce information from the MD entity
28 of the same PTP Port and saves the information.

29 **10.3.15 PortStateSettingExt state machine**

30 **10.3.15.1 State machine variables**

31 The following variables are used in the state diagram in Figure 10-17 (in 10.3.15.3):

32 **10.3.15.1.1 disabledExt:** A Boolean variable that notifies the current state machine (i.e., when it is set to
33 TRUE) when at least one of the variables portOper, ptpPortEnabled, or asCapable, for this PTP Port, has
34 changed from TRUE to FALSE. This variable is reset by this state machine.

35 **10.3.15.1.2 reenabledExt:** A Boolean variable that notifies the current state machine (i.e., when it is set to
36 TRUE) when all of the variables portOper, ptpPortEnabled, and asCapable, for this PTP Port, that are
37 FALSE have changed to TRUE. This variable is reset by this state machine.

38 **10.3.15.1.3 asymmetryMeasurementModeChangeThisPort:** A Boolean variable that notifies the current
39 state machine when the per-port variable asymmetryMeasurementMode (see 10.2.5.2) changes on this port.

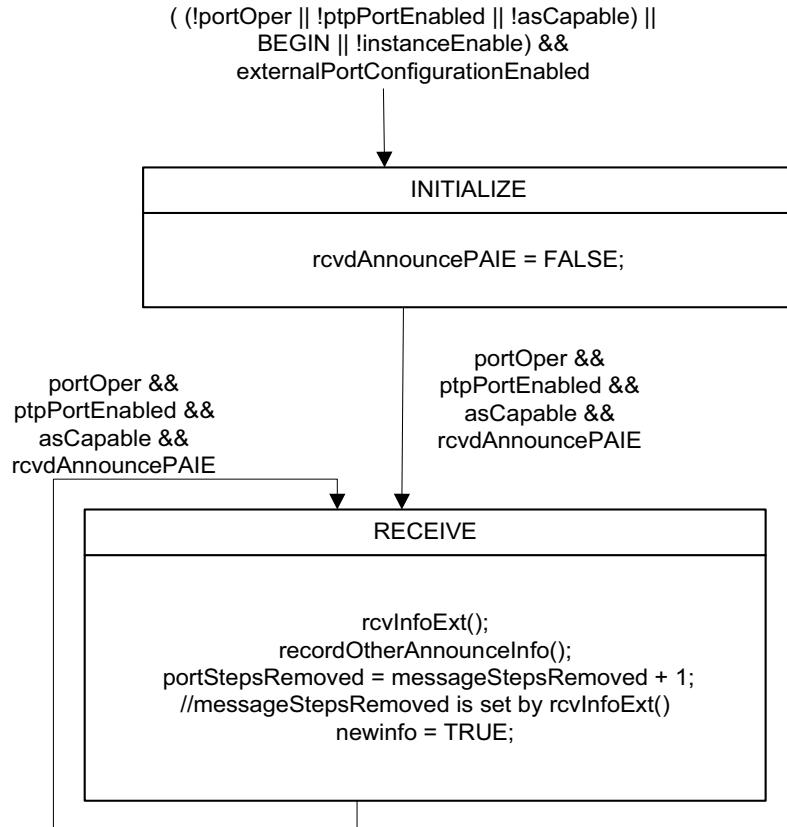


Figure 10-16—PortAnnounceInformationExt state machine

1 This variable is reset by this state machine. There is one instance of
2 asymmetryMeasurementModeChangeThisPort for all the domains (per port). The variable is accessible by
3 all the domains.

4 **10.3.15.1.4 rcvdPortStateInd:** A Boolean variable that notifies the current state machine (i.e., when it is set
5 to TRUE) when the PTP Port state of this PTP Port has been externally set. This variable is reset by this state
6 machine.

7 **10.3.15.1.5 portStateInd:** An Enumeration2 that indicates the PTP Port state that has been set. The values
8 are TimeTransmitterPort, TimeReceiverPort, and PassivePort.

9 NOTE—The PTP Port state can be externally set to DisabledPort by setting portOper or ptpPortEnabled to FALSE. The
10 PTP Port state is set to DisabledPort when asCapable becomes FALSE.

11 **10.3.15.2 State machine functions**

12 **10.3.15.2.1 resetStateTree(j):** Sets selectedState[j] (see 10.2.4.20) to
13 externalPortConfigurationPortDS.desiredState. Sets the pathTrace array (see 10.3.9.23) to contain the single
14 element thisClock (see 10.2.4.22) if no PTP Port of the PTP Instance has the PTP Port state
15 TimeReceiverPort.

1 **10.3.15.2.2 updPortState(j):** Performs the following operations for PTP Port j (see 10.3.4 and 10.3.5 for
2 details on the priority vectors):

- 3 a) Sets the per PTP Instance global variables leap61, leap59, currentUtcOffsetValid, ptptimescale,
4 timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource as follows:
 - 5 1) If the PTP Port state of any PTP Port of this **PTP Instance (see 8.5.2.3)** is TimeReceiverPort,
6 then leap61, leap59, currentUtcOffsetValid, ptptimescale, timeTraceable, frequencyTraceable,
7 currentUtcOffset, and timeSource are set to annLeap61, annLeap59,
8 annCurrentUtcOffsetValid, annPtptimescale, annTimeTraceable, annFrequencyTraceable,
9 annCurrentUtcOffset, and annTimeSource, respectively, for that PTP Port.
 - 10 2) If no PTP Port of this **PTP Instance (see 8.5.2.3)** has the PTP Port state TimeReceiverPort, then
11 leap61, leap59, currentUtcOffsetValid, ptptimescale, timeTraceable, frequencyTraceable,
12 currentUtcOffset, and timeSource are set to sysLeap61, sysLeap59, sysCurrentUtcOffsetValid,
13 sysPtptimescale, sysTimeTraceable, sysFrequencyTraceable, sysCurrentUtcOffset, and
14 sysTimeSource, respectively.
- 15 b) Update the **timePropertiesDS** members leap61, leap59, currentUtcOffsetValid, ptptimescale,
16 timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource with the values of the
17 corresponding global variables computed in item a) above.
- 18 c) Computes timeTransmitterStepsRemoved as follows:
 - 19 1) If the PTP Port state of any PTP Port of this **PTP Instance (see 8.5.2.3)** is TimeReceiverPort,
20 then timeTransmitterStepsRemoved is set equal to portStepsRemoved for that PTP Port.
 - 21 2) If no PTP Port of this **PTP Instance (see 8.5.2.3)** has the PTP Port state TimeReceiverPort, then
22 timeTransmitterStepsRemoved is set equal to 0.
- 23 d) Sets **currentDS.stepsRemoved** equal to **timeTransmitterStepsRemoved**.
- 24 e) Assigns the PTP Port state for PTP Port j, and sets selectedState[j] equal to this PTP Port state, as
25 follows:
 - 26 1) If disabledExt is TRUE, selectedState[j] is set to DisabledPort, else
 - 27 2) If asymmetryMeasurementMode is TRUE, selectedState[j] is set to PassivePort, else
 - 28 3) selectedState[j] is set to portStateInd. If portStateInd is equal to TimeTransmitterPort, newInfo
29 is set to TRUE.
- 30 f) Updates gmPresent as follows:
 - 31 1) If the PTP Port state of any PTP Port of this **PTP Instance (see 8.5.2.3)** is TimeReceiverPort
32 and the priority1 field of the rootSystemIdentity of the messagePriorityPAIE of the
33 timeReceiver port is less than 255, gmPresent is set to TRUE, else
 - 34 2) If the PTP Port state of any PTP Port of this **PTP Instance (see 8.5.2.3)** is TimeReceiverPort
35 and the priority1 field of the rootSystemIdentity of the messagePriorityPAIE of the
36 timeReceiver PTP Port is equal to 255, gmPresent is set to FALSE, else
 - 37 3) If no PTP Port of this **PTP Instance (see 8.5.2.3)** has the PTP Port state TimeReceiverPort,
38 gmPresent is set to TRUE if priority1 for this PTP Instance is less than 255 and FALSE if
39 priority1 for this PTP Instance is equal to 255.
- 40 g) Assigns the PTP Port state for PTP Port 0, and sets selectedState[0] as follows:
 - 41 1) If selectedState[j] is set to TimeReceiverPort, selectedState[0] is set to PassivePort.
 - 42 2) If selectedState[j] is not set to TimeReceiverPort and selectedState[k] is not equal to
43 TimeReceiverPort for every k not equal to 0 or j, selectedState[0] is set to TimeReceiverPort.
- 44 h) Computes the gmPriorityVector as follows:
 - 45 1) If selectedState[j] is set to TimeReceiverPort, the gmPriorityVector is set equal to
46 messagePriorityPAIE for PTP Port j.

- 1 2) If selectedState[j] is *not* set to TimeReceiverPort and selectedState[k] is not equal to
 2 TimeReceiverPort for every k not equal to 0 or j, the gmPriorityVector is set equal to the
 3 systemPriorityVector.
- 4 i) Update the parentDS members grandmasterIdentity, grandmasterClockQuality.clockClass,
 5 grandmasterClockQuality.clockAccuracy, grandmasterClockQuality.offsetScaledLogVariance,
 6 grandmasterPriority1, grandmasterPriority2 with the clockIdentity, clockClass, clockAccuracy,
 7 offsetScaledLogVariance, priority1, and priority2 attributes, respectively, of the systemIdentity
 8 component of the gmPriorityVector computed in item h) above.
- 9 j) Update the parentDS member parentPortIdentity with the sourcePortIdentity component of the
 10 gmPriorityVector computed in item h) above.
- 11 k) Computes the timeTransmitterPriorityVector for PTP Port j.
- 12 l) If no PTP Port of this PTP Instance has the PTP Port state TimeReceiverPort, the pathTrace array is
 13 set to contain the single element thisClock (see 10.2.4.22).

14 **10.3.15.3 State diagram**

15 The PortStateSettingExt state machine shall implement the function specified by the state diagram
 16 in Figure 10-17, the local variables specified in 10.3.15.1, the functions specified in 10.3.15.2, and the
 17 relevant global variables specified in 10.2.4, 10.2.5, 10.3.9, and 10.3.10. This state machine is used only if
 18 externalPortConfigurationEnabled is TRUE (if this variable is FALSE, the PortStateSelection state machine
 19 of 10.3.13.3 is used instead). A separate instance of this state machine runs on each PTP Port (unlike the
 20 PortStateSelection state machine, for which a single instance runs in the PTP Instance and performs
 21 operations on all the ports).

22 The state machine updates the gmPriorityVector for the PTP Instance and the timeTransmitterPriorityVector
 23 for each PTP Port of the PTP Instance. The state machine determines the PTP Port state for each PTP Port
 24 and updates gmPresent.

25 NOTE—It is possible to use the external port configuration mechanism to misconfigure the network, e.g., to produce a
 26 configuration where one or more PTP Instances have more than one timeReceiver port. Detecting and correcting
 27 misconfigurations is outside the scope of this standard.

28 **10.3.16 PortAnnounceTransmit state machine**

29 **10.3.16.1 State machine variables**

30 The following variables are used in the state diagram in Figure 10-18 (in 10.3.16.3):

31 **10.3.16.1.1 announceSendTime:** The time, relative to the LocalClock, at which the next transmission of
 32 Announce information is to occur. The data type for announceSendTime is UScaledNs.

33 **10.3.16.1.2 numberAnnounceTransmissions:** A count of the number of consecutive Announce message
 34 transmissions after the AnnounceIntervalSetting state machine (see Figure 10-19 in 10.3.17.3) has set
 35 announceSlowdown (see 10.3.10.2) to TRUE. The data type for numberAnnounceTransmissions is
 36 UInteger8.

37 **10.3.16.1.3 interval2:** A local variable that holds either announceInterval or oldAnnounceInterval. The data
 38 type for interval2 is UScaledNs.

39 **10.3.16.2 State machine functions**

40 **10.3.16.2.1 txAnnounce-0:** Transmits Announce information to the MD entity of this PTP Port. The
 41 Announce information is set as follows:

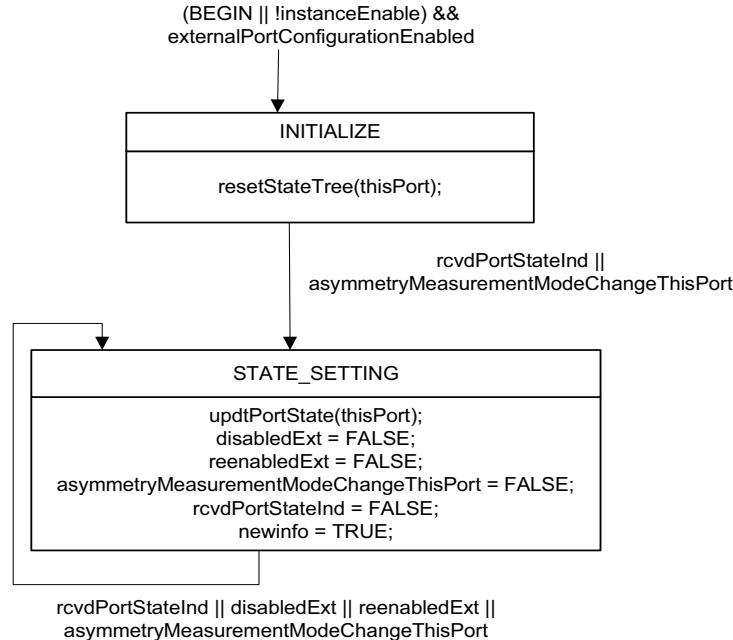


Figure 10-17—PortStateSettingExt state machine

- 1 a) The components of the messagePriorityVector are set to the values of the respective components of the timeTransmitterPriorityVector of this PTP Port.
- 2 b) The grandmasterIdentity, grandmasterClockQuality, grandmasterPriority1, and grandmasterPriority2 fields of the Announce message are set equal to the corresponding components of the messagePriorityVector.
- 3 c) The value of the stepsRemoved field of the Announce message is set equal to timeTransmitterStepsRemoved.
- 4 d) The Announce message flags leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, and frequencyTraceable, and the Announce message fields currentUtcOffset and timeSource, are set equal to the values of the global variables leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource, respectively (see 10.3.9.4 through 10.3.9.11).
- 5 e) The sequenceId field of the Announce message is set in accordance with 10.5.7.
- 6 f) A path trace TLV (see 10.6.3.3) is constructed, with its pathSequence field (see 10.6.3.3.4) set equal to the pathTrace array (see 10.3.9.23). If appending the path trace TLV to the Announce message does not cause the media-dependent layer frame to exceed any respective maximum size, the path trace TLV is appended to the Announce message; otherwise, it is not appended. If the pathTrace array is empty, the path trace TLV is not appended. See 10.3.9.23 for a description of the path trace feature.

20 10.3.16.3 State diagram

21 The PortAnnounceTransmit state machine shall implement the function specified by the state diagram in 22 Figure 10-18, the local variables specified in 10.3.16.1, the functions specified in 10.3.16.2, and the relevant 23 global variables specified in 10.2.4, 10.2.5, 10.3.9, and 10.3.10. The state machine transmits Announce 24 information to the MD entity when an announce interval has elapsed, PTP Port states have been updated,

1 and portPriority and portStepsRemoved information has been updated with newly determined
2 timeTransmitterPriority and timeTransmitterStepsRemoved information.

3 NOTE—When the external port configuration option is used (i.e., externalPortConfigurationEnabled is TRUE; see
4 10.3.9.24) the values of the variables updInfo and selected do not affect the operation of the PortAnnounceTransmit
5 state machine because the term of the conditions in which they appear, i.e., (selected && !updInfo) ||
6 externalPortConfiguartionEnabled, evaluates to TRUE when externalPortConfigurationEnabled is TRUE.

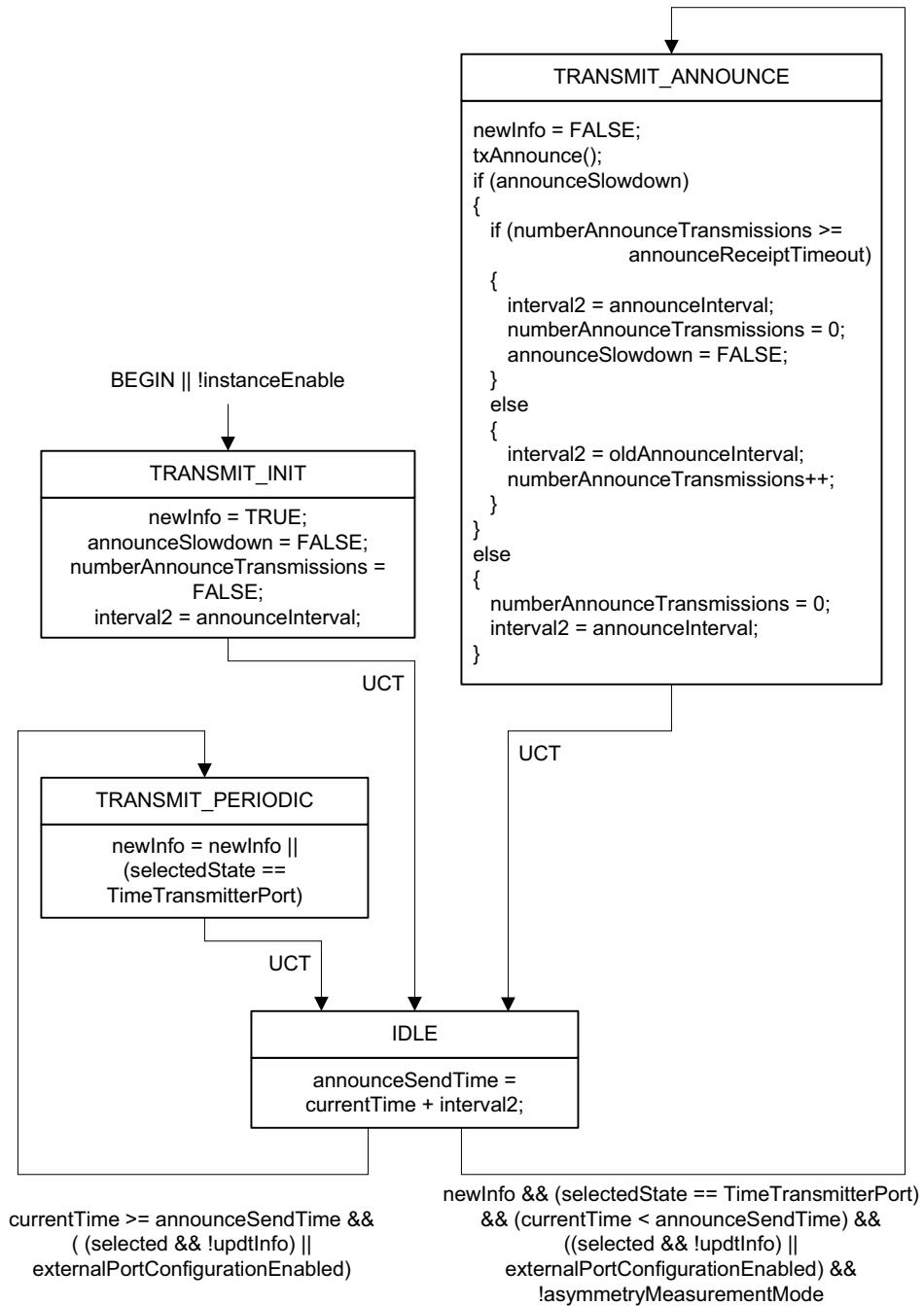


Figure 10-18—PortAnnounceTransmit state machine

2 10.3.17 AnnounceIntervalSetting state machine

1 10.3.17.1 State machine variables

- 2 The following variables are used in the state diagram in Figure 10-19 (in 10.3.17.3):
- 3 **10.3.17.1.1 rcvdSignalingMsg2:** A Boolean variable that notifies the current state machine when a
4 Signaling message that contains a Message Interval Request TLV (see 10.6.4.3) is received. This variable is
5 reset by the current state machine.
- 6 **10.3.17.1.2 rcvdSignalingPtrAIS:** A pointer to a structure whose members contain the values of the fields
7 of the received Signaling message that contains a Message Interval Request TLV (see 10.6.4.3).
- 8 **10.3.17.1.3 logSupportedAnnounceIntervalMax:** The maximum supported logarithm to base 2 of the
9 announce interval. The data type for logSupportedAnnounceIntervalMax is Integer8.
- 10 **10.3.17.1.4 logSupportedClosestLongerAnnounceInterval:** The logarithm to base 2 of the announce
11 interval, such that logSupportedClosestLongerAnnounceInterval > logRequestedAnnounceInterval, that is
12 numerically closest to logRequestedAnnounceInterval, where logRequestedAnnounceInterval is the
13 argument of the function computeLogAnnounceInterval() (see 10.3.17.2.2). The data type for
14 logSupportedClosestLongerAnnounceInterval is Integer8.
- 15 **10.3.17.1.5 computedLogAnnounceInterval:** A variable used to hold the result of the function
16 computeLogAnnounceInterval(). The data type for computedLogAnnounceInterval is Integer8.
- 17 **10.3.17.1.6 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 10-19). The
18 data type for TEMP is Integer16.

19 10.3.17.2 State machine functions

- 20 **10.3.17.2.1 isSupportedLogAnnounceInterval (logAnnounceInterval):** A Boolean function that returns
21 TRUE if the announce interval given by the argument logAnnounceInterval is supported by the PTP Port
22 and FALSE if the announce interval is not supported by the PTP Port. The argument logAnnounceInterval
23 has the same data type and format as the field logAnnounceInterval of the message interval request TLV (see
24 10.6.4.3.8).

1 **10.3.17.2.2 computeLogAnnounceInterval (logRequestedAnnounceInterval):** An Integer8 function that
 2 computes and returns the logAnnounceInterval, based on the logRequestedAnnounceInterval. This function
 3 is defined as indicated below. It is defined here so that the detailed code that it invokes does not need to be
 4 placed into the state machine diagram.

```

5 Integer8 computeLogAnnounceInterval (logRequestedAnnounceInterval)
6 Integer8 logRequestedAnnounceInterval;
7 {
8     Integer8 logSupportedAnnounceIntervalMax,
9         logSupportedClosestLongerAnnounceInterval;
10    if (isSupportedLogAnnounceInterval (logRequestedAnnounceInterval))
11        // The requested Announce Interval is supported and returned
12        return (logRequestedAnnounceInterval)
13    else
14    {
15        if (logRequestedAnnounceInterval > logSupportedAnnounceIntervalMax)
16            // Return the largest supported logAnnounceInterval, even if smaller than
17 the requested interval
18            return (logSupportedAnnounceIntervalMax);
19        else
20            // Return the smallest supported logAnnounceInterval that is still larger
21 than
22            // the requested interval.
23            return (logSupportedClosestLongerAnnounceInterval);
24    }
25 }
```

26 **10.3.17.3 State diagram**

27 The AnnounceIntervalSetting state machine shall implement the function specified by the state diagram in
 28 Figure 10-19, the local variables specified in 10.3.17.1, the functions specified in 10.3.17.2, the messages
 29 specified in 10.6, the relevant global variables specified in 10.2.5 and 10.3.10, the relevant managed objects
 30 specified in 14.8, and the relevant timing attributes specified in 10.7. This state machine is responsible for
 31 setting the global variables that give the duration of the mean interval between successive Announce
 32 messages, both at initialization and in response to the receipt of a Signaling message that contains a Message
 33 Interval Request TLV (see 10.6.4.3).

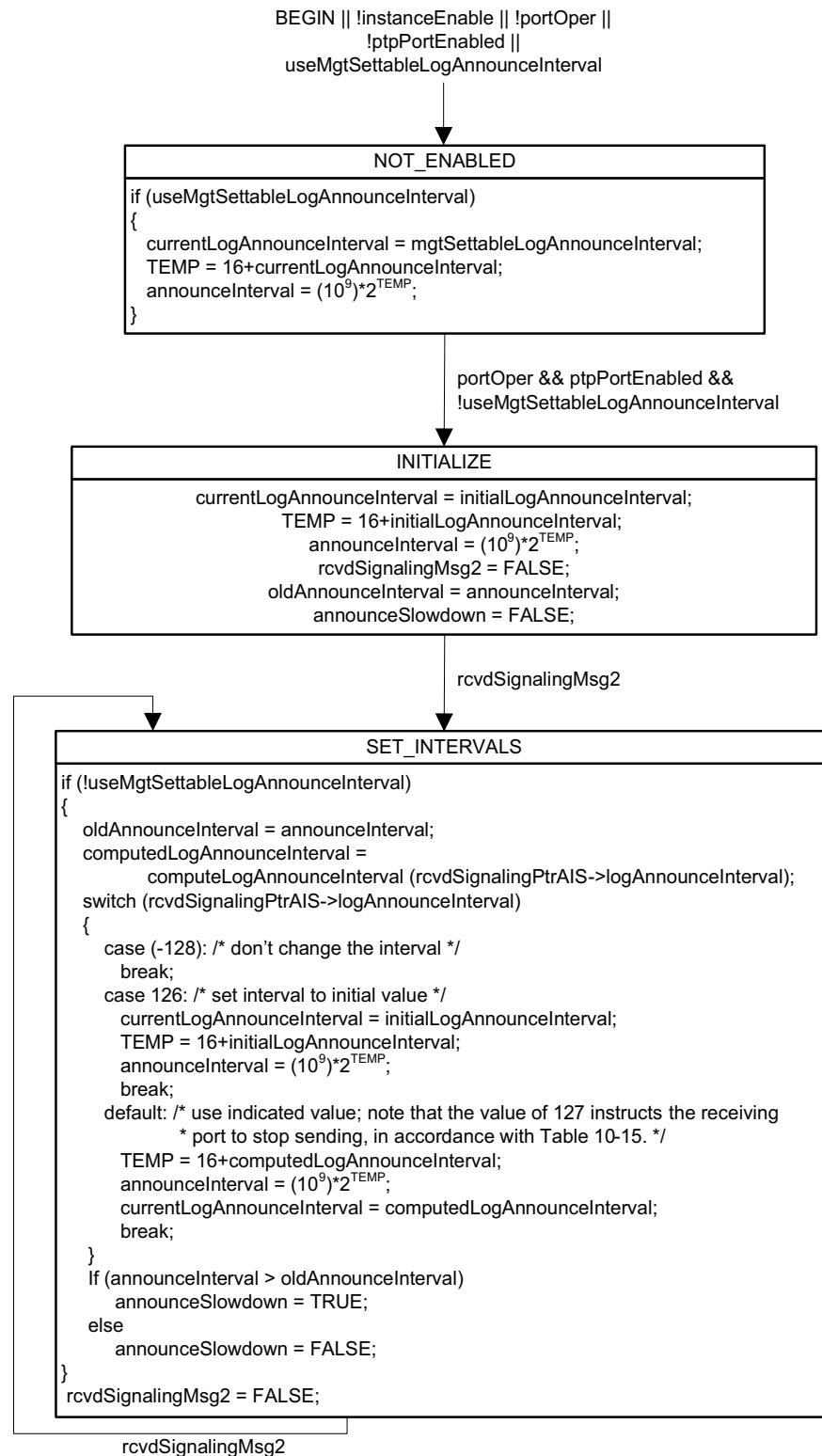


Figure 10-19—AnnounceIntervalSetting state machine

1 10.3.18 SyncIntervalSetting state machine

2 10.3.18.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-20 (in 10.3.18.3):

4 10.3.18.1.1 **recvSignalingMsg3**: A Boolean variable that notifies the current state machine when a
5 Signaling message that contains a Message Interval Request TLV (see 10.6.4.3) is received. This variable is
6 reset by the current state machine.

7 10.3.18.1.2 **recvSignalingPtrSIS**: A pointer to a structure whose members contain the values of the fields
8 of the received Signaling message that contains a Message Interval Request TLV (see 10.6.4.3).

9 10.3.18.1.3 **logSupportedSyncIntervalMax**: The maximum supported logarithm to base 2 of the sync
10 interval. The data type for logSupportedSyncIntervalMax is Integer8.

11 10.3.18.1.4 **logSupportedClosestLongerSyncInterval**: The logarithm to base 2 of the sync interval, such
12 that $\text{logSupportedClosestLongerSyncInterval} > \text{logRequestedSyncInterval}$, that is numerically closest to
13 $\text{logRequestedSyncInterval}$, where $\text{logRequestedSyncInterval}$ is the argument of the function
14 $\text{computeLogSyncInterval}()$ (see 10.3.18.2.2). The data type for logSupportedClosestLongerSyncInterval is
15 Integer8.

16 10.3.18.1.5 **computedLogSyncInterval**: A variable used to hold the result of the function
17 $\text{computeLogSyncInterval}()$. The data type for computedLogSyncInterval is Integer8.

18 10.3.18.1.6 **TEMP**: A temporary variable used to reduce clutter in the state diagram (see Figure 10-20). The
19 data type for TEMP is Integer16.

20 10.3.18.2 State machine functions

21 10.3.18.2.1 **isSupportedLogSyncInterval (logSyncInterval)**: A Boolean function that returns TRUE if the
22 sync interval given by the argument logSyncInterval is supported by the PTP Port and FALSE if the sync
23 interval is not supported by the PTP Port. The argument logSyncInterval has the same data type and format
24 as the field syncInterval of the message interval request TLV (see 10.6.4.3.7).

1 **10.3.18.2.2 computeLogSyncInterval (logRequestedSyncInterval):** An Integer8 function that computes
 2 and returns the logSyncInterval, based on the logRequestedSyncInterval. This function is defined as
 3 indicated below. It is defined here so that the detailed code that it invokes does not need to be placed into the
 4 state machine diagram.

```
5 Integer8 computeLogSyncInterval (logRequestedSyncInterval)
6 Integer8 logRequestedSyncInterval;
7 {
8     Integer8 logSupportedSyncIntervalMax, logSupportedClosestLongerSyncInterval;
9     if (isSupportedLogSyncInterval (logRequestedSyncInterval))
10         // The requested Sync Interval is supported and returned
11         return (logRequestedSyncInterval)
12     else
13     {
14         if (logRequestedSyncInterval > logSupportedSyncIntervalMax)
15             // Return the largest supported logSyncInterval, even if smaller than the
16 requested interval
17             return (logSupportedSyncIntervalMax);
18         else
19             // Return the smallest supported logSyncInterval that is still larger than
20             // the requested interval.
21             return (logSupportedClosestLongerSyncInterval);
22     }
23 }
```

24 **10.3.18.3 State diagram**

25 The SyncIntervalSetting state machine shall implement the function specified by the state diagram in
 26 Figure 10-20, the local variables specified in 10.3.18.1, the functions specified in 10.3.18.2, the messages
 27 specified in 10.6, the relevant global variables specified in 10.2.5, the relevant managed objects specified in
 28 14.8, and the relevant timing attributes specified in 10.7. This state machine is responsible for setting the
 29 global variables that give the duration of the mean intervals between successive Sync messages, both at
 30 initialization and in response to the receipt of a Signaling message that contains a Message Interval Request
 31 TLV (see 10.6.4.3).

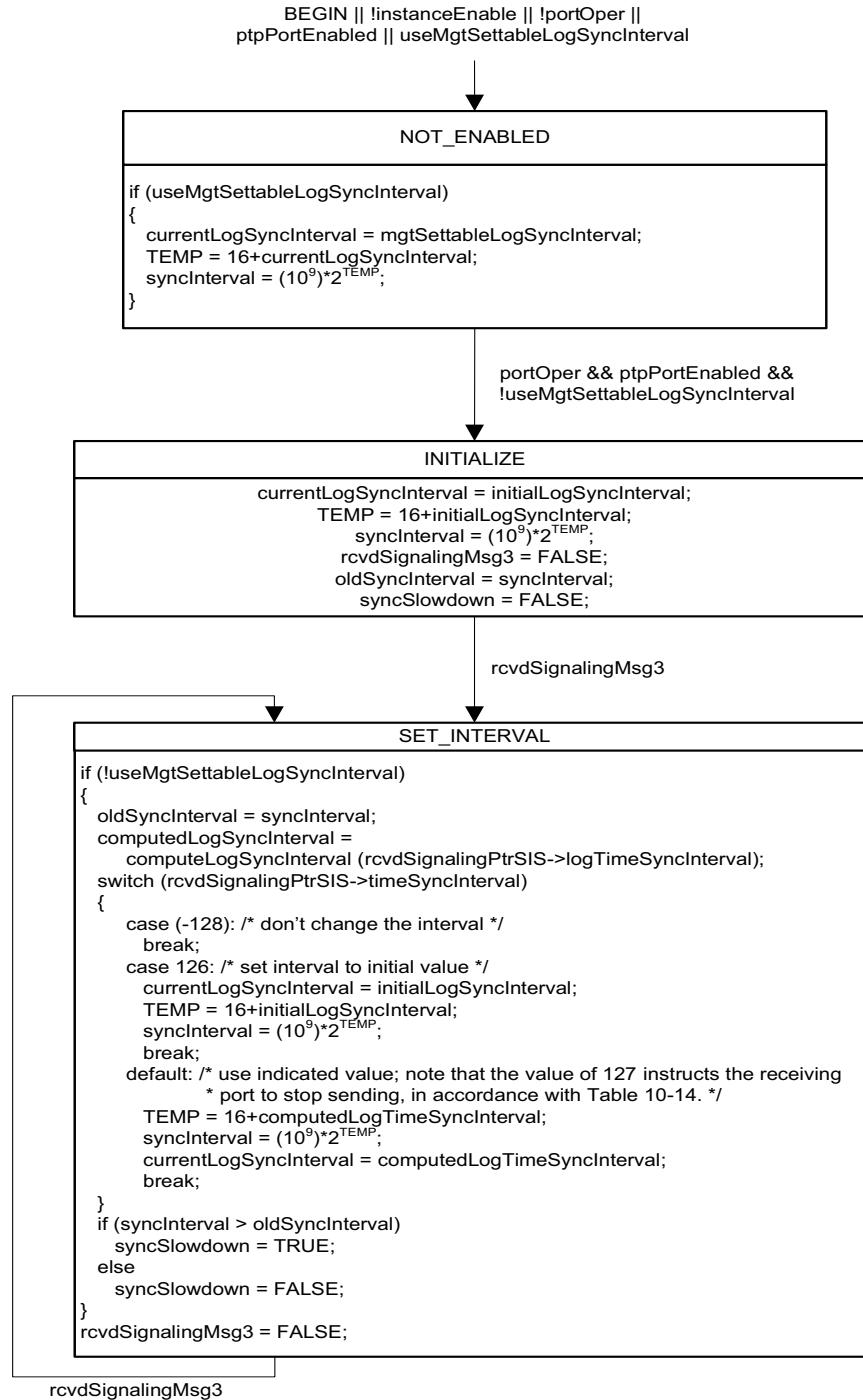


Figure 10-20—SyncIntervalSetting state machine

1 10.4 State machines related to signaling gPTP capability

2 10.4.a Enabling and disabling the state machines

3 If the managed object `gptpCapableStateMachinesEnabled` (see 14.8.54a) is TRUE, the
 4 `GptpCapableTransmit` and `GptpCapableReceive` state machines shall be enabled and shall function as
 5 specified in 10.4.1 and 10.4.2, respectively. The `GptpCapableIntervalSetting` state machine is enabled if it is
 6 implemented.

7 If the managed object `gptpCapableStateMachinesEnabled` is FALSE, the `GptpCapableTransmit` and
 8 `GptpCapableReceive` state machines shall be disabled. The `GptpCapableIntervalSetting` state machine shall
 9 be disabled if it is implemented.

10 If the managed object `gptpCapableStateMachinesEnabled` is FALSE, the global variable
 11 `neighborGptpCapable` for the port (see 10.2.5.16) shall be set to TRUE.

12 NOTE 1—The global variable `neighborGptpCapable` indicates to a PTP Port whether the PTP Port at the other end of the
 13 attached PTP Link is capable of invoking gPTP, and is used in the determination of `asCapable` (see 10.2.5.1, 11.2.2, 12.4,
 14 and 13.4). If `gptpCapableStateMachinesEnabled` is TRUE, the variable is set to TRUE or FALSE by the
 15 `GptpCapableTransmit` state machine. If `gptpCapableStateMachinesEnabled` is FALSE, the variable is automatically set
 16 to TRUE, that is, it is assumed that the network is engineered such that the PTP Port at the other end of the attached link
 17 is capable of invoking gPTP. In this case, it is essential that the network be engineered to fulfill this condition; if
 18 `neighborGptpCapable` is TRUE and the PTP Port at the other end of the link is not capable of invoking gPTP,
 19 undesirable behavior can occur.

20 NOTE 2—If the `GptpCapableTransmit` and `GptpCapableReceive` state machines are disabled, the exchange of peer
 21 delay messages can occur immediately, that is, it is not necessary to wait until gPTP capability is determined by the
 22 `GptpCapableTransmit` and `GptpCapableReceive` state machines.

23 10.4.1 GptpCapableTransmit state machine

24 10.4.1.1 State machine variables

25 The following variables are used in the state diagram in Figure 10-21 (in 10.4.1.3):

26 **10.4.1.1.1 `gPtpCapableSendTime`:** The time, relative to the `LocalClock` entity, when a Signaling message
 27 containing a gPTP-capable TLV is next sent. The data type for `gPtpCapableSendTime` is `UScaledNs`.

28 **10.4.1.1.2 `txSignalingMsgPtr`:** A pointer to a structure whose members contain the values of the fields of a
 29 Signaling message to be transmitted, which contains a gPTP-capable TLV (see 10.6.4.4).

30 **10.4.1.1.3 `interval3`:** A local variable that holds either `gPtpCapableMessageInterval` or
 31 `oldGptpCapableMessageInterval`. The data type for `interval3` is `UScaledNs`.

32 **10.4.1.1.4 `numberGptpCapableMessageTransmissions`:** A count of the number of consecutive
 33 transmissions of Signaling messages that contain a gPTP-capable TLV, after the `GptpCapableIntervalSetting`
 34 state machine (see Figure 10-23 in 10.4.3.3) has set `gPtpCapableMessageSlowdown` (see 10.2.5.19) to
 35 TRUE. The data type for `numberGptpCapableMessageTransmissions` is `UInteger8`.

36 10.4.1.2 State machine functions

37 **10.4.1.2.1 `setGptpCapableTlv()`:** Creates a structure containing the parameters of a Signaling message that
 38 contains a gPTP-capable TLV, to be transmitted (see 10.6.4), and returns a pointer to this structure. The
 39 parameters are set as follows:

- 1 a) logGptpCapableMessageInterval is set to the value of the managed object
- 2 currentLogGptpCapableMessageInterval (see 14.8.28).
- 3 b) The remaining parameters are set as specified in 10.6.4 and 10.6.4.4.

4 **10.4.1.3 State diagram**

5 The GptpCapableTransmit state machine shall implement the function specified by the state diagram in
6 Figure 10-21, the local variables specified in 10.4.1.1, the functions specified in 10.4.1.2, the relevant
7 parameters specified in 10.6.4 and 10.6.4.4, and the relevant timing attributes specified in 10.7. This state
8 machine is responsible for setting the parameters of each Signaling message that contains the gPTP-capable
9 TLV, and causing these Signaling messages to be transmitted at a regular rate.

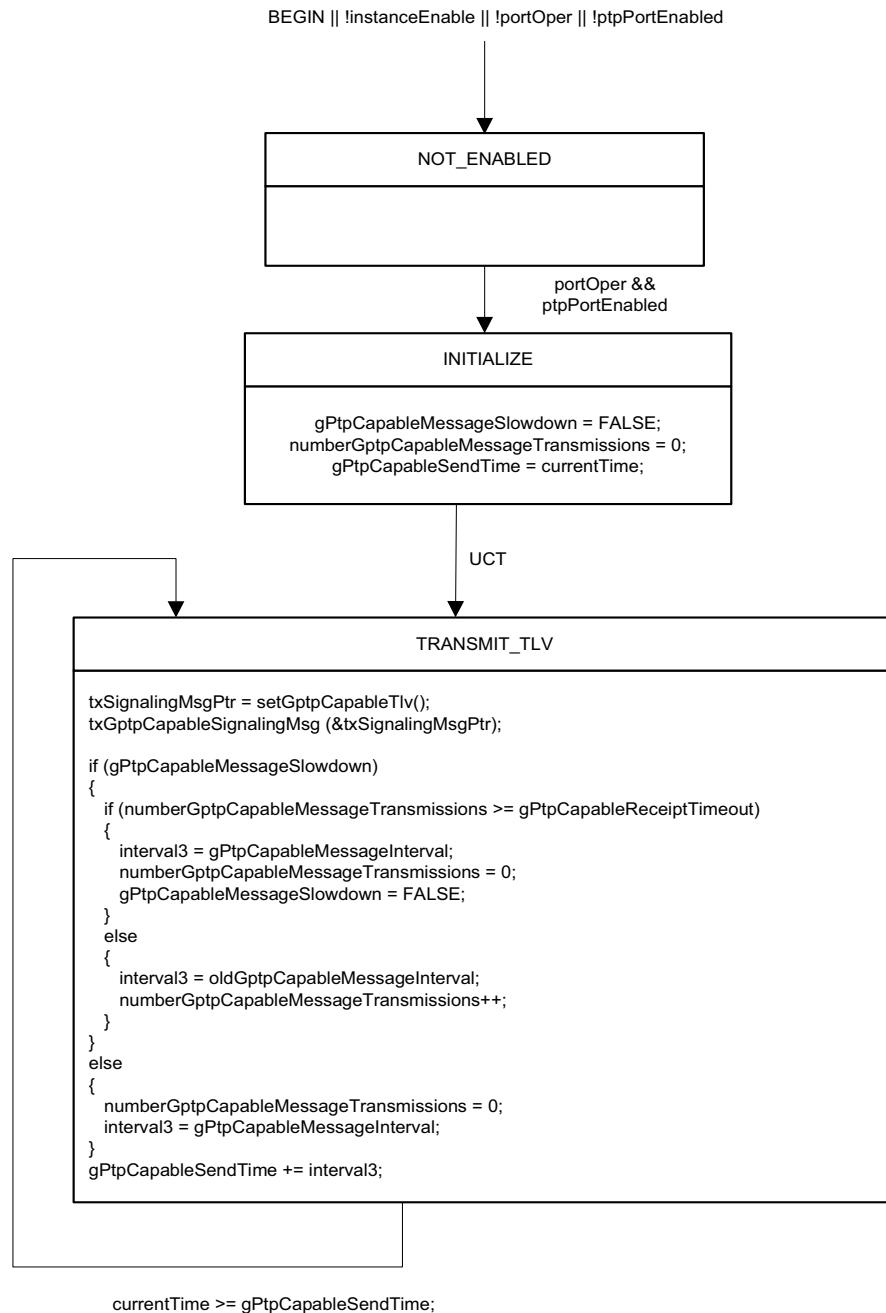


Figure 10-21—GptpCapableTransmit state machine

1 10.4.2 GptpCapableReceive state machine

2 10.4.2.1 State machine variables

3 The following variables are used in the state diagram in Figure 10-22 (in 10.4.2.2):

4 **10.4.2.1.1 rcvdGptpCapableTlv:** A Boolean variable that notifies the current state machine when a
5 Signaling message containing a gPTP-capable TLV is received. This variable is reset by the current state
6 machine.

7 **10.4.2.1.2 rcvdSignalingMsgPtr:** A pointer to a structure whose members contain the values of the fields of
8 a Signaling message whose receipt is indicated by rcvdGptpCapableTlv (see 10.4.2.1.1).

9 **10.4.2.1.3 gPtpCapableReceiptTimeoutTimeInterval:** The time interval after which, if a Signaling
10 message containing a gPTP-capable TLV is not received, the neighbor of this PTP Port is considered to no
11 longer be invoking gPTP. The data type for gPtpCapableReceiptTimeoutTimeInterval is UScaledNs.

12 **10.4.2.1.4 timeoutTime:** A variable used to save the time at which the neighbor of this PTP Port is
13 considered to no longer be invoking gPTP if a Signaling message containing a gPTP-capable TLV is not
14 received. The data type for timeoutTime is UScaledNs.

15 **10.4.2.1.5 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 10-22). The
16 data type for TEMP is Integer16.

17 10.4.2.2 State diagram

18 The GptpCapableReceive state machine shall implement the function specified by the state diagram in
19 Figure 10-22, the local variables specified in 10.4.2.1, the relevant parameters specified in 10.6.4 and
20 10.6.4.4, and the relevant timing attributes specified in 10.7. This state machine is responsible for setting
21 neighborGptpCapable to TRUE on receipt of a Signaling message containing the gPTP-capable TLV, and
22 setting the timeout time after which neighborGptpCapable is set to FALSE.

23 10.4.3 GptpCapableIntervalSetting state machine

24 10.4.3.1 State machine variables

25 The following variables are used in the state diagram in Figure 10-23 (in 10.4.3.3):

26 **10.4.3.1.1 rcvdSignalingMsg4:** A Boolean variable that notifies the current state machine when a Signaling
27 message that contains a gPTP-capable Message Interval Request TLV (see 10.6.4.5) is received. This
28 variable is reset by the current state machine.

29 **10.4.3.1.2 rcvdSignalingPtrGIS:** A pointer to a structure whose members contain the values of the fields of
30 the received Signaling message that contains a gPTP-capable Message Interval Request TLV (see 10.6.4.5).

31 **10.4.3.1.3 logSupportedGptpCapableMessageIntervalMax:** The maximum supported logarithm to base 2
32 of the gPTP-capable message interval. The data type for logSupportedGptpCapableMessageIntervalMax is
33 Integer8.

34 **10.4.3.1.4 logSupportedClosestLongerGptpCapableMessageInterval:** The logarithm to base 2 of the
35 gPTP-capable message interval, such that $\log_{\text{Supported}} \text{Closest} > \log_{\text{Requested}} \text{Closest}$,
36 that is numerically closest to
37 $\log_{\text{Requested}} \text{Closest}$, where $\log_{\text{Requested}} \text{Closest}$ is the

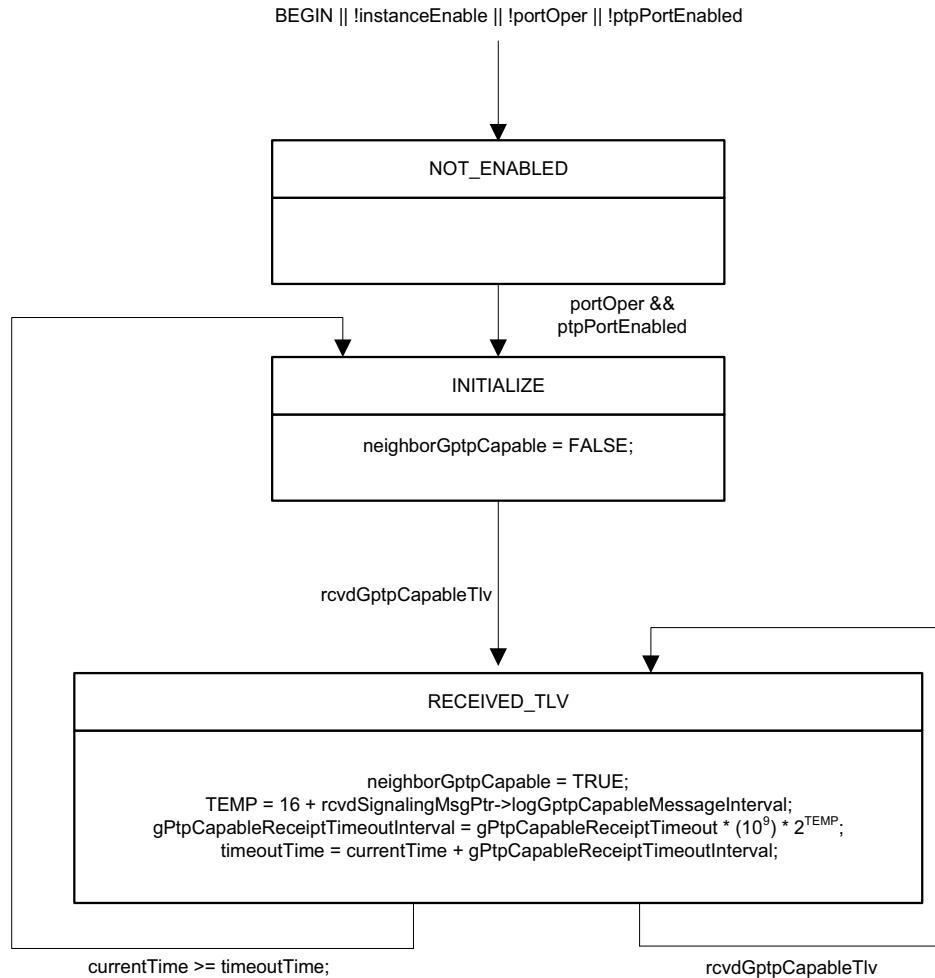


Figure 10-22—GtpCapableReceive state machine

1 argument of the function computeLogGtpCapableMessageInterval() (see 10.4.3.2.2). The data type for
 2 logSupportedClosestLongerGtpCapableMessageInterval is Integer8.

3 **10.4.3.1.5 computedLogGtpCapableMessageInterval:** A variable used to hold the result of the function
 4 computeLogGtpCapableMessageInterval(). The data type for computedLogGtpCapableMessageInterval
 5 is Integer8.

6 **10.4.3.1.6 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 10-23). The
 7 data type for TEMP is Integer16.

8 **10.4.3.2 State machine functions**

9 **10.4.3.2.1 isSupportedLogGtpCapableMessageInterval (logGtpCapableMessageInterval):** A
 10 Boolean function that returns TRUE if the gPTP-capable message interval given by the argument
 11 logGtpCapableMessageInterval is supported by the PTP Port and FALSE if the gPTP-capable message
 12 interval is not supported by the PTP Port. The argument logGtpCapableMessageInterval has the same data

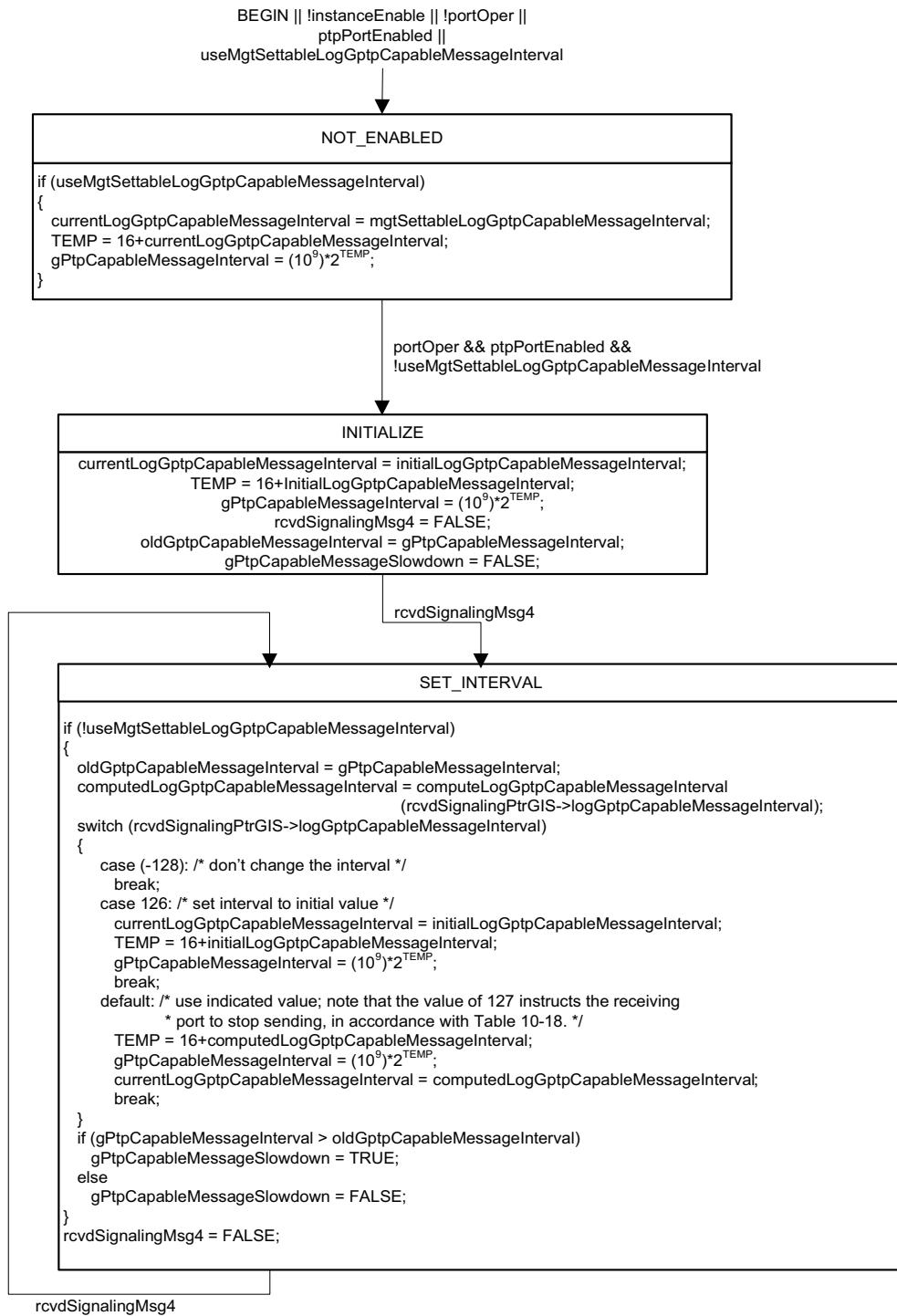
1 type and format as the field logGptpCapableMessageInterval of the gPTP-capable message interval request
2 TLV (see 10.6.4.5.6).

3 **10.4.3.2.2 computeLogGptpCapableMessageInterval (logRequestedGptpCapableMessageInterval):**
4 An Integer8 function that computes and returns the logGptpCapableMessageInterval, based on the
5 logRequestedGptpCapableMessageInterval. This function is defined as indicated below. It is defined here so
6 that the detailed code that it invokes does not need to be placed into the state machine diagram.

```
7 Integer8 computeLogGptpCapableMessageInterval (logRequestedGptpCapableMessageInterval)
8 Integer8 logRequestedGptpCapableMessageInterval;
9 {
10     Integer8 logSupportedGptpCapableMessageIntervalMax,
11         logSupportedClosestLongerGptpCapableMessageInterval;
12     if (isSupportedLogGptpCapableMessageInterval
13         (logRequestedGptpCapableMessageInterval))
14         // The requested gPTP-capable Message Interval is supported and returned
15         return (logRequestedGptpCapableMessageInterval)
16     else
17     {
18         if (logRequestedGptpCapableMessageInterval >
19             logSupportedGptpCapableMessageIntervalMax)
20             // Return the largest supported logGptpCapableMessageInterval, even if
21 smaller than the requested interval
22             return (logSupportedGptpCapableMessageIntervalMax);
23         else
24             // Return the smallest supported logGptpCapableMessageInterval that is
25 still larger than
26             // the requested interval.
27             return (logSupportedClosestLongerGptpCapableMessageInterval);
28     }
29 }
```

30 **10.4.3.3 State diagram**

31 The GptpCapableIntervalSetting state machine shall implement the function specified by the state diagram
32 in Figure 10-23, the local variables specified in 10.4.3.1, the functions specified in 10.4.3.2, the messages
33 specified in 10.6, the relevant global variables specified in 10.2.5, the relevant managed objects specified in
34 14.8, and the relevant timing attributes specified in 10.7. This state machine is responsible for setting the
35 global variables that give the duration of the mean intervals between successive Signaling messages
36 containing the gPTP-capable TLV, both at initialization and in response to the receipt of a Signaling message
37 that contains a gPTP-capable Message Interval Request TLV (see 10.6.4.5).

**Figure 10-23—GtpCapableIntervalSetting state machine**

1 10.5 Message attributes

2 10.5.1 General

3 This subclause describes media-independent attributes of the Announce message and the Signaling message
4 that are not described in 8.4.2 and whose descriptions are not generic to all messages used in this standard.
5 This subclause also describes media-independent attributes of all time-synchronization event messages.

6 10.5.2 Message class

7 The Announce message is a general message, i.e., it is not timestamped. An Announce message provides
8 status and characterization information of the PTP Instance that transmitted the message and the
9 Grandmaster PTP Instance. This information is used by the receiving PTP Instance when executing the
10 BTCA.

11 The Signaling message is a general message, i.e., it is not timestamped. A Signaling message carries
12 information, requests, and/or commands between PTP Instances, via one or more TLVs.

13 NOTE—In this standard, the Signaling message is used by a port of a PTP Instance to request that the port at the other
14 end of the link send time-synchronization event messages, link delay measurement messages, or Announce messages at
15 desired intervals; to indicate whether the port at the other end of the link should compute neighborRateRatio and/or
16 meanLinkDelay; and to indicate whether a PTP Port can receive and correctly process one-step Syncs. The message
17 interval request TLV is defined to carry this information (see 10.6.4.3). One usage of this functionality is to allow a time-
18 aware system in power-saving mode to remain connected to a gPTP domain via the port on which the Signaling message
19 is sent.

20 10.5.3 Addresses

21 The destination address of the Announce and Signaling messages shall be the reserved multicast address
22 given in Table 10-4 unless otherwise specified in a media-dependent clause (see 12.2 and 16.2).

Table 10-4—Destination address for Announce and Signaling messages

Destination address
01-80-C2-00-00-0E
NOTE—This address is taken from Table 8-1, Table 8-2, and Table 8-3 of IEEE Std 802.1Q-2018.

23 NOTE—Frames whose destination address is the address of Table 10-4 are never forwarded, according to IEEE 802.1Q
24 protocol. Use of this address is shared by IEEE 802.1AS and other IEEE 802.1 protocols.

25 If the transport is full-duplex IEEE 802.3, all Announce and Signaling messages shall use the MAC address
26 of the respective egress physical port as the source address.

27 10.5.4 EtherType

28 The EtherType of the Announce and Signaling messages shall be the EtherType given in Table 10-5.

29 NOTE—This EtherType is used for all PTP messages.

30 10.5.5 Subtype

31 The subtype for the Announce and Signaling messages is indicated by the majorSdoId field (see 10.6.2.2.1).

Table 10-5—EtherType for Announce and Signaling messages

EtherType
88-F7

1 10.5.6 Source port identity

2 The Announce message, Signaling message, and all time-synchronization messages contain a
3 sourcePortIdentity field (see 10.6.2.2.11), which identifies the egress port (see 8.5) on which the respective
4 message is sent.

5 10.5.7 Sequence number

6 Each PortSync entity of a PTP Instance maintains a separate sequenceId pool for each of the message types
7 Announce and Signaling, respectively, transmitted by the MD entity of the PTP Port.

8 Each Announce and Signaling message contains a sequenceId field (see 10.6.2.2.12), which carries the
9 message sequence number. The sequenceId of an Announce message shall be one greater than the
10 sequenceId of the previous Announce message sent by the transmitting PTP Port, subject to the constraints
11 of the rollover of the UIInteger16 data type used for the sequenceId field. The sequenceId of a Signaling
12 message shall be one greater than the sequenceId of the previous Signaling message sent by the transmitting
13 PTP Port, subject to the constraints of the rollover of the UIInteger16 data type used for the sequenceId field.

14 10.6 Message formats

15 10.6.1 General

16 The PTP messages Announce and Signaling each have a header, body, and, if present, a suffix that contains
17 one or more TLVs (see 10.6.2, 10.6.3, and 10.6.4 of this standard and Clause 14 of IEEE Std 1588-2019).
18 Reserved fields shall be transmitted with all bits of the field 0 and ignored by the receiver, unless otherwise
19 specified. The data type of the field shall be the type indicated in brackets in the title of each subclause.

20 Subclause 10.6 defines the path trace TLV, which is carried by the Announce message (see 10.6.3.2.8), and
21 the message interval request TLV, which is carried by the Signaling message (see 10.6.4.3).

22 PTP Management Messages are not used in this standard. They are specified in IEEE Std 1588-2019.

23 IEEE Std 1588-2019 specifies various optional features that have associated TLVs. These optional features,
24 including the associated TLVs, may be supported by an implementation of this standard. IEEE Std
25 1588-2019 also specifies that certain TLVs are propagated by a Boundary Clock if they are attached to an
26 Announce message and are not supported (see 14.2.2.2 and Table 52 of IEEE Std 1588-2019). These TLVs
27 are listed in Table 10-6. The TLV Propagate requirement in IEEE Std 1588-2019 means that a Propagate
28 TLV is propagated through a PTP Relay Instance (e.g., from an ingress PTP Port in the TimeReceiver state
29 to an egress PTP Port in the TimeTransmitter state, even when the TLV is unsupported by the PTP Relay
30 Instance). If the corresponding optional feature is not supported by the PTP Relay Instance, the PTP Relay
31 Instance shall propagate the TLV unchanged.

32 If a PTP Instance cannot parse a non-forwarding TLV, it shall ignore it and attempt to parse the next TLV
33 (see 14.1 of IEEE Std 1588-2019).

34 NOTE—Any overhead specific to the respective medium is added to each message.

Table 10-6—Propagate TLVs of IEEE Std 1588-2019

tlvType values	Value (hex)	TLV defined in
PATH_TRACE	0008	16.2 of IEEE Std 1588-2019, and required by 10.6.3.3 of this standard
ALTERNATE_TIME_OFFSET_INDICATOR	0009	16.3 of IEEE Std 1588-2019
ORGANIZATION_EXTENSION_PROPAGATE	4000	14.3 of IEEE Std 1588-2019
ENHANCED_ACCURACY_METRICS	4001	16.12 of IEEE Std 1588-2019
Reserved for assignment by the IEEE 1588 Working Group for TLVs that propagate	4002–7EFF	
Experimental values (see 4.2.9 of IEEE Std 1588-2019)	7FF0–7FFF	

1 10.6.2 Header

2 10.6.2.1 General header specifications

3 The common header for all PTP messages shall be as specified in Table 10-7 and 10.6.2.2.

Table 10-7—PTP message header

Bits	Octets	Offset
7 6 5 4 3 2 1 0		
majorSdId	1	0
minorVersionPTP	1	1
messageLength	2	2
domainNumber	1	4
minorSdId	1	5
flags	2	6
correctionField	8	8
messageTypeSpecific	4	16
sourcePortIdentity	10	20
sequenceId	2	30
controlField	1	32
logMessageInterval	1	33

1 10.6.2.2 Header field specifications

2 10.6.2.2.1 majorSdold (Nibble)

3 The value is specified in 8.1 for all transmitted PTP messages of a gPTP domain. The value is specified in
 4 11.2.17 for all transmitted PTP messages of the Common Mean Link Delay Service. Any PTP message
 5 received for which the value is not one of the values specified in those subclauses shall be ignored.

6 10.6.2.2.2 messageType (Enumeration4)

7 The value indicates the type of the message, as defined in Table 10-8.

8 The most significant bit of the messageType field divides this field in half between event and general
 9 messages, i.e., it is 0 for event messages and 1 for general messages.

Table 10-8—Values for messageType field

Message type	Message class	Value
Announce	General	0xB
Signaling	General	0xC
NOTE—Values for the messageType field for other PTP messages that are used only for specific media are defined in the respective media-dependent clause(s).		

10 10.6.2.2.3 minorVersionPTP (UInteger4)

11 For transmitted messages, the value shall be 1 (see 7.5.4 and 13.3.2.5 of IEEE Std 1588-2019). For received
 12 messages, the value is ignored.

13 NOTE—minorVersionPTP indicates the minor version number of IEEE 1588 PTP used in the PTP profile contained in
 14 this standard for information only.

15 10.6.2.2.4 versionPTP (UInteger4)

16 For transmitted messages, the value shall be 2 (see 7.5.4 and 13.3.2.4 of IEEE Std 1588-2019). For received
 17 messages, if the value is not 2, the entire message shall be ignored.

18 NOTE—versionPTP indicates the version number of IEEE 1588 PTP used in the PTP profile contained in this standard.

19 10.6.2.2.5 messageLength (UInteger16)

20 The value is the total number of octets that form the PTP message. The counted octets start with, and
 21 include, the first octet of the header and terminate with, and include, the last octet of the last TLV or, if there
 22 are no TLVs, the last octet of the message as defined in this clause.

23 NOTE—For example, the Follow_Up message (see 11.4.4) contains a PTP header (34 octets), preciseOriginTimestamp
 24 (10 octets), and Follow_Up information TLV (32 octets). The value of the messageLength field is $34+10+32 = 76$.

25 10.6.2.2.6 domainNumber (UInteger8)

26 The value is the gPTP **domainNumber** specified in 8.1.

1 10.6.2.2.7 minorSdold (UInteger8)

2 The value is specified in 8.1 for all transmitted PTP messages of a gPTP domain. The value is specified in
 3 11.2.17 for all transmitted PTP messages of the Common Mean Link Delay Service. Any PTP message
 4 received for which the value is not one of the values specified in those subclauses shall be ignored.

5 10.6.2.2.8 flags (Octet2)

6 The value of the bits of the array are defined in Table 10-9. For message types where the bit is not defined in
 7 Table 10-9, the value of the bit is set to FALSE.

Table 10-9—Values of flag bits

Octet	Bit	Message types	Name	Value
0	0	All	alternateTimeTransmitterFlag in Announce, Sync, Follow_Up, and Delay_Resp messages	Not used in this standard; transmitted as FALSE and ignored on reception
0	1	Sync, Pdelay_Resp	twoStepFlag	<p><i>For Sync messages:</i></p> <ul style="list-style-type: none"> a) For a one-step transmitting PTP Port (see 11.1.3 and 11.2.13.9), the value is FALSE. b) For a two-step transmitting PTP Port, the value is TRUE. <p><i>For Pdelay_Resp messages:</i></p> <p>The value is transmitted as TRUE and ignored on reception.</p>
0	2	All	unicastFlag	Not used in this standard; transmitted as FALSE and ignored on reception
0	3	All	Reserved	Not used by IEEE Std 1588-2019; reserved as FALSE and ignored on reception
0	4	All	Reserved	Not used by IEEE Std 1588-2019; reserved as FALSE and ignored on reception
0	5	All	PTP profileSpecific 1	Not used in this standard; transmitted as FALSE and ignored on reception
0	6	All	PTP profileSpecific 2	Not used in this standard; transmitted as FALSE and ignored on reception
0	7	All	Reserved	Not used in this standard; transmitted as FALSE and ignored on reception
1	0	Announce	leap61	The value of the global variable leap61 (see 10.3.9.4)
1	1	Announce	leap59	The value of the global variable leap59 (see 10.3.9.5)
1	2	Announce	currentUtcOffsetValid	For domain 0, transmitted as TRUE and ignored on receipt. For domains other than domain 0, the value of the global variable currentUtcOffsetValid (see 10.3.9.6)

Table 10-9—Values of flag bits (*continued*)

Octet	Bit	Message types	Name	Value
1	3	Announce	ptpTimescale	The value of the global variable ptpTimescale (see 10.3.9.7)
1	4	Announce	timeTraceable	The value of the global variable timeTraceable (see 10.3.9.8)
1	5	Announce	frequencyTraceable	The value of the global variable frequencyTraceable (see 10.3.9.9)
1	6	All	Reserved	Not used by IEEE Std 1588-2019; reserved as FALSE and ignored on reception
1	7	All	Reserved	Not used in this standard; reserved as FALSE and ignored on reception

1 10.6.2.2.9 correctionField (Integer64)

2 The value is 0.

3 10.6.2.2.10 messageTypeSpecific (Octet4)

4 The value of the messageTypeSpecific field varies, based on the value of the messageType field, as described in Table 10-10.

6 For Event messages only, the four octets of the messageTypeSpecific field may be used for internal implementation of a PTP Instance and its ports. For example, if the clock consists of multiple hardware components that are not synchronized, messageTypeSpecific can be used to transfer an internal timestamp between components (e.g., a physical layer chip and the clock's processor).

10 The messageTypeSpecific field is not used for features of this standard, and it has no meaning from one clock to another. In the on-the-wire format at each PTP Port, for all messageType values, the messageTypeSpecific field is transmitted with all bits of the field 0 and ignored on receive.

Table 10-10—messageTypeSpecific semantics

Value of messageType	Description
Follow_Up, Pdelay_Resp_Follow_Up, Announce, Signaling, Management	For the General message class, this field is reserved; it is transmitted as 0 and ignored on reception.
Sync, Pdelay_Req, Pdelay_Resp	For the Event message class, this field may be used for internal implementation as specified in this subclause.

13 10.6.2.2.11 sourcePortIdentity (PortIdentity)

14 The value is the PTP Port identity attribute (see 8.5.2) of the PTP Port that transmits the PTP message.

15 10.6.2.2.12 sequenceId (UInteger16)

16 The sequenceId field is assigned as specified in 10.5.7.

1 10.6.2.2.13 controlField (UInteger8)

2 The value is 0.

3 10.6.2.2.14 logMessageInterval (Integer8)

4 For an Announce message, the value is the value of currentLogAnnounceInterval (see 10.3.10.6) for the PTP Port that transmits the Announce message. For a Signaling message, the value is transmitted as 0x7F and ignored on reception.

7 10.6.3 Announce message

8 10.6.3.1 General Announce message specifications

9 The fields of the body of the Announce message shall be as specified in Table 10-11 and 10.6.3.2.

Table 10-11—Announce message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 10.6.2)								34	0
reserved								10	34
currentUtcOffset								2	44
reserved								1	46
grandmasterPriority1								1	47
grandmasterClockQuality								4	48
grandmasterPriority2								1	52
grandmasterIdentity								8	53
stepsRemoved								2	61
timeSource								1	63
path trace TLV								4+8N	64

10 10.6.3.2 Announce message field specifications

11 10.6.3.2.1 currentUtcOffset (Integer16)

12 The value is the value of currentUtcOffset (see 10.3.9.10) for the PTP Instance that transmits the Announce message.

14 10.6.3.2.2 grandmasterPriority1 (UInteger8)

15 The value is the value of the priority1 component of the rootSystemIdentity of the gmPriorityVector (see 10.3.5) of the PTP Instance that transmits the Announce message.

1 10.6.3.2.3 grandmasterClockQuality (ClockQuality)

2 The value is the clockQuality formed by the clockClass, clockAccuracy, and offsetScaledLogVariance of the
 3 rootSystemIdentity of the gmPriorityVector (see 10.3.5) of the PTP Instance that transmits the Announce
 4 message.

5 10.6.3.2.4 grandmasterPriority2 (UInteger8)

6 The value is the value of the priority2 component of the rootSystemIdentity of the gmPriorityVector
 7 (see 10.3.5) of the PTP Instance that transmits the Announce message.

8 10.6.3.2.5 grandmasterIdentity (ClockIdentity)

9 The value is the value of the clockIdentity component of the rootSystemIdentity of the gmPriorityVector
 10 (see 10.3.5) of the PTP Instance that transmits the Announce message.

11 10.6.3.2.6 stepsRemoved (UInteger16)

12 The value is the value of timeTransmitterStepsRemoved (see 10.3.9.3) for the PTP Instance that transmits
 13 the Announce message.

14 10.6.3.2.7 timeSource (TimeSource)

15 The value is the value of timeSource (see 8.6.2.7 and 10.3.9.11) for the PTP Instance that transmits the
 16 Announce message.

17 10.6.3.2.8 Path trace TLV

18 The Announce message carries the path trace TLV, defined in 10.6.3.3.

19 10.6.3.3 Path trace TLV definition

20 10.6.3.3.1 General

21 The fields of the path-trace TLV shall be as specified in Table 10-12 and in 10.6.4.3.2 through 10.6.4.3.9.
 22 This TLV and its use are defined in IEEE Std 1588-2019 (see 16.2 and Table 52 of IEEE Std 1588-2019).

Table 10-12—Path trace TLV

Bits	Octets	Offset from start of TLV
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
pathSequence	8N	4

1 10.6.3.3.2 tlvType (Enumeration16)

2 The value of the tlvType field is 0x8.

3 NOTE—This value indicates the TLV is a path trace TLV, as specified in 16.2.5.1 and Table 52 of IEEE Std 1588-2019.
 4 The value 0x8 is specified in that standard as PATH_TRACE.

5 10.6.3.3 lengthField (UInteger16)

6 The value of the lengthField is 8N.

7 10.6.3.3.4 pathSequence (ClockIdentity[N])

8 The value of pathSequence is a ClockIdentity array. The array elements are the clockIdentities of the
 9 successive PTP Instances that receive and send an Announce message. The quantity N is the number of PTP
 10 Instances, including the Grandmaster PTP Instance, that the Announce information has traversed.

11 NOTE—N is equal to stepsRemoved+1 (see 10.6.3.2.6). The size of the pathSequence array increases by 1 for each PTP
 12 Instance that the Announce information traverses.

13 10.6.4 Signaling message

14 10.6.4.1 General Signaling message specifications

15 The fields of the body of the Signaling message shall be as specified in Table 10-13 and 10.6.4.2.

Table 10-13—Signaling message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
								34	0
								10	34
								N	44

header (see 10.6.2)
 targetPortIdentity
 one or more TLVs

16 10.6.4.2 Signaling message field specifications

17 10.6.4.2.1 targetPortIdentity (PortIdentity)

18 The value of targetPortIdentity.clock identity is all ones, i.e., 0xFFFFFFFFFFFFFF. The value of
 19 targetPortIdentity.portNumber is all ones, i.e., 0xFFFF.

20 10.6.4.2.2 TLVs carried in one Signaling message

21 The Signaling message carries either:

- 22 a) One or both interval request TLVs, defined in 10.6.4.3 and 10.6.4.5, or
- 23 b) The gPTP-capable TLV, defined in 10.6.4.4.

24

1

2 **10.6.4.3 Message interval request TLV definition**

3 **10.6.4.3.1 General**

4 The fields of the message interval request TLV shall be as specified in Table 10-14 and in 10.6.4.3.2 through
 5 10.6.4.3.9. This TLV is a standard organization extension TLV for the Signaling message, as specified in
 6 14.3 of IEEE Std 1588-2019.

Table 10-14—Message interval request TLV

Bits	Octets	Offset from start of TLV
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
organizationId	3	4
organizationSubType	3	7
logLinkDelayInterval	1	10
logTimeSyncInterval	1	11
logAnnounceInterval	1	12
flags	1	13
reserved	2	14

7 **10.6.4.3.2 tlvType (Enumeration16)**

8 The value of the tlvType field is 0x3.

9 NOTE—This value indicates the TLV is a vendor and standard organization extension TLV, as specified in 14.3.2.1 and
 10 Table 52 of IEEE Std 1588-2019. The tlvType is specified in that standard as ORGANIZATION_EXTENSION with a
 11 value of 0x3.

12 **10.6.4.3.3 lengthField (UInteger16)**

13 The value of the lengthField is 12.

14 **10.6.4.3.4 organizationId (Octet3)**

15 The value of organizationId is 00-80-C2.

16 **10.6.4.3.5 organizationSubType (Enumeration24)**

17 The value of organizationSubType is 2.

1 10.6.4.3.6 logLinkDelayInterval (Integer8)

2 The value is the logarithm to base 2 of the mean time interval, desired by the port that sends this TLV,
 3 between successive Pdelay_Req messages sent by the port at the other end of the link. The format and
 4 allowed values of logLinkDelayInterval are the same as the format and allowed values of
 5 initialLogPdelayReqInterval (see 11.5.2.2).

6 The values 127, 126, and -128 are interpreted as defined in Table 10-15.

Table 10-15—Interpretation of special values of logLinkDelayInterval

Value	Instruction to PTP Instance that receives this TLV
127	Instructs the port that receives this TLV to stop sending link delay measurement messages.
126	Instructs the port that receives this TLV to set currentLogPdelayReqInterval to the value of initialLogPdelayReqInterval (see 11.5.2.2).
-128	Instructs the port that receives this TLV not to change the mean time interval between successive Pdelay_Req messages.
All values in the ranges [-127, -25] and [25, 125] are reserved.	

7 10.6.4.3.7 logTimeSyncInterval (Integer8)

8 The value is the logarithm to base 2 of the mean time interval, desired by the PTP Port that sends this TLV,
 9 between successive time-synchronization event messages sent by the PTP Port at the other end of the link.
 10 The format and allowed values of logTimeSyncInterval are the same as the format and allowed values of
 11 initialLogSyncInterval (see 10.7.2.3, 11.5.2.3, 12.8, and 13.9.2).

12 The values 127, 126, and -128 are interpreted as defined in Table 10-16.

Table 10-16—Interpretation of special values of logTimeSyncInterval

Value	Instruction to PTP Instance that receives this TLV
127	Instructs the PTP Port that receives this TLV to stop sending time-synchronization event messages.
126	Instructs the PTP Port that receives this TLV to set currentLogSyncInterval to the value of initialLogSyncInterval (see 10.7.2.3, 11.5.2.3, 12.8, and 13.9.2).
-128	Instructs the PTP Port that receives this TLV not to change the mean time interval between successive time-synchronization event messages.
All values in the ranges [-127, -25] and [25, 125] are reserved.	

13 When a Signaling message that contains this TLV is sent by a PTP Port, the value of syncReceiptTimeoutTimeInterval for that PTP Port (see 10.2.5.3) shall be set equal to syncReceiptTimeout (see 10.7.3.1) multiplied by the value of the interval, in seconds, reflected by logTimeSyncInterval.

1 10.6.4.3.8 logAnnounceInterval (Integer8)

2 The value is the logarithm to base 2 of the mean time interval, desired by the PTP Port that sends this TLV,
 3 between successive Announce messages sent by the PTP Port at the other end of the link. The format and
 4 allowed values of logAnnounceInterval are the same as the format and allowed values of
 5 initialLogAnnounceInterval (see 10.7.2.2).

6 The values 127, 126, and -128 are interpreted as defined in Table 10-17.

Table 10-17—Interpretation of special values of logAnnounceInterval

Value	Instruction to PTP Instance that receives this TLV
127	Instructs the PTP Port that receives this TLV to stop sending Announce messages.
126	Instructs the PTP Port that receives this TLV to set currentLogAnnounceInterval to the value of initialLogAnnounceInterval (see 10.7.2.2).
-128	Instructs the PTP Port that receives this TLV not to change the mean time interval between successive Announce messages.
All values in the ranges [-127, -25] and [25, 125] are reserved.	

7 When a Signaling message that contains this TLV is sent by a PTP Port, the value of
 8 announceReceiptTimeoutTimeInterval for that PTP Port (see 10.3.10.1) shall be set equal to
 9 announceReceiptTimeout (see 10.7.3.2) multiplied by the value of the interval, in seconds, reflected by
 10 logAnnounceInterval.

11 10.6.4.3.9 flags (Octet)

12 Bits 0 through 2 of the octet are defined in Table 10-18 and take on the values TRUE and FALSE. Bits not
 13 defined in Table 10-18 are set to FALSE and ignored on receipt.

**Table 10-18—Definitions of bits of flags field
 of message interval request TLV**

Bit	Name
0	computeNeighborRateRatio
1	computeMeanLinkDelay
2	oneStepReceiveCapable

14 NOTE—For full-duplex point-to-point links (see Clause 11), it is expected that the PTP Port sending this TLV will set
 15 bits 0 and/or 1 to FALSE if this PTP Port will not provide valid timing information in its subsequent responses
 16 (Pdelay_Resp and Pdelay_Resp_Follow_Up) to Pdelay_Req messages. Similarly, it is expected that the PTP Port
 17 sending this TLV will set bit 2 to TRUE if it is capable of receiving and correctly processing one-step Sync messages.

¹ **10.6.4.4 gPTP-capable TLV definition**

² **10.6.4.4.1 General**

³ The fields of the gPTP-capable TLV shall be as specified in Table 10-19 and in 10.6.4.4.2 through ⁴ 10.6.4.4.7. This TLV is a standard organization extension TLV for the Signaling message, as specified in ⁵ 14.3 of IEEE Std 1588-2019.

Table 10-19—gPTP-capable TLV

Bits	Octets	Offset from start of TLV
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
organizationId	3	4
organizationSubType	3	7
logGptpCapableMessageInterval	1	10
flags	1	11
reserved	4	12

⁶ **10.6.4.4.2 tlvType (Enumeration16)**

⁷ The value of the tlvType field is 0x8000.

⁸ NOTE—This value indicates the TLV is a vendor and standard organization extension TLV that is not propagated, as ⁹ specified in 14.3.2.1 and Table 52 of IEEE Std 1588-2019. The tlvType is specified in that standard as ¹⁰ ORGANIZATION_EXTENSION_DO_NOT_PROPAGATE with a value of 0x8000.

¹¹ **10.6.4.4.3 lengthField (UInteger16)**

¹² The value of the lengthField is 12.

¹³ **10.6.4.4 organizationId (Octet3)**

¹⁴ The value of organizationId is 00-80-C2.

¹⁵ **10.6.4.4.5 organizationSubType (Enumeration24)**

¹⁶ The value of organizationSubType is 4.

¹⁷ **10.6.4.4.6 logGptpCapableMessageInterval (Integer8)**

¹⁸ The value of logGptpCapableMessageInterval is the logarithm to base 2 of the mean gPTP-capable message ¹⁹ interval in seconds (see 10.7.2.1 and 10.7.2.5)

1 10.6.4.4.7 flags (Octet)

2 The flag bits shall be transmitted as FALSE and ignored on receipt.

3 10.6.4.5 gPTP-capable message interval request TLV definition

4 10.6.4.5.1 General

5 The fields of the gPTP-capable message interval request TLV shall be as specified in Table 10-20 and in
 6 10.6.4.5.2 through 10.6.4.5.6. This TLV is a standard organization extension TLV for the Signaling message,
 7 as specified in 14.3 of IEEE Std 1588-2019.

Table 10-20—gPTP-capable message interval request TLV

Bits	Octets	Offset from start of TLV
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
organizationId	3	4
organizationSubType	3	7
logGptpCapableMessageInterval	1	10
reserved	3	11

8 10.6.4.5.2 tlvType (Enumeration16)

9 The value of the tlvType field is 0x8000.

10 NOTE—This value indicates the TLV is a vendor and standard organization extension TLV that is not propagated, as
 11 specified in 14.3.2.1 and Table 52 of IEEE Std 1588-2019. The tlvType is specified in that standard as
 12 ORGANIZATION_EXTENSION_DO_NOT_PROPAGATE with a value of 0x8000.

13 10.6.4.5.3 lengthField (UInteger16)

14 The value of the lengthField is 10.

15 10.6.4.5.4 organizationId (Octet3)

16 The value of organizationId is 00-80-C2.

17 10.6.4.5.5 organizationSubType (Enumeration24)

18 The value of organizationSubType is 5.

1 10.6.4.5.6 logGptpCapableMessageInterval (Integer8)

2 The value is the logarithm to base 2 of the mean time interval, desired by the PTP Port that sends this TLV,
 3 between successive Signaling messages that contain the gPTP-capable TLV (see 10.6.4.4), sent by the PTP
 4 Port at the other end of the link. The format and allowed values of logGptpCapableMessageInterval are the
 5 same as the format and allowed values of initialLogGptpCapableMessageInterval (see 10.7.2.5).

6 The values 127, 126, and -128 are interpreted as defined in Table 10-21.

Table 10-21—Interpretation of special values of logGptpCapableMessageInterval

Value	Instruction to PTP Instance that receives this TLV
127	Instructs the PTP Port that receives this TLV to stop sending Signaling messages that contain the gPTP-capable TLV.
126	Instructs the PTP Port that receives this TLV to set currentLogGptpCapableMessageInterval to the value of initialLogGptpCapableMessageInterval (see 10.7.2.4).
-128	Instructs the PTP Port that receives this TLV not to change the mean time interval between successive Signaling messages that contain the gPTP-capable TLV.
All values in the ranges [-127, -25] and [25, 125] are reserved.	

7 When a Signaling message that contains this TLV is sent by a PTP Port, the value of
 8 gPtpCapableReceiptTimeoutTimeInterval for that PTP Port (see 10.3.10.1) shall be set equal to
 9 gPtpCapableReceiptTimeout (see 10.7.3.3) multiplied by the value of the interval, in seconds, reflected by
 10 logGptpCapableMessageInterval.

11 10.7 Protocol timing characterization

12 10.7.1 General

13 This subclause specifies timing and timeout attributes for the media-independent sublayer state machines.

14 10.7.2 Message transmission intervals

15 10.7.2.1 General interval specification

16 The mean time interval between the sending of successive Announce messages, known as the *announce*
 17 *interval*, shall be as specified in 10.7.2.2.

18 The mean time interval between the sending of successive time-synchronization event messages for full-
 19 duplex point-to-point, IEEE 802.11, and CSN links, and successive general messages containing time-
 20 synchronization information for IEEE 802.3 EPON links, is known as the *sync interval*. The sync interval
 21 shall be as specified in 10.7.2.3.

22 The mean time interval between the sending of successive Signaling messages that contain the gPTP-
 23 capable TLV, known as the *gPTP-capable message interval*, shall be as specified in 10.7.2.5.

1 10.7.2.2 Announce message transmission interval

2 The logarithm to the base 2 of the announce interval (in seconds) is carried in the logMessageInterval field
 3 of the Announce message.

4 When useMgtSettableLogAnnounceInterval (see 14.8.14) is FALSE, the initialLogAnnounceInterval
 5 specifies the announce interval when the PTP Port is initialized and the value to which the announce interval
 6 is set when a message interval request TLV is received with the logAnnounceInterval field set to 126 (see
 7 the AnnounceIntervalSetting state machine in 10.3.17). The currentLogAnnounceInterval specifies the
 8 current value of the announce interval. The default value of initialLogAnnounceInterval is 0. Every PTP
 9 Port supports the value 127; the PTP Port does not send Announce messages when
 10 currentLogAnnounceInterval has this value (see 10.3.17). A PTP Port may support other values, except for
 11 the reserved values indicated in Table 10-17. A PTP Port ignores requests for unsupported values (see
 12 10.3.17). The initialLogAnnounceInterval and currentLogAnnounceInterval are per-PTP Port attributes.

13 When useMgtSettableLogAnnounceInterval is TRUE, currentLogAnnounceInterval is set equal to
 14 mgtSettableLogAnnounceInterval (see 14.8.15), and initialLogAnnounceInterval is ignored.

15 Announce messages shall be transmitted such that the value of the arithmetic mean of the intervals, in
 16 seconds, between message transmissions is within $\pm 30\%$ of $2^{\text{currentLogAnnounceInterval}}$. In addition, a PTP
 17 Port shall transmit Announce messages such that at least 90% of the inter-message intervals are within
 18 $\pm 30\%$ of the value of $2^{\text{currentLogAnnounceInterval}}$. The interval between successive Announce messages should
 19 not exceed twice the value of $2^{\text{currentLogAnnounceInterval}}$ in order to prevent causing an
 20 announceReceiptTimeout event. The PortAnnounceTransmit state machine (see 10.3.16) is consistent with
 21 these requirements, i.e., the requirements here and the requirements of the PortAnnounceTransmit state
 22 machine can be met simultaneously.

23 NOTE 1—A minimum number of inter-message intervals is necessary in order to verify that a PTP Port meets these
 24 requirements. The arithmetic mean is the sum of the inter-message interval samples divided by the number of samples.
 25 For more detailed discussion of statistical analyses, see Papoulis [B25].

26 NOTE 2—if useMgtSettableLogAnnounceInterval is FALSE, the value of initialLogAnnounceInterval is the value of
 27 the mean time interval between successive Announce messages when the PTP Port is initialized. The value of the mean
 28 time interval between successive Announce messages can be changed, e.g., if the PTP Port receives a Signaling message
 29 that carries a message interval request TLV (see 10.6.4.3) and the current value is stored in currentLogAnnounceInterval.
 30 The value of the mean time interval between successive Announce messages can be reset to the initial value, e.g., by a
 31 message interval request TLV for which the value of the field logAnnounceInterval is 126 (see 10.6.4.3.8).

32 NOTE 3—A PTP Port that requests (using a Signaling message that contains a message interval request TLV; see 10.6.4
 33 and 10.3.17) that the PTP Port at the other end of the attached link set its currentLogAnnounceInterval to a specific value
 34 can determine if the request was honored by examining the logMessageInterval field of subsequent received Announce
 35 messages.

36 10.7.2.3 Time-synchronization event message transmission interval

37 The logarithm to the base 2 of the sync interval (in seconds) is carried in the logMessageInterval field of the
 38 time-synchronization messages.

39 When useMgtSettableLogSyncInterval (see 14.8.19) is FALSE, the initialLogSyncInterval specifies the
 40 sync interval when the PTP Port is initialized and the value to which the sync interval is set when a message
 41 interval request TLV is received with the logTimeSyncInterval field set to 126 (see the SyncIntervalSetting
 42 state machine in 10.3.18). The default value is media-dependent; the value is specified in the respective
 43 media-dependent clauses. The initialLogSyncInterval is a per-PTP Port attribute.

44 The currentLogSyncInterval specifies the current value of the sync interval and is a per-PTP Port attribute.

1 When `useMgtSettableLogSyncInterval` is TRUE, `currentLogSyncInterval` is set equal to
 2 `mgtSettableLogSyncInterval` (see 14.8.20), and `initialLogSyncInterval` is ignored.

3 When the value of `syncLocked` is FALSE, time-synchronization messages shall be transmitted such that the
 4 value of the arithmetic mean of the intervals, in seconds, between message transmissions is within $\pm 30\%$ of
 5 $2^{currentLogSyncInterval}$. In addition, a PTP Port shall transmit time-synchronization messages such that at least
 6 90% of the inter-message intervals are within $\pm 30\%$ of the value of $2^{currentLogSyncInterval}$. The interval
 7 between successive time-synchronization messages should not exceed twice the value of
 8 $2^{currentLogSyncInterval}$ in order to prevent causing a `syncReceiptTimeout` event. The `PortSyncSyncSend` state
 9 machine (see 10.2.12) is consistent with these requirements, i.e., the requirements here and the requirements
 10 of the `PortSyncSyncSend` state machine can be met simultaneously.

11 NOTE 1—A minimum number of inter-message intervals is necessary in order to verify that a PTP Port meets these
 12 requirements. The arithmetic mean is the sum of the inter-message interval samples divided by the number of samples.
 13 For more detailed discussion of statistical analyses, see Papoulis [B25].

14 NOTE 2—If `useMgtSettableLogSyncInterval` is FALSE the value of `initialLogSyncInterval` is the value of the sync
 15 interval when the PTP Port is initialized. The value of the sync interval can be changed, e.g., if the PTP Port receives a
 16 Signaling message that carries a message interval request TLV (see 10.6.4.3) and the current value is stored in
 17 `currentLogSyncInterval`. The value of the sync interval can be reset to the initial value, e.g., by a message interval
 18 request TLV for which the value of the field `logTimeSyncInterval` is 126 (see 10.6.4.3.7).

19 **10.7.2.4 Interval for providing synchronization information by ClockTimeTransmitter entity**

20 The `clockTimeTransmitterLogSyncInterval` specifies the mean time interval between successive instants at
 21 which the `ClockTimeTransmitter` entity provides time-synchronization information to the `SiteSync` entity.
 22 The value is less than or equal to the smallest `currentLogSyncInterval` (see 10.7.2.3) value for all the ports of
 23 the PTP Instance. The `clockTimeTransmitterLogSyncInterval` is an internal, per PTP Instance variable.

24 **10.7.2.5 Interval for sending the gPTP-capable TLV Signaling message**

25 The logarithm to the base 2 of the gPTP-capable message interval (in seconds) is carried in the
 26 `logGptpCapableMessageInterval` field of the gPTP-capable TLV. The default value shall be 0. The range
 27 shall be -24 through 24. Other values in the range [-128, 127] shall be reserved.

28 When `useMgtSettableLogGptpCapableMessageInterval` (see 14.8.14) is FALSE, the
 29 `initialLogGptpCapableMessageInterval` specifies the following:

30 a) The mean gPTP-capable message interval when the PTP Port is initialized, and
 31 b) The value to which the gPTP-capable message interval is set when a gPTP-capable TLV is received
 32 with the `logGptpCapableMessageInterval` field set to 126 (see the
 33 `GptpCapableMessageIntervalSetting` state machine in 10.4.3).

34 The `currentLogGptpCapableMessageInterval` specifies the current value of the gPTP-capable message
 35 interval. Every PTP Port supports the value 127; the PTP Port does not send Signaling messages containing
 36 a gPTP-capable TLV when `currentLogGptpCapableMessageInterval` has this value (see 10.4.3). A PTP Port
 37 may support other values, except for the reserved values indicated in Table 10-21. A PTP Port shall ignore
 38 requests for unsupported values (see 10.4.3). The `initialLogGptpCapableMessageInterval` and
 39 `currentLogGptpCapableMessageInterval` are per-PTP Port attributes.

40 When `useMgtSettableLogGptpCapableMessageInterval` is TRUE, `currentLogGptpCapableMessageInterval`
 41 is set equal to `mgtSettableLogGptpCapableMessageInterval` (see 14.8.15), and
 42 `initialLogGptpCapableMessageInterval` is ignored.

1 NOTE 1—If `useMgtSettableLogGptpCapableMessageInterval` is FALSE, the value of
 2 `initialLogGptpCapableMessageInterval` is the value of the mean time interval between successive Signaling messages
 3 containing the gPTP-capable TLV when the PTP Port is initialized. The value of the mean time interval between
 4 successive Signaling messages containing the gPTP-capable TLV can be changed, e.g., if the PTP Port receives a
 5 Signaling message that carries a gPTP-capable TLV (see 10.6.4.4) and the current value is stored in
 6 `currentLogGptpCapableMessageInterval`. The value of the mean time interval between successive Signaling messages
 7 containing the gPTP-capable TLV can be reset to the initial value, e.g., by a Signaling message containing a gPTP-
 8 capable message interval request TLV for which the value of the field `logGptpCapableMessageInterval` is 126
 9 (see 10.6.4.5.6).

10 NOTE 2—A PTP Port that requests (using a Signaling message that contains a gPTP-capable message interval request
 11 TLV; see 10.6.4 and 10.6.4.5) that the PTP Port at the other end of the attached link set its
 12 `currentLogGptpCapableMessageInterval` to a specific value can determine if the request was honored by examining the
 13 `logGptpCapableMessageInterval` field of subsequent received Signaling messages containing the gPTP-capable TLV.

14 NOTE 3—The `GptpCapableTransmit` state machine ensures that the times between transmission of successive Signaling
 15 messages, containing the gPTP-capable TLV, in seconds, are not smaller than $2^{currentLogGptpCapableMessageInterval}$. This is
 16 consistent with the manner in which `Pdelay_Req` messages are transmitted (see NOTE 3 of 11.5.2.2).

17 **10.7.3 Timeouts**

18 **10.7.3.1 syncReceiptTimeout**

19 The value of this attribute tells a `timeReceiver` port the number of sync intervals to wait without receiving
 20 synchronization information, before assuming that the `timeTransmitter` is no longer transmitting
 21 synchronization information and that the BTCA needs to be run, if appropriate. The condition of the
 22 `timeReceiver` port not receiving synchronization information for `syncReceiptTimeout` sync intervals is
 23 known as *sync receipt timeout*.

24 The default value shall be 3. **The range shall be 2 through 255.** The `syncReceiptTimeout` is a per-PTP Port
 25 attribute.

26 **10.7.3.2 announceReceiptTimeout**

27 The value of this attribute tells a `timeReceiver` port the number of announce intervals to wait without
 28 receiving an `Announce` message, before assuming that the `timeTransmitter` is no longer transmitting
 29 `Announce` messages, and that the BTCA needs to be run, if appropriate. The condition of the `timeReceiver`
 30 port not receiving an `Announce` message for `announceReceiptTimeout` announce intervals is known as
 31 *announce receipt timeout*.

32 The default value shall be 3. **The range shall be 2 through 255.** The `announceReceiptTimeout` is a per-PTP
 33 Port attribute.

34 **10.7.3.3 gPtpCapableReceiptTimeout**

35 The value of this attribute tells a PTP Port the number of gPTP-capable message intervals to wait without
 36 receiving from its neighbor a Signaling message containing a gPTP-capable TLV, before determining that its
 37 neighbor is no longer invoking gPTP.

38 NOTE—A determination that its neighbor is no longer invoking gPTP **causes** the PTP Port to set `asCapable` to FALSE.

39 The default value shall be 9. The range shall be 2 through 255.

11. Media-dependent layer specification for full-duplex point-to-point links

11.1 Overview

11.1.1 General

A port attached to a full-duplex point-to-point PTP Link uses the PTP peer delay protocol to measure propagation delay on the PTP Link. An overview of the propagation delay measurement is given in 11.1.2. Synchronization information is transported using the PTP messages Sync and Follow_Ups. An overview of the transport of synchronization information is given in 11.1.3. An overview of the MD entity model for a full-duplex point-to-point medium is given in 11.1.4.

11.1.2 Propagation delay measurement

The measurement of propagation delay on a full-duplex point-to-point PTP Link using the peer-to-peer delay mechanism is illustrated in Figure 11-1. The instance of the peer-to-peer delay mechanism that runs only on domain 0 is referred to as the transport-specific peer-to-peer delay mechanism. The instance of the peer-to-peer delay mechanism that runs as part of the Common Mean Link Delay Service (CMLDS, see 11.2.17) is referred to as CMLDS. The mechanism is the same as the peer-to-peer delay mechanism described in IEEE Std 1588-2019, specialized to a two-step PTP Port¹⁸ and sending the requestReceiptTimestamp and the responseOriginTimestamp separately [see item c) 8) of 11.4.2 in IEEE Std 1588-2019]. The transport-specific peer-to-peer delay mechanism is mathematically equivalent to CMLDS, though its computations are organized differently. The description in this subclause focuses on the transport-specific peer-to-peer delay mechanism. In addition, when it is not important in this standard to distinguish the transport-specific peer-to-peer delay mechanism from CMLDS, the mechanism is simply referred to as the “peer-to-peer delay mechanism,” or more briefly as the “peer delay mechanism.”

The measurement is made by each port at the end of every full-duplex point-to-point PTP Link. Thus, both ports sharing a PTP Link will independently make the measurement, and both ports will know the propagation delay as a result. This allows the time-synchronization information described in 11.1.3 to be transported irrespective of the direction taken by a Sync message. The propagation delay measurement is made on ports otherwise blocked by non-PTP algorithms (e.g., Rapid Spanning Tree Protocol) used to eliminate cyclic topologies. This enables either no loss of synchronization or faster resynchronization, after a reconfiguration, because propagation delays are already known and do not have to be initially measured when the reconfiguration occurs.

Since the propagation delay measurement is made using timestamps relative to the LocalClock entities at each port at the ends of the PTP Link and the resulting mean delay is expressed in the responder timebase (see 11.2.19.3.4), there is no need to measure the mean delay for the PTP Link in each domain because the mean delay is the same in each domain. In addition, the quantity **nrrPdelay** (see 11.2.13.13) is the ratio of the responder to requester LocalClock frequency and is also the same in all domains. Therefore, the propagation delay measurement and **nrrPdelay** measurement, made using the peer-to-peer delay mechanism (i.e., **nrrPdelay**, see 11.2.13.13), are domain-independent. Single instances of the respective state machines that cause these measurements to be made are invoked, rather than one instance per domain, and the results are available to all domains. The PTP messages used for the measurements (i.e., Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up; see 11.4.5 through 11.4.7) carry 0 in the domainNumber field, but this value is not used.

In Figure 11-1, the propagation delay measurement is initiated by a time-aware system at one end of a PTP Link; this time-aware system is known as the *peer delay initiator*. For purposes of the measurement, the other time-aware system is the *peer delay responder*. A similar measurement occurs in the opposite

¹⁸See 3.1.86 of IEEE Std 1588-2019 for the definition of a *two-step PTP Port*.

direction, with the initiator and responder interchanged and the directions of the messages in Figure 11-1 reversed.

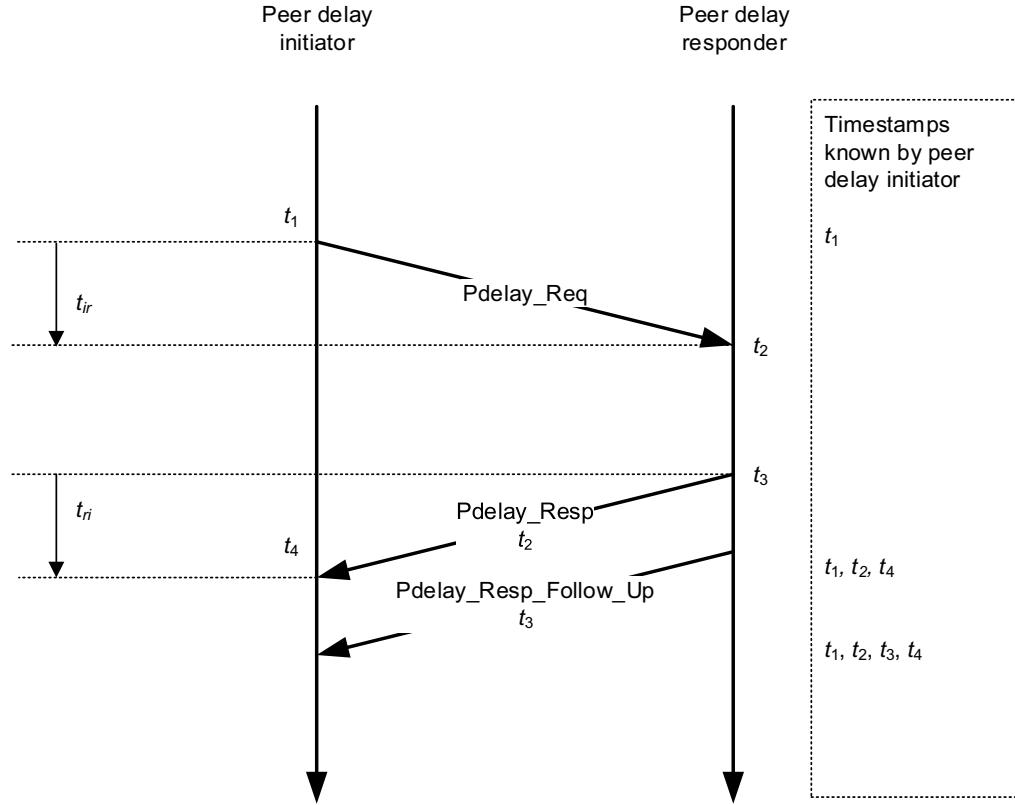


Figure 11-1—Propagation delay measurement using peer-to-peer delay mechanism

The propagation delay measurement starts with the initiator issuing a **Pdelay_Req** message and generating a timestamp, t_1 . The responder receives this message and timestamps it with time t_2 . The responder returns a **Pdelay_Resp** message and timestamps it with time t_3 . The responder returns the time t_2 in the **Pdelay_Resp** message and the time t_3 in a **Pdelay_Resp_Follow_Up** message. The initiator generates a timestamp, t_4 , upon receiving the **Pdelay_Resp** message. The initiator then uses these four timestamps to compute the mean propagation delay (i.e., `meanLinkDelay`; see 8.3) as shown in Equation (11-1).

$$\begin{aligned} t_{ir} &= t_2 - t_1 \\ t_{ri} &= t_4 - t_3 \\ D &= \frac{t_{ir} + t_{ri}}{2} = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \end{aligned} \quad (11-1)$$

where D is the measured mean propagation delay and the other quantities are defined in Figure 11-1.

Note that it is the mean propagation delay that is measured here. Any PTP Link asymmetry is modeled as described in 8.3. Any asymmetry that is not corrected for introduces an error in the transported synchronized time value.

The accuracy of the mean propagation delay measurement depends on how accurately the times t_1 , t_2 , t_3 , and t_4 are measured. In addition, Equation (11-1) assumes that the initiator and responder timestamps are taken relative to clocks that have the same frequency. In practice, t_1 and t_4 are measured relative to the LocalClock entity of the initiator time-aware system, and t_2 and t_3 are measured relative to the LocalClock entity of the responder time-aware system. If the propagation delay measurement is desired relative to the responder time base, the term $(t_4 - t_1)$ in Equation (11-1) must be multiplied by the rate ratio of the responder relative to the initiator, otherwise there will be an error equal to $0.5y(t_4 - t_1)$, where y is the frequency offset of the responder relative to the initiator. Likewise, if the propagation delay measurement is desired relative to the initiator time base, the term $(t_3 - t_2)$ in Equation (11-1) must be multiplied by the rate ratio of the initiator relative to the responder, otherwise there will be an error equal to $0.5y(t_3 - t_2)$, where y is the frequency offset of the initiator relative to the responder. Finally, if the propagation delay measurement is desired relative to the Grandmaster Clock time base, each term must be multiplied by the rate ratio of the Grandmaster Clock relative to the time base in which the term is expressed.

There can also be an error in measured propagation delay due to time measurement granularity (see B.1.2). For example, if the time measurement granularity is 40 ns (as specified in B.1.2), the timestamps t_1 , t_2 , t_3 , and/or t_4 can undergo 40 ns step changes. When this occurs, the measured propagation delay, D , will change by 20 ns (or by a multiple of 20 ns if more than one of the timestamps has undergone a 40 ns step change). The actual propagation delay has not changed by 20 ns; the effect is due to time measurement granularity. The effect can be reduced, and the accuracy improved, by averaging successive measured propagation delay values. For example, an exponential averaging filter can be used, i.e., as shown in Equation (11-2).

$$D_{avg,k} = aD_{avg,k-1} + (1-a)D_{k-1} \quad (11-2)$$

where

- D_k is the k^{th} propagation delay measurement
- $D_{avg,k}$ is the k^{th} computed average propagation delay
- k is an index for the propagation delay measurements (i.e., peer delay message exchange)

The quantity a is the exponential weighting factor; it can be set so that the weight of a past propagation delay measurement is $1/e$ after M measurements, i.e., as shown in Equation (11-3).

$$a = e^{-\frac{1}{M}} \quad (11-3)$$

The above averager must be initialized. One method is to use a simple average (i.e., the sum of the sample values divided by the number of samples) of the measurements made up to the current measurement until a window of M measurements has been accumulated. In this case, Equation (11-2) is used only for $k > M$. For $k \leq M$, the averaged propagation delay is given by Equation (11-4).

$$D_{avg,k} = \frac{(k-1)D_{avg,k-1} + D_{k-1}}{k} \quad (11-4)$$

The rate ratio of the responder relative to the initiator is the quantity [nrrPdelay](#) (see 11.2.13.13). When computed using peer-to-peer delay messages, it is computed by the function `computePdelayRateRatio()` (see 11.2.19.3.3) of the MDPdelayReq state machine (see 11.2.19) using successive values of t_3 and t_4 . As indicated in the description of `computePdelayRateRatio()`, any scheme that uses this information is acceptable as long as the performance requirements of B.2.4 are met. One example scheme is given in NOTE 1 of 11.2.19.3.3.

11.1.3 Transport of time-synchronization information

The transport of time-synchronization information by a PTP Instance, using Sync and Follow_Up (or just Sync) messages, is illustrated in Figure 11-2. The mechanism is mathematically equivalent to the mechanism described in IEEE Std 1588-2019 for a peer-to-peer Transparent Clock that is syntonized (see 10.1, 10.3, 11.1, and 11.4 of IEEE Std 1588-2019). However, the processes of transporting synchronization by a peer-to-peer Transparent Clock that is syntonized and by a Boundary Clock are mathematically and functionally equivalent. The main functional difference between the two types of clocks is that the Boundary Clock participates in best timeTransmitter selection and invokes the BTCA, while the peer-to-peer Transparent Clock does not participate in best timeTransmitter selection and does not invoke the BTCA (and implementations of the two types of clocks can be different).

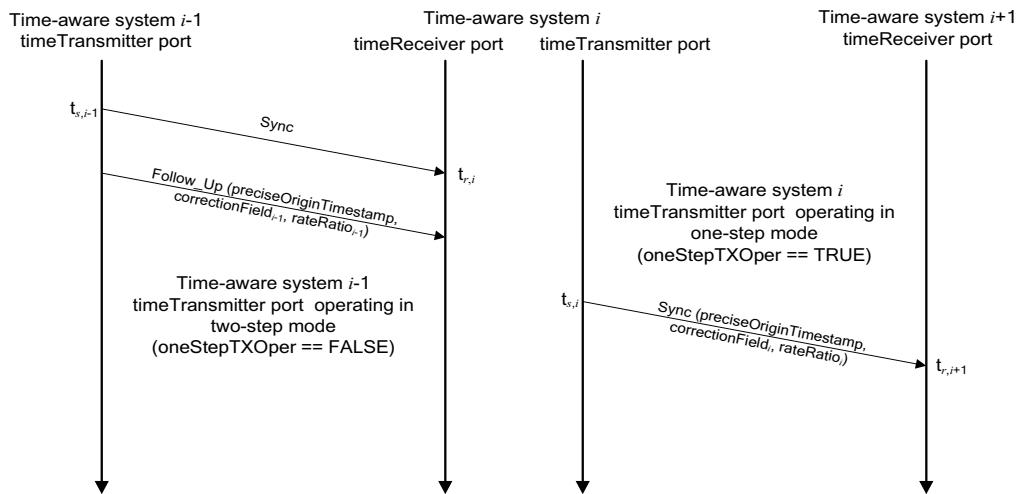


Figure 11-2—Transport of time-synchronization information

Figure 11-2 shows three adjacent PTP Instances, indexed $i-1$, i , and $i+1$. Synchronization is transported from PTP Instance $i-1$ to PTP Instance i using two-step time transport, and then to PTP Instance $i+1$ using one-step time transport. PTP Instance $i-1$ sends a Sync message to PTP Instance i at time $t_{s,i-1}$, relative to the LocalClock entity of PTP Instance $i-1$. At a later time (since it is using two-step time transport), PTP Instance $i-1$ sends an associated Follow_Up message to PTP Instance i , which contains a preciseOriginTimestamp, correctionField $_{i-1}$, and rateRatio $_{i-1}$. The preciseOriginTimestamp contains the time of the Grandmaster Clock when it originally sent this synchronization information. It is not indexed here because it normally does not change as the Sync and Follow_Up messages traverse the network. The quantity correctionField $_{i-1}$ contains the difference between the synchronized time when the Sync message is sent (i.e., the synchronized time that corresponds to the local time $t_{s,i-1}$) and the preciseOriginTimestamp. The sum of preciseOriginTimestamp and correctionField $_{i-1}$ gives the synchronized time that corresponds to $t_{s,i-1}$. The quantity rateRatio $_{i-1}$ is the ratio of the Grandmaster Clock frequency to the frequency of the LocalClock entity of PTP Instance $i-1$.

PTP Instance i receives the Sync message from PTP Instance $i-1$ at time $t_{r,i}$, relative to its LocalClock entity. It timestamps the receipt of the Sync message, and the timestamp value is $t_{r,i}$. It receives the associated Follow_Up message some time later.

PTP Instance i will eventually send a new Sync message at time $t_{s,i}$, relative to its LocalClock entity. The time $t_{s,i}$ occurs after the Follow_Up message is received from PTP Instance $i-1$. It will have to compute correctionField_i , i.e., the difference between the synchronized time that corresponds to $t_{s,i}$ and the $\text{preciseOriginTimestamp}$. To do this calculation, it must compute the value of the time interval between $t_{s,i-1}$ and $t_{s,i}$, expressed in the Grandmaster Clock time base. This interval is equal to the sum of the following quantities:

- a) The propagation delay on the PTP Link between PTP Instances $i-1$ and i , expressed in the Grandmaster Clock time base, and
- b) The difference between $t_{s,i}$ and $t_{r,i}$ (i.e., the residence time), expressed in the Grandmaster Clock time base.

The mean propagation delay on the PTP Link between PTP Instances $i-1$ and i , relative to the LocalClock entity of PTP Instance $i-1$, is equal to meanLinkDelay (see 10.2.5.8). This must be multiplied by rateRatio_{i-1} to express it in the Grandmaster Clock time base. The total propagation delay is equal to the mean propagation delay plus the quantity delayAsymmetry (see 8.3 and 10.2.5.9); delayAsymmetry is already expressed in the Grandmaster Clock time base. The residence time, $t_{s,i} - t_{r,i}$, must be multiplied by rateRatio_i to express it in the Grandmaster Clock time base.

The preceding computation is organized slightly differently in the state machines of 11.2.14 and 11.2.15. Rather than explicitly expressing the PTP Link propagation delay in the Grandmaster Clock time base, the local time at PTP Instance i that corresponds to $t_{s,i-1}$ is computed; this is the upstreamTxTime member of the MDSyncReceive structure (see 10.2.2.2.7; recall that $t_{s,i-1}$ is relative to the LocalClock entity of PTP Instance $i-1$). upstreamTxTime is equal to the quantity $t_{r,i}$ minus the PTP Link propagation delay expressed relative to the LocalClock entity of PTP Instance i . The PTP Link propagation delay expressed relative to the LocalClock entity of PTP Instance i is equal to the sum of the following:

- c) The quantity meanLinkDelay (see 10.2.5.8) divided by nrrPdelay (see 11.2.13.13), and
- d) The quantity delayAsymmetry (see 10.2.5.9) divided by rateRatio_i .

The division of delayAsymmetry by rateRatio_i is performed after rateRatio_i has been updated. The computation of upstreamTxTime is done by the MDSyncReceiveSM state machine in the function $\text{setMDSyncReceiveMSR}()$ (see 11.2.14.2.1). When PTP Instance i sends a Sync message to PTP Instance $i+1$, it computes the sum of the PTP Link propagation delay and residence time, expressed in the Grandmaster Clock time base, as shown in item e).

- e) The quantity $(t_{s,i} - \text{upstreamTxTime})(\text{rateRatio}_i)$.

As in item d) in this subclause, this computation is performed after rateRatio_i has been updated. The quantity of item e) is added to $\text{correctionField}_{i-1}$ to obtain correctionField_i . The computation of item e) and correctionField_i is done by the MDSyncSendSM state machine in the function $\text{setFollowUp}()$ (see 11.2.15.2.3). The quantity correctionField_i is inserted in the Sync message sent by PTP Instance i .

The difference between mean propagation delay relative to the Grandmaster Clock time base and relative to the time bases of the PTP Instance at the other end of the attached PTP Link or of the current PTP Instance is usually negligible. The former can be obtained from the latter by multiplying the latter by the ratio of the Grandmaster Clock frequency to the frequency of the LocalClock entity of the PTP Instance at the other end of the PTP Link attached to this PTP Port. This ratio differs from 1 by 200 ppm or less. For example, for a worst-case frequency offset of the LocalClock entity of the PTP Instance at the other end of the PTP Link,

relative to the Grandmaster Clock, of 200 ppm, and a measured propagation time of 100 ns, the difference in D relative to the two time bases is 20 ps. The corresponding difference for PTP Link delay asymmetry in this example is also negligible because the magnitude of the PTP Link delay asymmetry is of the same order of magnitude as the mean propagation time, or less. However, the difference is usually not negligible for residence time because residence time can be much larger (see B.2.2).

It was indicated in the first paragraph of this subclause that the processes of transporting synchronization by a peer-to-peer Transparent Clock that is syntonized and by a Boundary Clock are mathematically and functionally equivalent. The reason for this equivalency is that the computations described so far in this subclause compute the synchronized time when the Sync message is sent by the PTP Instance. The same computations are done if PTP Instance i sends a Sync message without having received a new Sync message, i.e., if Sync receipt timeout occurs (see 10.7.3.1). In this case, PTP Instance i uses the most recently received time-synchronization information from PTP Instance $i-1$, which would be prior to when PTP Instance i sent its most recent Sync message. The synchronized time corresponding to the sending of a Sync message is equal to the sum of the preciseOriginTimestamp and correctionField. Normally a Boundary Clock places this entire value, except for any sub-nanosecond portion, in the preciseOriginTimestamp, while a Transparent Clock retains the preciseOriginTimestamp and updates the correctionField. However, the sum of the two fields is equal to the synchronized time when the Sync message is sent in both cases.

The ratio of the Grandmaster Clock frequency to the frequency of the LocalClock entity at PTP Instance i , rateRatio_i , is equal to the same quantity at PTP Instance $i-1$, rateRatio_{i-1} , multiplied by the ratio of the frequency of the LocalClock entity at PTP Instance $i-1$ to the frequency of the LocalClock entity at PTP Instance i , nrrPdelay (see 11.2.13.13). If nrrPdelay is sufficiently small, this is approximately equal to the sum of rateRatio_{i-1} and the quantity nrrPdelay_{i-1} , which is the frequency offset of PTP Instance $i-1$ relative to PTP Instance i . This computation is done by the PortSyncSyncReceive state machine (see 10.2.8).

NOTE—The sending of time-synchronization information by the timeTransmitter ports of a PTP Instance might or might not be tightly synchronized with the receipt of time-synchronization information by the timeReceiver port. If a timeTransmitter port has the same logMessageInterval as the timeReceiver port, it will transmit timing event messages as soon as possible after the timeReceiver port has received the corresponding timing event messages and the timeTransmitter port is operating in “syncLocked” mode (see 10.2.5.15). If a timeTransmitter port and timeReceiver port have different logMessageInterval values, then the timeTransmitter port can send timing event messages without any synchronization with the timeReceiver port.

11.1.4 Model of operation

A PTP Instance contains one MD entity per PTP Instance, per PTP Port. This entity contains functions generic to all media, which are described in Clause 10, and functions specific to the respective medium for the PTP Link. Functions specific to full-duplex point-to-point links are described in the current clause.

NOTE—Full-duplex point-to-point IEEE 802.3 links are in the category of links specified in this clause.

The model for a PTP Instance of a time-aware system with full-duplex point-to-point links is shown in Figure 11-3. It assumes the presence of one full-duplex point-to-point MD entity per PTP Port. The media-independent entities shown in Figure 11-3 are described in 10.1.2.

A general, media-independent description of the generation of timestamps is given in 8.4.3. A more specific description for PTP event messages is given in 11.3.2.1. A PTP event message is timestamped relative to the LocalClock entity when the message timestamp point (see 3.17) crosses the timestamp measurement plane (see 3.33). The timestamp is corrected for any ingressLatency or egressLatency (see 8.4.3) to produce a timestamp relative to the reference plane (see 3.26). The corrected timestamp value is provided to the MD entity.

The MD entity behavior and detailed state machines specific to full-duplex point-to-point links are described in 11.2. The behavior of the MD entity that is generic to all media is described in Clause 10.

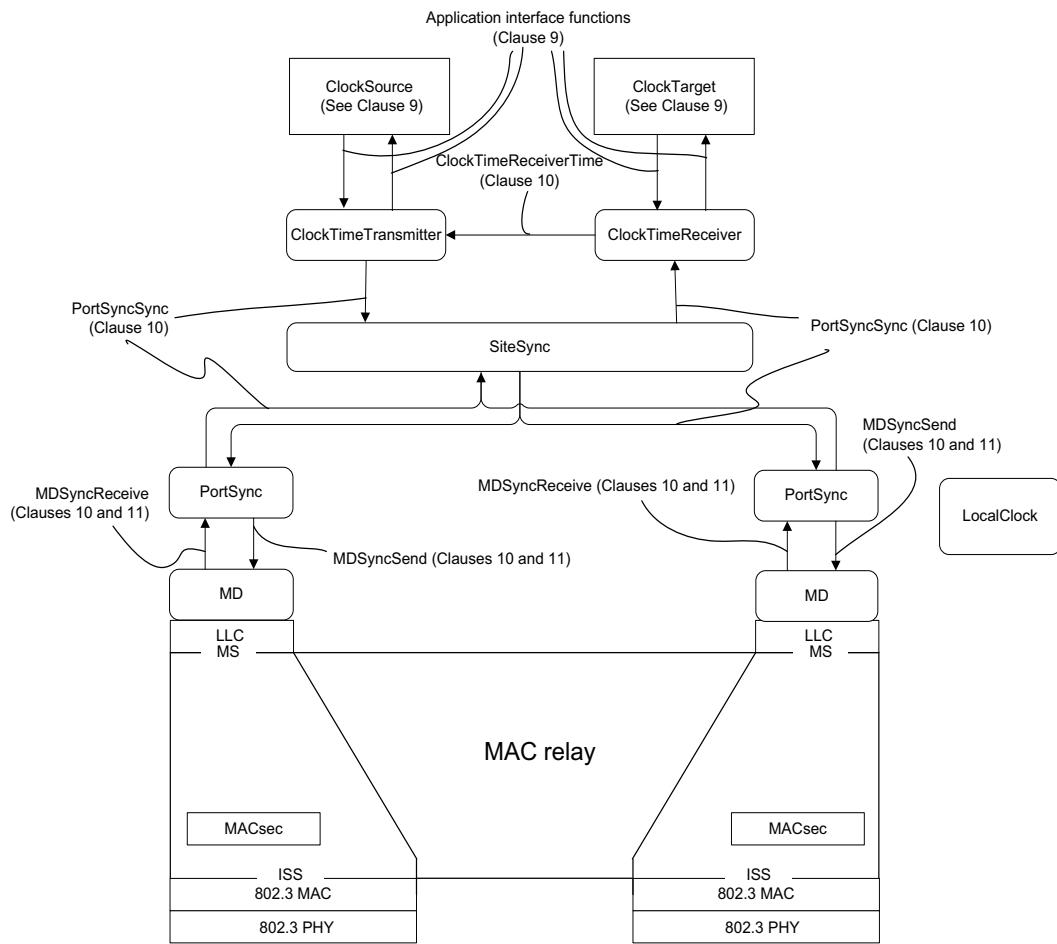


Figure 11-3—Model for a PTP Instance of a time-aware system with full-duplex point-to-point links

11.2 State machines for MD entity specific to full-duplex point-to-point links

11.2.1 General

This subclause describes the media-dependent state machines for an MD entity, for full-duplex point-to-point links. The state machines are all per port because an instance of each is associated with an MD entity. The state machines are as follows:

- a) MDSyncReceiveSM (shown in Figure 10-2, and in more detail in Figure 11-4): receives Sync and, if the received information is two-step, Follow_Up messages, and sends the time-synchronization information carried in these messages to the PortSync entity of the same PTP Port. In addition, if the global variable driftTrackingTlvSupport is TRUE, this state machine computes the ratio of the frequency of the LocalClock entity in the PTP Instance at the other end of the attached PTP Link to the frequency of the LocalClock entity in this PTP Instance using the information contained in successive Drift_Tracking TLVs, and places the result in the global variable nrrSync (see 11.2.13.14). The global variable neighborRateRatio (see 10.2.5.7) is set equal to nrrSync or nrrPdelay (see 11.2.13.13) depending on whether the value of the global variable nrrCompMethod (see 11.2.13.15) is equal to Sync or Pdelay, respectively. There is one instance of this state machine per PTP Port, per domain.
- b) MDSyncSendSM (shown in Figure 10-2, and in more detail in Figure 11-4): receives an MDSyncSend structure from the PortSync entity of the same PTP Port; transmits a Sync message; uses the syncEventEgressTimestamp, corrected for egressLatency, and information contained in the MDSyncSend structure to compute information needed for the Sync message if the PTP Port is currently operating as a one-step PTP Port and for the corresponding Follow_Up message if the PTP Port is currently operating as a two-step PTP Port; and transmits the Follow_Up message if the PTP Port is two-step. There is one instance of this state machine per PTP Port, per domain.

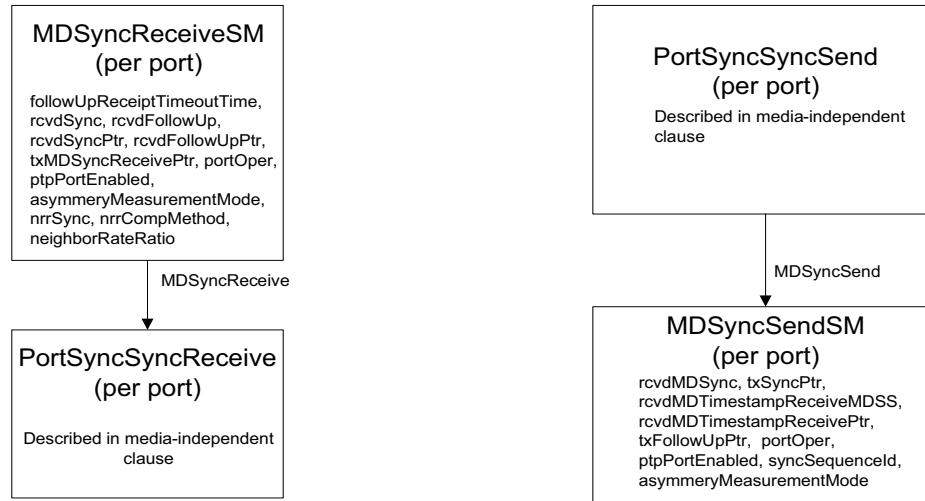
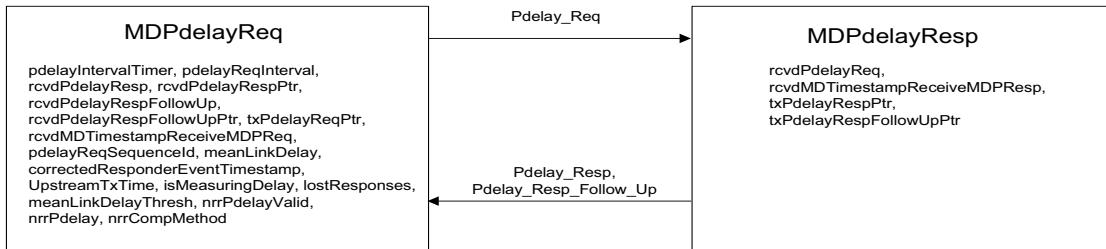


Figure 11-4—Detail of MD entity time-synchronization state machines for full-duplex point-to-point links

- c) MDPdelayReq (shown in Figure 11-5): transmits a Pdelay_Req message, receives Pdelay_Resp and Pdelay_Resp_Follow_Up messages corresponding to the transmitted Pdelay_Req message, uses the information contained in successive Pdelay_Resp and Pdelay_Resp_Follow_Up messages to compute the ratio of the frequency of the LocalClock entity in the PTP Instance at the other end of the attached PTP Link to the frequency of the LocalClock entity in this PTP Instance, places the

result in the global variable `nrrPdelay` (see 11.2.13.13), and uses the information obtained from the message exchange and the computed frequency ratio to compute propagation delay on the attached PTP Link. There is one instance of this state machine for all the domains, per port.

- d) MDPdelayResp (shown in Figure 11-5): receives a `Pdelay_Req` message from the MD entity at the other end of the attached PTP Link, and responds with `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages. There is one instance of this state machine for all the domains, per port.



**Figure 11-5—Peer-to-peer delay mechanism state machines—
overview and interrelationships**

- e) SyncIntervalSetting state machine (not shown): receives a Signaling message that contains a Message Interval Request TLV (see 10.6.4.3), and sets the global variables that give the duration of the mean intervals between successive Sync messages. There is one instance of this state machine per PTP Port, per domain.
- f) LinkDelayIntervalSetting state machine (not shown): receives a Signaling message that contains a Message Interval Request TLV (see 10.6.4.3), and sets the global variables that give the duration of the mean intervals between successive `Pdelay_Req` messages. There is one instance of this state machine for all the domains, per port.

Figure 10-2, Figure 11-4, and Figure 11-5 are not themselves state machines, but illustrate the machines, their interrelationships, the principle variables and messages used to communicate between them, their local variables, and performance parameters. The figures do not show the service interface primitives between the media-dependent layer and the logical link control (LLC). Figure 11-5 is analogous to Figure 10-2; while Figure 10-2 applies to the general time-synchronization protocol, Figure 11-5 is limited to the peer-to-peer delay mechanism for measurement of propagation delay in full-duplex point-to-point links. Figure 11-4 shows greater detail of the `MDSyncReceiveSM` and `MDSyncSendSM` state machines for full-duplex point-to-point links than Figure 10-2.

The state machines described in 11.2 use some of the global per PTP Instance system variables specified in 10.2.4, the global per-port variables specified in 10.2.5, and the functions specified in 10.2.6.

11.2.2 Determination of `asCapable` and `asCapableAcrossDomains`

There is one instance of the global variable `asCapable` (see 10.2.5.1) per PTP Port, per domain. There is one instance of the global variable `asCapableAcrossDomains` (see 11.2.13.12), per port, that is common across, and accessible by, all the domains.

The per-PTP Port global variable `asCapable` (see 10.2.5.1) indicates whether the IEEE 802.1AS protocol is operating, in this domain, on the PTP Link attached to this PTP Port, and can provide the time-

synchronization performance described in B.3. asCapable is used by the PortSync entity, which is media-independent; however, the determination of asCapable is media-dependent.

The per-port global variable asCapableAcrossDomains is set by the MDPdelayReq state machine (see 11.2.19 and Figure 11-9). For a port attached to a full-duplex point-to-point PTP Link, asCapableAcrossDomains shall be set to TRUE if and only if it is determined, via the **transport-specific** peer-to-peer delay mechanism or CMLDS, that the following conditions hold for the port:

- a) The port is exchanging peer delay messages with its neighbor,
- b) The measured delay does not exceed meanLinkDelayThresh,
- c) The port does not receive multiple Pdelay_Resp or Pdelay_Resp_Follow_Up messages in response to a single Pdelay_Req message, and
- d) The port does not receive a response from itself or another PTP Port of the same PTP Instance.

NOTE 1—if a PTP Instance implements only domain 0 and the MDPdelayReq and MDPdelayResp state machines are invoked on domain 0 (see 11.2.19), asCapableAcrossDomains is still set by the MDPdelayReq state machine.

The default value of meanLinkDelayThresh shall be set as specified in Table 11-1.

Table 11-1—Value of meanLinkDelayThresh for various links

Link	Value of meanLinkDelayThresh (ns) (see NOTE)
100BASE-TX, 1000BASE-T	800 ₁₀
100BASE-FX, 1000BASE-X	FFFF FFFF FFFF FFFF FFFF FFFF ₁₆

NOTE—The actual propagation delay for 100BASE-TX and 1000BASE-T links is expected to be smaller than the above respective threshold. If the measured mean propagation delay (i.e., meanLinkDelay; see 10.2.5.8) exceeds this threshold, it is assumed that this is due to the presence of equipment that does not implement gPTP. For 100BASE-FX and 1000BASE-X links, the actual propagation delay can be on the order of, or larger than, the delay produced by equipment that does not implement gPTP; therefore, such equipment cannot be detected by comparing measured propagation delay with a threshold. In this case, meanLinkDelayThresh is set to the largest possible value (i.e., all 1s).

The per-PTP Port, per-domain global variable asCapable shall be set to TRUE if and only if the following conditions hold:

- e) The value of asCapableAcrossDomains is TRUE, and
- f) One of the following conditions holds:
 - 1) The value of neighborGptpCapable for this PTP Port is TRUE, or
 - 2) The value of domainNumber is zero, and the value of sdId for peer delay messages received on this PTP Port is 0x100.

NOTE 2—Condition f) 2) ensures backward compatibility with the 2011 edition of this standard. A PTP Instance compliant with the current edition of this standard that is attached, via a full-duplex point-to-point PTP Link, to a PTP Instance compliant with the 2011 edition of this standard will not receive Signaling messages that contain the **gPTP-capable TLV** and will not set neighborGptpCapable to TRUE. However, condition f) 2) ensures that asCapable for this PTP Port and domain (i.e., domain 0) will still be set to TRUE if condition e) holds because the peer delay messages received from the time-aware system compliant with the 2011 edition of this standard will have sdId set to 0x100.

11.2.3 Use of MAC Control PAUSE operation

A PTP Instance shall not use the MAC Control PAUSE operation.

11.2.4 Use of priority-based flow control

A PTP Instance that implements priority-based flow control shall neither transmit nor honor upon receipt priority-based flow control messages that act on the IEEE 802.1AS message priority code point (see 8.4.4).

11.2.5 Use of link aggregation

gPTP PDUs, when used with physical ports that are attached to an IEEE 802.1AX™ Link Aggregation Group, shall be transmitted and received over the physical ports (Aggregation Ports), not the Aggregator Port. Using the Parser/Multiplexer functions described in 6.1.3 and 6.2.7 of IEEE Std 802.1AX-2014, received gPTP PDUs are recognized as control frames, separated from the incoming data stream, and directed to the gPTP layers. gPTP PDUs output from the gPTP layers are integrated into the port’s data stream by the Parser/Multiplexer.

Assuming that `ptpPortEnabled` is true for all the physical links (Aggregation Links) in a Link Aggregation Group and that all the physical links are connected to the same two systems, gPTP will measure the delay on each physical link, and the IEEE 802.1AS protocol will choose one of the physical links for transmitting time from the `timeTransmitter` clock.

11.2.6 Service interface primitives and data structures communicated between state machines

The following subclauses describe the service primitives and data structures communicated between the time-synchronization state machines of the MD entity. First the service primitives are described, followed by the data structures.

11.2.7 DL-UNITDATA.request

This service primitive is used by an MD entity to request to the associated LLC the transmission of a Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, or Pdelay_Resp_Follow_Up message. The primitive is described in 2.2.1.1.1 of ISO/IEC 8802-2:1998 [B16].

11.2.8 DL-UNITDATA.indication

This service primitive is used by the LLC to indicate to the associated MD entity the receipt of a Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, or Pdelay_Resp_Follow_Up message. The primitive is described in 2.2.1.1.1 of ISO/IEC 8802-2:1998 [B16].

11.2.9 MDTimestampReceive

11.2.9.1 General

This structure provides the timestamp, relative to the timestamp measurement plane, of the event message that was just sent or just received. The structure is received by the MD entity of the PTP Port. The MD entity corrects this timestamp for any ingressLatency or egressLatency (see 8.4.3) before using it and/or passing it to higher layer entities.

```
MDTimestampReceive{  
    timestamp  
}
```

The member of the structure is defined in the following subclause.

11.2.9.2 timestamp (UScaledNs)

The timestamp is the value of the timestamp of the event message (i.e., Sync, Pdelay_Req, Pdelay_Resp) that was just transmitted or received. This timestamp is taken relative to the timestamp measurement plane.

11.2.10 MDSyncReceive

This structure is specified in 10.2.2.2.

11.2.11 MDSyncSend

This structure is specified in 10.2.2.1.

11.2.12 Overview of MD entity global variables

Subclause 11.2.13 defines global variables used by the MD entity state machines whose scopes are as follows:

- Per PTP Instance (i.e., per domain)
- Per PTP Instance, per PTP Port
- Instances used by CMLDS (see 11.2.17) (i.e., variable is common across all [Link Ports](#))
- Instances used by CMLDS, per [Link Port](#)

Table 11-2 summarizes the scope of each global variable of 11.2.13.

Table 11-2 — Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per Link Port
currentLogPdelayReqInterval	11.2.13.1	No	Yes	No	Yes
initialLogPdelayReqInterval	11.2.13.2	No	Yes	No	Yes
pdelayReqInterval	11.2.13.3	No	Yes	No	Yes
allowedLostResponses	11.2.13.4	No	Yes	No	Yes
allowedFaults	11.2.13.5	No	Yes	No	Yes

Table 11-2 — Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5)

Variable name	Subclause of definition	Per PTP Instance (i.e., per domain)	Per PTP Instance, per PTP Port	Instance used by CMLDS (i.e., variable is common across all Link Ports)	Instance used by CMLDS, per Link Port
isMeasuringDelay	11.2.13.6	No	Yes	No	Yes
meanLinkDelayThresh	11.2.13.7	No	Yes	No	Yes
syncSequenceId	11.2.13.8	No	Yes	No	No
oneStepReceive	11.2.13.9	No	Yes	No	No
oneStepTransmit	11.2.13.10	No	Yes	No	No
oneStepTxOper	11.2.13.11	No	Yes	No	No
asCapableAcrossDomains	11.2.13.12	No	Yes	No	Yes
nrrPdelay	11.2.13.13	No	Yes	No	Yes
nrrSync	11.2.13.14	No	Yes	No	No
nrrCompMethod	11.2.13.15	No	Yes	No	No

11.2.13 MD entity global variables

11.2.13.1 currentLogPdelayReqInterval: The current value of the logarithm to base 2 of the mean time interval, in seconds, between the sending of successive Pdelay_Req messages (see 11.5.2.2). This value is set in the LinkDelayIntervalSetting state machine (see 11.2.21). The data type for currentLogPdelayReqInterval is Integer8. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.2 initialLogPdelayReqInterval: The initial value of the logarithm to base 2 of the mean time interval, in seconds, between the sending of successive Pdelay_Req messages (see 11.5.2.2). The data type for initialLogPdelayReqInterval is Integer8. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.3 pdelayReqInterval: A variable containing the mean Pdelay_Req message transmission interval for the port corresponding to this MD entity. The value is set in the LinkDelayIntervalSetting state machine (see 11.2.21). The data type for pdelayReqInterval is UScaledNs. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.4 allowedLostResponses: The number of Pdelay_Req messages without valid responses above which a port is considered to be not exchanging peer delay messages with its neighbor. The data type for allowedLostResponses is UInteger8. The default value and range of allowedLostResponses is given in 11.5.3. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.5 allowedFaults: The number of faults [i.e., instances where meanLinkDelay (see 10.2.5.8) exceeds meanLinkDelayThresh (see 11.2.13.7) and/or the computation of nrrPdelay (see 11.2.13.13) is invalid

(see 11.2.19.2.10)] above which `asCapableAcrossDomains` is set to FALSE, i.e., the port is considered not capable of interoperating with its neighbor via the IEEE 802.1AS protocol (see 10.2.5.1). The data type for `allowedFaults` is `UInteger8`. The default value and range of `allowedFaults` is given in 11.5.4. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.6 `isMeasuringDelay`: A Boolean that is TRUE if the port is receiving `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages from the port at the other end of the PTP Link. For a full-duplex point-to-point PTP Link, the port is measuring PTP Link propagation delay if it is receiving `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages from the port at the other end of the PTP Link (i.e., it performs the measurement using the peer-to-peer delay mechanism). There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

11.2.13.7 `meanLinkDelayThresh`: The propagation time threshold above which a port is considered not capable of participating in the IEEE 802.1AS protocol. If `meanLinkDelay` (see 10.2.5.8) exceeds `meanLinkDelayThresh`, then `asCapableAcrossDomains` (see 11.2.13.12) is set to FALSE. The data type for `meanLinkDelayThresh` is `UScaledNs`. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port), and also one instance of this variable for domain 0 if domain 0 is implemented. The variable is accessible by all the domains.

NOTE—The variable `meanLinkDelayThresh` was named `neighborPropDelayThresh` in the 2011 edition of this standard.

11.2.13.8 `syncSequenceId`: The `sequenceId` (see 11.4.2.8) for the next Sync message to be sent by this MD entity. The data type for `syncSequenceId` is `UInteger16`.

11.2.13.9 `oneStepReceive`: A Boolean variable that is TRUE if the PTP Port is capable of receiving one-step Sync messages.

11.2.13.10 `oneStepTransmit`: A Boolean variable that is TRUE if the PTP Port is capable of transmitting one-step Sync messages.

11.2.13.11 `oneStepTxOper`: A Boolean variable that is TRUE if the PTP Port will be transmitting one-step Sync messages (see 11.1.3).

NOTE—Since a one-step port modifies Sync message fields (e.g., the `correctionField`) “on the fly” as the Sync message is being transmitted, the modifying of the `correctionField` of the Sync message is often implemented in hardware. Each distinct PTP Instance of a time-aware system runs in a distinct domain and tracks a distinct time. If a time-aware system supports more than one PTP Instance, support for one-step often requires hardware for each PTP Instance (e.g., four domains requires four hardware components), but support for two-step can share hardware among all PTP Instances.

11.2.13.12 `asCapableAcrossDomains`: A Boolean that is TRUE if and only if conditions a) through d) of 11.2.2 are satisfied. This Boolean is set by the MDPdelayReq state machine and is used in determining `asCapable` for a port (see 11.2.2). There is one instance of this variable for all the domains (per port). The variable is accessible by all the domains. When only one domain is active, `asCapableAcrossDomains` is equivalent to the variable `asCapable` (see 10.2.5.1).

11.2.13.13 `nrrPdelay`: The `neighborRateRatio` (see 10.2.5.7) computed using the transport-specific peer-to-peer delay mechanism or CMLDS (see 11.2.19). The data type for `nrrPdelay` is `Float64`.

11.2.13.14 `nrrSync`: The `neighborRateRatio` (see 10.2.5.7) computed using the `Drift_Tracking` TLV carried by successive Sync or, if two-step, `Follow_Up` messages (see 11.2.14). The data type for `nrrSync` is `Float64`.

11.2.13.15 `nrrCompMethod`: An `Enumeration1` that takes on the values `Sync` and `Pdelay` to indicate the source of the value of `neighborRateRatio` (see 10.2.5.7):

- a) If the value is Sync and driftTrackingTlvSupport (see 10.2.4.28) is TRUE, neighborRateRatio is populated with the value of nrrSync whenever a new value of nrrSync is computed.
- b) If the value is Pdelay or if driftTrackingTlvSupport (10.2.4.28) is FALSE, neighborRateRatio is populated with the value of nrrPdelay whenever a new value of nrrPdelay is computed.

11.2.14 MDSyncReceiveSM state machine

11.2.14.1 State machine variables

The following variables are used in the state diagram in Figure 11-6 (in 11.2.14.3):

11.2.14.1.1 followUpReceiptTimeoutTime: A variable used to save the time at which the information conveyed by a received Sync message will be discarded if the associated Follow_Up message is not received by then. The data type for followUpReceiptTimeoutTime is UScaledNs.

11.2.14.1.2 rcvdSync: A Boolean variable that is set to TRUE when a Sync message is received. This variable is reset to FALSE by the current state machine.

11.2.14.1.3 rcvdFollowUp: A Boolean variable that notifies the current state machine when a Follow_Up message is received. This variable is reset by the current state machine.

11.2.14.1.4 rcvdSyncPtr: A pointer to a structure whose members contain the values of the fields of the Sync message whose receipt is indicated by rcvdSync (see 11.2.14.1.2).

11.2.14.1.5 rcvdFollowUpPtr: A pointer to a structure whose members contain the values of the fields of the Follow_Up message whose receipt is indicated by rcvdFollowUp (see 11.2.14.1.3).

11.2.14.1.6 txMDSyncReceivePtrMDSR: A pointer to a structure whose members contain the values of the parameters of an MDSyncReceive structure to be transmitted.

11.2.14.1.7 upstreamSyncInterval: The sync interval (see 10.7.2.1) for the upstream PTP Port that sent the received Sync message. The data type for upstreamSyncInterval is UScaledNs.

11.2.14.1.8 TEMP: A temporary variable used to reduce clutter in the state diagram (see Figure 11-6). The data type for TEMP is Integer16.

11.2.14.2 State machine functions

11.2.14.2.1 setMDSyncReceiveMDSR(): Creates an MDSyncReceive structure, and returns a pointer to this structure. The members of this structure are set as follows:

- a) If twoStepFlag of the most recently received Sync message is TRUE, followUpCorrectionField is set equal to the sum of the correctionField (see 11.4.2.6) of the most recently received Sync message, plus the correctionField of the most recently received Follow_Up message. Otherwise, followUpCorrectionField is set equal to the correctionField of the most recently received Sync message.
- b) sourcePortIdentity is set equal to the sourcePortIdentity (see 11.4.2.7) of the most recently received Sync message (see 11.4.3).
- c) logMessageInterval is set equal to the logMessageInterval (see 11.4.2.9) of the most recently received Sync message (see 11.4.3).
- d) If twoStepFlag of the most recently received Sync message is TRUE, preciseOriginTimestamp is set equal to the preciseOriginTimestamp (see 11.4.4.2.1) of the most recently received Follow_Up

message. Otherwise, preciseOriginTimestamp is set equal to the originTimestamp (see 11.4.3.2.1) of the most recently received Sync message.

- e) rateRatio is set equal to the quantity $(\text{cumulativeScaledRateOffset} \times 2^{-41}) + 1.0$, where the cumulativeScaledRateOffset field is for the most recently received Follow_Up information TLV (see 11.4.4.3.6).
- f) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the most recently received Sync message (if twoStepFlag is FALSE) or Follow_Up message (if twoStepFlag is TRUE) contains a Drift_Tracking TLV:
 - 1) syncGrandmasterIdentity (see 10.2.4.26) is set equal to the syncGrandmasterIdentity of the Drift_Tracking TLV.
 - 2) If the value of syncStepsRemoved of the Drift_Tracking TLV is not equal to 0xFFFF, syncStepsRemoved (see 10.2.4.27) is set equal to the value of syncStepsRemoved of the Drift_Tracking TLV plus one. If the value of syncStepsRemoved of the Drift_Tracking TLV is equal to 0xFFFF, syncStepsRemoved (see 10.2.4.27) is set equal to 0xFFFF.
 - 3) nrSync (see 11.2.13.14) is computed using the following information conveyed by successive Sync and, if twoStepFlag is TRUE, Follow_Up messages; the specific algorithms are beyond the scope of this document:
 - i) The syncEventIngressTimestamp values for the respective Sync messages.
 - ii) The successive correctedSyncEgressTimestamp values, whose data type is UScaledNs, obtained by adding (A) the seconds field of the syncEgressTimestamp of the Drift_Tracking TLV, multiplied by 10^9 , (B) the nanoseconds field of the syncEgressTimestamp of the Drift_Tracking TLV, and (C) CMLDS delayAsymmetry (i.e., cmldsLinkPort.delayAsymmetry, see 10.2.5.9) if CMLDS is provided, otherwise instance-specific delayAsymmetry (i.e., portDS.delayAsymmetry, see 14.8.10) divided by rateRatio (see 10.2.8.1.4)
 - iii) The rateRatioDrift field of the Drift_Tracking TLV.

NOTE 1—If delayAsymmetry does not change during the time interval over which nrSync is computed, it is not necessary to add it in f) 3) ii) (C) because in that case it will be canceled when computing the difference between earlier and later correctedSyncEgressTimestamps.

NOTE 2—The methodology in f) 3) for computing nrSync follows the methodology described in 11.2.19.3.3 for computing nrPdelay using information contained in peer delay messages. See the NOTES and description in 11.2.19.3.3 for more detail. The variable correctedSyncEgressTimestamp is computed in a manner analogous to the computation of the variable correctedResponderEventTimestamp in 11.2.19.3.3.

NOTE 3—The use of the rateRatioDrift, e.g., in compensating for frequency drift when computing rateRatio (see the RECEIVED_SYNC state of the PortSyncSyncReceive state machine of Figure 10-4), is implementation specific or profile standard specific.

- g) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the most recently received Sync message (if twoStepFlag is FALSE) or Follow_Up message (if twoStepFlag is TRUE) does not contain a Drift_Tracking TLV:
 - 1) syncGrandmasterIdentity (see 10.2.4.26) is set equal to 0xFFFF FFFF FFFF FFFF.
 - 2) syncStepsRemoved (see 10.2.4.27) is set equal to 0xFFFF.
- h) The value of neighborRateRatio (see 10.2.5.7) is set equal to the value of nrSync (see 11.2.13.14) or nrPdelay (see 11.2.13.13) based on the value of nrCompMethod (see 11.2.13.15).
- i) upstreamTxTime is set equal to the syncEventIngressTimestamp for the most recently received Sync message (see 11.4.3), minus the mean propagation time on the PTP Link attached to this PTP Port (meanLinkDelay; see 10.2.5.8) divided by nrPdelay (see 11.2.13.13), minus delayAsymmetry (see 10.2.5.9) for this PTP Port divided by the sum of rateRatio [see item e) in this subclause] and the quantity neighborRateRatio – 1. The syncEventIngressTimestamp is equal to the timestamp value

measured relative to the timestamp measurement plane, minus any ingressLatency (see 8.4.3). The upstreamTxTime can be written as follows:

$$\text{upstreamTxTime} = \text{syncEventIngressTimestamp} - (\text{meanLinkDelay}/\text{nrrPdelay}) - (\text{delayAsymmetry}/(\text{rateRatio} \times \text{neighborRateRatio}))$$

NOTE 4—The syncEventIngressTimestamp is measured relative to the time base of the current PTP Instance. The mean propagation delay is divided by neighborRateRatio to convert it from the time base of the PTP Instance at the other end of the attached PTP Link to the time base of the current PTP Instance. delayAsymmetry (i.e., portDS.delayAsymmetry, see 14.8.10) is divided by the new rateRatio to convert it from the time base of the Grandmaster Clock to the time base of the current PTP Instance.

NOTE 5—meanLinkDelay is divided by nrrPdelay because the meanLinkDelay calculation is done using the peer-to-peer delay message timestamps. However, neighborRateRatio appears in the denominator of the delayAsymmetry term because delayAsymmetry, which is supplied via the managed object portDS.delayAsymmetry or cmlDsLinkPort.delayAsymmetry, is obtained via measurements outside of gPTP and is relative to the grandmaster time base. delayAsymmetry is converted to the local clock time base based on the respective method used to measure neighbor rate ratio.

NOTE 6—This represents a media-dependent modification to the calculation given in 10.2.2.7.

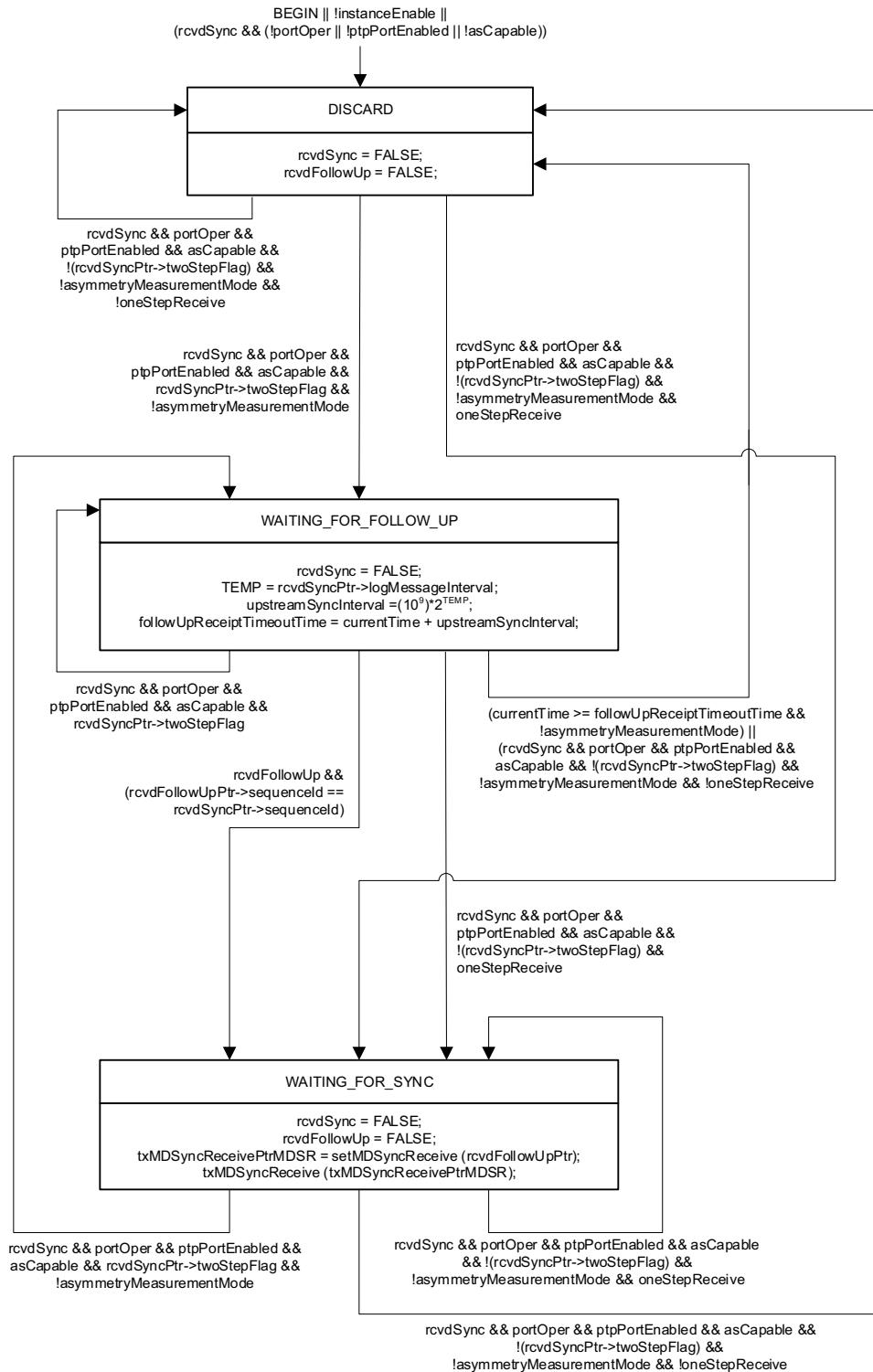
NOTE 7—The difference between the mean propagation time in the Grandmaster Clock time base, the time base of the PTP Instance at the other end of the PTP Link, and the time base of the current PTP Instance is usually negligible. The same is true of any delayAsymmetry (see NOTE 2 of 11.2.19.3.4).

- j) gmTimeBaseIndicator is set equal to the gmTimeBaseIndicator of the most recently received Follow_Up information TLV (see 11.4.4.3.7).
- k) lastGmPhaseChange is set equal to the lastGmPhaseChange of the most recently received Follow_Up information TLV (see 11.4.4.3.8).
- l) lastGmFreqChange is set equal to the scaledLastGmFreqChange of the most recently received Follow_Up information TLV (see 11.4.4.3.9), multiplied by 2^{-41} .
- m) domainNumber is set equal to the domainNumber of the most recently received Sync message (see 11.4.3).

11.2.14.2.2 txMDSyncReceive (txMDSyncReceivePtrMDSR): Transmits an MDSyncReceive structure to the PortSyncSyncReceive state machine of the PortSync entity of this PTP Port.

11.2.14.3 State diagram

The MDSyncReceiveSM state machine shall implement the function specified by the state diagram in Figure 11-6, the local variables specified in 11.2.14.1, the functions specified in 11.2.14.2, the structure specified in 10.2.2.2, the messages specified in 11.4, the MD entity global variables specified in 11.2.13, and the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives Sync and, if the received information is two-step, Follow_Up messages, places the time-synchronization information in an MDSyncReceive structure, and sends the structure to the PortSyncSyncReceive state machine of the PortSync entity of this PTP Port.

**Figure 11-6—MDSyncReceiveSM state machine**

11.2.15 MDSyncSendSM state machine

11.2.15.1 State machine variables

The following variables are used in the state diagram in Figure 11-7 (in 11.2.15.3):

11.2.15.1.1 rcvdMDSyncMDSS: A Boolean variable that notifies the current state machine when an MDSyncSend structure is received. This variable is reset by the current state machine.

11.2.15.1.2 txSyncPtr: A pointer to a structure whose members contain the values of the fields of a Sync message to be transmitted.

11.2.15.1.3 rcvdMDTimestampReceiveMDSS: A Boolean variable that notifies the current state machine when the syncEventEgressTimestamp (see 11.3.2.1) for a transmitted Sync message is received. This variable is reset by the current state machine.

11.2.15.1.4 rcvdMDTimestampReceivePtr: A pointer to the received MDTimestampReceive structure (see 11.2.9).

11.2.15.1.5 txFollowUpPtr: A pointer to a structure whose members contain the values of the fields of a Follow_Up message to be transmitted.

11.2.15.2 State machine functions

11.2.15.2.1 setSyncTwoStep(): Creates a structure whose parameters contain the fields (see 11.4) of a Sync message to be transmitted, and returns a pointer to this structure. The parameters are set as follows:

- a) correctionField is set equal to 0.
- b) sourcePortIdentity is set equal to the sourcePortIdentity member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- c) sequenceId is set equal to syncSequenceId (see 11.2.13.8).
- d) logMessageInterval is set equal to the logMessageInterval member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- e) The remaining fields are set as specified in 11.4.2 and 11.4.3 when twoStepFlag is TRUE.

11.2.15.2.2 txSync (txSyncPtr): Transmits a Sync message from this MD entity, whose fields contain the parameters in the structure pointed to by txSyncPtr (see 11.2.15.1.2).

11.2.15.2.3 setFollowUp(): Creates a structure whose parameters contain the fields (see 11.4) of a Follow_Up message to be transmitted, and returns a pointer to this structure. The parameters are set as follows:

- a) followUpCorrectionField is set equal to the sum of the following:
 - 1) The followUpCorrectionField member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11)
 - 2) The quantity

$$\text{rateRatio} \times (\text{syncEventEgressTimestamp} - \text{upstreamTxTime})$$

where

rateRatio is the rateRatio member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11)

upstreamTxTime is the upstreamTxTime member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11)

syncEventEgressTimestamp is the timestamp pointed to by rcvdMDTimestampReceivePtr, corrected for egressLatency (see 8.4.3)

NOTE—If the PTP Instance that contains this PortSync entity is a PTP Relay Instance, the quantity $\text{syncEventEgressTimestamp} - \text{upstreamTxTime}$ is the sum of the residence time and PTP Link propagation delay on the upstream PTP Link, relative to the LocalClock entity, and the quantity $\text{rateRatio} \times (\text{syncEventEgressTimestamp} - \text{upstreamTxTime})$ is the sum of the residence time and PTP Link propagation delay on the upstream PTP Link, relative to the Grandmaster Clock (see 11.2.14.2.1 for details on the setting of upstreamTxTime).

- b) sourcePortIdentity is set equal to the sourcePortIdentity member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- c) sequenceId is set equal to syncSequenceId (see 11.2.13.8).
- d) logMessageInterval is set equal to the logMessageInterval member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- e) preciseOriginTimestamp is set equal to the preciseOriginTimestamp member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- f) rateRatio is set equal to the rateRatio member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- g) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the PTP Instance is the Grandmaster PTP Instance, a Drift_Tracking TLV is attached to the transmitted Follow_Up message with the syncGrandmasterIdentity, syncStepsRemoved, syncEgressTimestamp, and rateRatioDrift fields set as follows:
 - 1) syncGrandmasterIdentity is set equal to the defaultDS.clockIdentity.
 - 2) syncStepsRemoved is set equal to 0.
 - 3) syncEgressTimestamp is set equal to the syncEventEgressTimestamp of the associated Sync message.
 - 4) rateRatioDrift (see 11.4.4.4.9) is the measured estimate of the rate of change per second of the frequency of the Grandmaster Clock relative to the frequency of the local clock at the Grandmaster PTP Instance. If the Grandmaster Clock and the local clock at the Grandmaster PTP Instance are the same, rateRatioDrift is zero.
- h) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the PTP Instance is not the Grandmaster PTP Instance, a Drift_Tracking TLV is attached to the transmitted Follow_Up message with the syncGrandmasterIdentity, syncStepsRemoved, syncEgressTimestamp, and rateRatioDrift fields set as follows:
 - 1) syncGrandmasterIdentity is set equal to the value of the global variable syncGrandmasterIdentity (see 10.2.4.26).
 - 2) syncStepsRemoved is set equal to the value of the global variable syncStepsRemoved (see 10.2.4.27).
 - 3) syncEgressTimestamp is set equal to the syncEventEgressTimestamp of the associated Sync message.
 - 4) rateRatioDrift (see 11.4.4.4.9) is the measured estimate of the rate of change per second of the frequency of the Grandmaster Clock relative to the frequency of the local clock of the PTP Instance that sends the message.
- i) gmTimeBaseIndicator is set equal to the gmTimeBaseIndicator member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- j) lastGmPhaseChange is set equal to the lastGmPhaseChange member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- k) scaledLastGmFreqChange is set equal to the lastGmFreqChange member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11), multiplied by 2^{41} .

- i) domainNumber is set equal to the domainNumber of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- m) The remaining fields are set as specified in 11.4.2 and 11.4.4.

11.2.15.2.4 txFollowUp (txFollowUpPtr): Transmits a Follow_Up message from this MD entity, whose fields contain the parameters in the structure pointed to by txFollowUpPtr (see 11.2.15.1.5).

11.2.15.2.5 setSyncOneStep(): Creates a structure whose parameters contain the fields (see 11.4) of a Sync message to be transmitted, and returns a pointer to this structure. The parameters are set as follows:

- a) correctionField is set equal to the followUpCorrectionField member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- b) sourcePortIdentity is set equal to the sourcePortIdentity member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- c) sequenceId is set equal to syncSequenceId (see 11.2.13.8).
- d) logMessageInterval is set equal to the logMessageInterval member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- e) originTimestamp is set equal to the preciseOriginTimestamp member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- f) rateRatio is set equal to the rateRatio member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- g) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the PTP Instance is the Grandmaster PTP Instance, a Drift_Tracking TLV is attached to the transmitted Sync message with the syncGrandmasterIdentity, syncStepsRemoved, syncEgressTimestamp, and rateRatioDrift fields set as follows:
 - 1) syncGrandmasterIdentity is set equal to the defaultDS.clockIdentity.
 - 2) syncStepsRemoved is set equal to 0.
 - 3) syncEgressTimestamp is set equal to the syncEventEgressTimestamp of the associated Sync message.
 - 4) rateRatioDrift (see 11.4.4.4.9) is the measured estimate of the rate of change per second of the frequency of the Grandmaster Clock relative to the frequency of the local clock at the Grandmaster PTP Instance. If the Grandmaster Clock and the local clock at the Grandmaster PTP Instance are the same, rateRatioDrift is zero.
- h) If the value of the global variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and the PTP Instance is not the Grandmaster PTP Instance, a Drift_Tracking TLV is attached to the transmitted Sync message with the syncGrandmasterIdentity, syncStepsRemoved, syncEgressTimestamp, and rateRatioDrift fields set as follows:
 - 1) syncGrandmasterIdentity is set equal to the value of the global variable syncGrandmasterIdentity (see 10.2.4.26).
 - 2) syncStepsRemoved is set equal to the value of the global variable syncStepsRemoved (see 10.2.4.27).
 - 3) syncEgressTimestamp is set equal to the syncEventEgressTimestamp of the associated Sync message.
 - 4) rateRatioDrift (see 11.4.4.4.9) is the measured estimate of the rate of change per second of the frequency of the Grandmaster Clock relative to the frequency of the local clock of the PTP Instance that sends the message.
- i) gmTimeBaseIndicator is set equal to the gmTimeBaseIndicator member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).

- j) `lastGmPhaseChange` is set equal to the `lastGmPhaseChange` member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- k) `scaledLastGmFreqChange` is set equal to the `lastGmFreqChange` member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11), multiplied by 2^{41} .
- l) `domainNumber` is set equal to the `domainNumber` of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11).
- m) The remaining fields are set as specified in 11.4.2 and 11.4.3 when `twoStepFlag` is FALSE.

11.2.15.2.6 `modifySync()`: Adds to the `correctionField` of the transmitted Sync message, after the `syncEventEgressTimestamp` is taken and known, the quantity

$$\text{rateRatio} \times (\text{syncEventEgressTimestamp} - \text{upstreamTxTime})$$

where

<code>rateRatio</code>	is the <code>rateRatio</code> member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11)
<code>upstreamTxTime</code>	is the <code>upstreamTxTime</code> member of the most recently received MDSyncSend structure (see 10.2.2.1 and 11.2.11)
<code>syncEventEgressTimestamp</code>	is the timestamp pointed to by <code>rcvdMDTimestampReceivePtr</code> , corrected for <code>egressLatency</code> (see 8.4.3)

NOTE—If the PTP Instance that contains this PortSync entity is a PTP Relay Instance, the quantity

$$\text{syncEventEgressTimestamp} - \text{upstreamTxTime}$$

is the sum of the residence time and PTP Link propagation delay on the upstream PTP Link, relative to the LocalClock entity, and the quantity

$$\text{rateRatio} \times (\text{syncEventEgressTimestamp} - \text{upstreamTxTime})$$

is the sum of the residence time and PTP Link propagation delay on the upstream PTP Link, relative to the Grandmaster Clock (see 11.2.14.2.1 for details on the setting of `upstreamTxTime`).

11.2.15.3 State diagram

The MDSyncSendSM state machine shall implement the function specified by the state diagram in Figure 11-7; the local variables specified in 11.2.15.1; the functions specified in 11.2.15.2; the structures specified in 11.2.9 and 10.2.2.1; the messages specified in 11.4; the MD entity global variables specified in 11.2.13; and the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives an MDSyncSend structure from the PortSyncSyncSend state machine of the PortSync entity of this PTP Port and transmits a Sync and corresponding Follow_Up message.

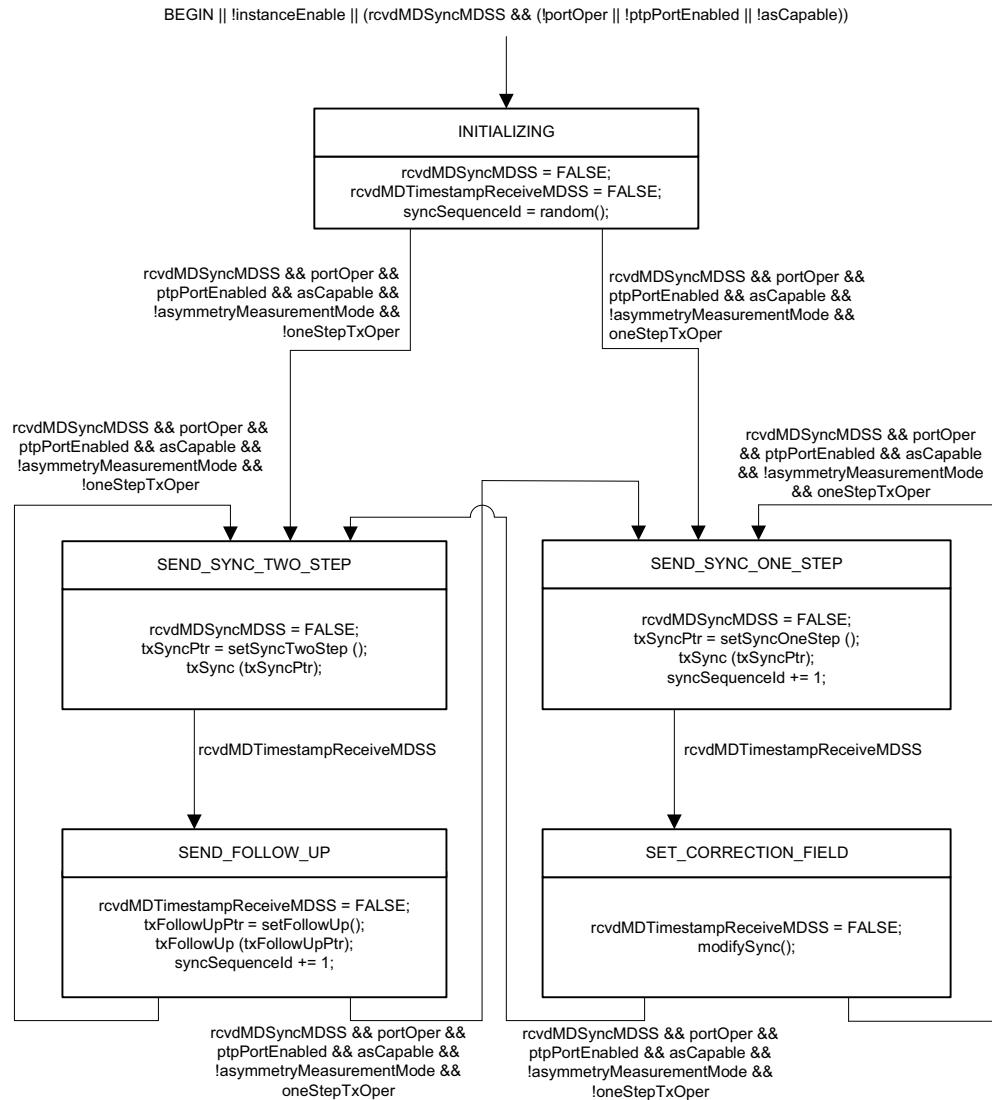


Figure 11-7—MDSyncSendSM state machine

11.2.16 OneStepTxOperSetting state machine

11.2.16.1 State machine variables

The following variables are used in the state diagram in Figure 11-8 (in 11.2.16.2):

11.2.16.1.1 rcvdSignalingMsg4: A Boolean variable that notifies the current state machine when a Signaling message that contains a Message Interval Request TLV (see 10.6.4.3) is received. This variable is reset by the current state machine.

11.2.16.1.2 rcvdSignalingPtrOSTOS: A pointer to a structure whose members contain the values of the fields of the received Signaling message that contains a Message Interval Request TLV (see 10.6.4.3).

11.2.16.2 State diagram

The OneStepTxOperSetting state machine shall implement the function specified by the state diagram in Figure 11-8, the local variables specified in 11.2.16.1, the messages specified in 10.6 and 11.4, the relevant global variables specified in 10.2.5 and 11.2.13, and the relevant managed objects specified in 14.8. This state machine is responsible for setting the relevant global variables and managed objects pertaining to one-step/two-step operation.

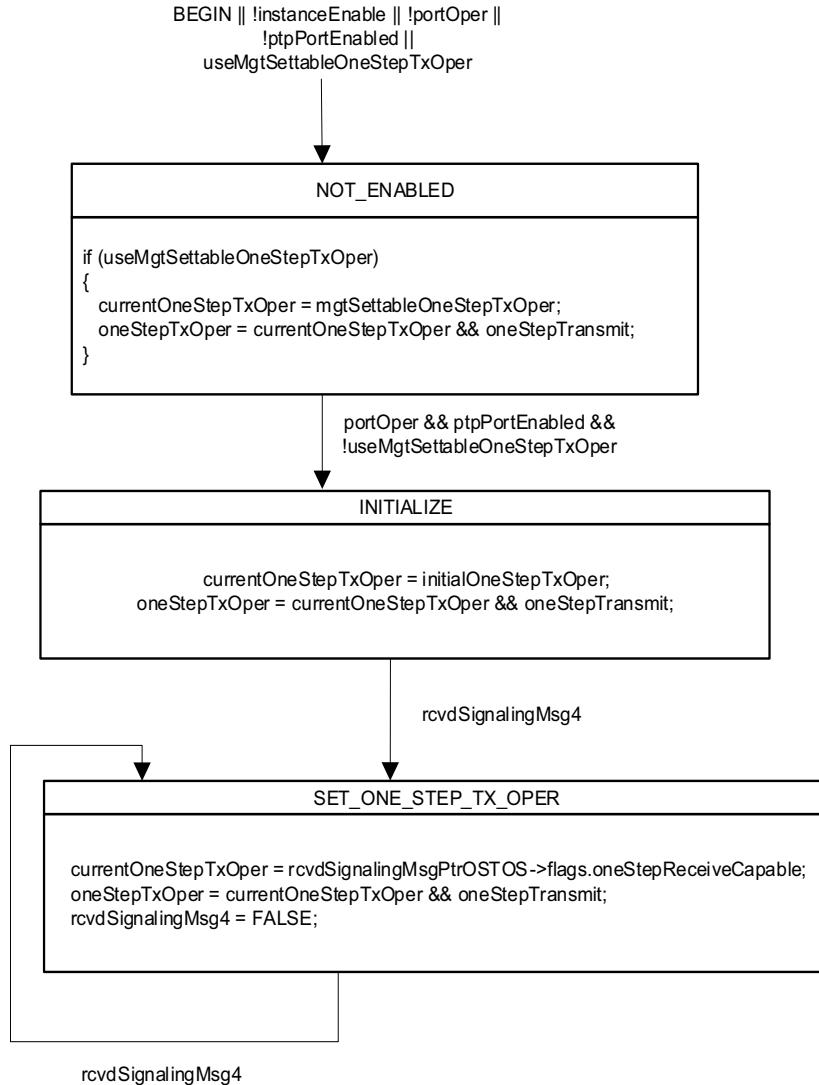


Figure 11-8—OneStepTxOperSetting state machine

11.2.17 Common Mean Link Delay Service (CMLDS)

11.2.17.1 General

Each **PTP Port or Link Port** of a time-aware system invokes a single instance of the MDPdelayReq state machine (see 11.2.19) and the MDPdelayResp state machine (see 11.2.20). If the time-aware system implements more than one domain **or if domainNumber 0 is not present**, these two state machines shall provide a Common Mean Link Delay Service (CMLDS), as described in this subclause, that measures mean propagation delay on the PTP Link attached to the port and the neighbor rate ratio for the port (i.e., the ratio of the frequency of the LocalClock entity of the time-aware system at the other end of the PTP Link attached to this port, to the frequency of the LocalClock entity of this time-aware system). The CMLDS makes the mean propagation delay and neighbor rate ratio available to all active domains. If the time-aware system implements one domain, **and if the domainNumber of this domain is 0 (see 8.1)**, these two state machines may provide the CMLDS; however, if **the state machines do not provide the CMLDS (i.e., if only the transport-specific peer delay mechanism is provided)**, they shall be invoked on domain 0. In other words, if the **domainNumber is not 0, portDS.delayMechanism (see Table 14-8 in 14.8.5) must not be P2P. If CMLDS is used, the LocalClock entity for CMLDS and the LocalClock entity for each PTP Instance shall be the same LocalClock.**

NOTE 1—In the above, the **case where** the time-aware system implements only one domain implicitly assumes that IEEE 802.1AS is the only PTP profile present on the respective port of the time-aware system, i.e., no other PTP profiles are implemented on that port. If another PTP profile (see IEEE Std 1588) besides IEEE Std 802.1AS is active on the port, the **CMLDS provides the mean propagation delay and neighbor rate ratio to all of them.**

In accordance with IEEE Std 1588-2019, the term *Link Port* refers to a port of the CMLDS. A PTP Port for which **portDS.delayMechanism is COMMON_P2P** uses the CMLDS provided by the *Link Port* whose **cmlsLinkPortDS.portIdentity.portNumber (see 14.16.2)** is equal to the **commonServicesPortDS.cmlsLinkPortPortNumber (see 14.14.2)** for this PTP Port.

The value of majorSdoId for the CMLDS shall be 0x2. The value of minorSdoId for the Common Mean Link Delay Service shall be 0x00. As a result, the value of sdoId for the Common Mean Link Delay Service is 0x200.

NOTE 2—The above requirements for majorSdoId and minorSdoId are for the CMLDS. The requirements for gPTP domains, including **transport-specific peer delay messages**, are given in 8.1.

NOTE 3—The requirements of this subclause for CMLDS are consistent with the requirements of IEEE Std 1588-2019.

If a PTP Port **on a physical interface** that **also supports CMLDS and a Link Port** receives a **Pdelay_Req** message with majorSdoId value of 0x1, minorSdoId value of 0x00, and domainNumber value of 0, the PTP Port shall respond with PTP **transport-specific peer delay messages** (i.e., the **Pdelay_Resp** and **Pdelay_Resp_Follow_Up** corresponding to this **Pdelay_Req**) using the **transport-specific peer-to-peer delay mechanism if domain 0 is enabled**. These **transport-specific messages** have majorSdoId value of 0x1, minorSdoId value of 0x00, and domainNumber value of 0.

NOTE 4—The above requirement ensures:

- a) Backward compatibility with time-aware systems that comply with the 2011 version of this standard and
- b) Compatibility with time-aware systems that implement only one domain and invoke the MDPdelayReq and MDPdelayResp state machines on domain 0.

NOTE 5—In general, a port can receive:

- a) Peer delay messages of the CMLDS (**with sdoId of 0x200**),
- b) **Transport-specific peer delay messages of domain 0 (with sdoId of 0x100)**, and
- c) If there are other PTP profiles on the neighbor port that use **transport-specific peer delay**, peer delay messages of those profiles (i.e., with an **sdoId of neither 0x100 nor 0x200**).

The port responds to the messages of type a) if it invokes CMLDS, the messages of type b) if it invokes gPTP domain 0, and the messages of type c) if it invokes the respective other PTP profiles.

The CMLDS shall be enabled on a Link Port if the value of portDS.delayMechanism (see 14.8.5) is COMMON_P2P for at least one PTP Port that is enabled (i.e., for which portOper and ptPortEnabled are both TRUE) and corresponds to the same physical port as the Link Port (see 14.1). The value of cmldsLinkPortEnabled is TRUE if the CMLDS is enabled on the Link Port and FALSE if the CMLDS is not enabled on the Link Port.

11.2.17.2 Differences between transport-specific peer-to-peer delay mechanism and CMLDS in computations of meanLinkDelay and nrPdelay

The MDPdelayReq state machine (see 11.2.19), and the MDPdelay_Resp state machine (see 11.2.20), are invoked either by the transport-specific peer-to-peer delay mechanism or by CMLDS. The resulting values of meanLinkDelay and nrPdelay are the same for both transport-specific peer delay and CMLDS; however, the organization of the computations for transport-specific peer delay and CMLDS are different. Some of the computations are done at the Initiator and some of the computations are done at the Responder. It is necessary for the Initiator and the Responder to both perform the transport-specific peer delay computations or both perform the CMLDS computations in order to obtain the correct results for meanLinkDelay and nrPdelay. The differences are:

- a) Both transport-specific peer delay and CMLDS compute meanLinkDelay averaged over the two directions and adds delayAsymmetry separately when computing upstreamTxTime by the setMDSyncReceiveMDSR() function of the MDSyncReceive state machine. However, in the computation of meanLinkDelay, CMLDS subtracts delayAsymmetry from the correctionField when sending the pdelayReq message at the Initiator (see Figure 11-1) and adds delayAsymmetry back when computing the quantity t_3 in the function computePropTime() (at the Initiator), while transport-specific peer delay does not subtract and add back delayAsymmetry.
- b) Transport-specific peer delay sets the correctionField of a transmitted Pdelay_Req message to 0, while CMLDS sets it to -delayAsymmetry (see 11.2.19.3.1).
- c) Transport-specific peer delay sets the correctionField of Pdelay_Resp equal to the fractional nanoseconds portion of the pdelayReqEventIngressTimestamp of the corresponding Pdelay_Req, while CMLDS sets the correctionField of Pdelay_Resp equal to minus the fractional nanoseconds portion of the pdelayReqEventIngressTimestamp of the corresponding Pdelay_Req (see 11.2.20.3.1).
- d) Transport-specific peer delay sets the correctionField of Pdelay_Resp_Follow_Up equal to the fractional nanoseconds portion of the pdelayRespEventEgressTimestamp, while CMLDS sets the correctionField of Pdelay_Resp_Follow_Up equal to the sum of the correctionField of the corresponding Pdelay_Req and the fractional nanoseconds portion of the pdelayRespEventEgressTimestamp (see 11.2.20.3.3).
- e) When computing the quantity t_2 for the meanLinkDelay computation [i.e., the quantity D in Equation (11-5), see 11.2.19.3.4], the correctionField of the Pdelay_Resp message, divided by 2^{16} , is added if transport-specific peer delay is used, while it is subtracted if CMLDS is used (see 11.2.19.3.4), and the quantity -delayAsymmetry, divided by 2^{16} , is added if CMLDS is used (i.e., delayAsymmetry is subtracted in the case of CMLDS).
- f) When computing nrPdelay, the computation of the correctedResponderEventTimestamp is corrected for delayAsymmetry if, and only if, CMLDS is used. The reason for this correction is that, with CMLDS, delayAsymmetry was subtracted from the Pdelay_Req correctionField, and then the Pdelay_Resp_Follow_Up correctionField was set equal to the sum of the Pdelay_Req correctionField and the fractional nanoseconds portion of the PdelayRespEventEgressTimestamp, while with transport-specific peer delay, the correctionField of Pdelay_Req was set equal to 0 (see 11.2.19.3.1, 11.2.19.3.3, and 11.2.20.3.3).

The computations in this standard for the **transport**-specific peer-to-peer delay mechanism are the same as in IEEE Std 802.1AS-2011, for backward compatibility. However, the computations in this standard for CMLDS **needs to** be consistent with IEEE Std 1588-2019 because CMLDS can be used by other PTP profiles, in addition to the PTP profile included in IEEE Std 802.1AS, that might be present in a gPTP node. Therefore, the computations for the **transport**-specific peer-to-peer delay mechanism and CMLDS are different (i.e., are organized differently), even though they produce the same results.

11.2.18 Common Mean Link Delay Service (CMLDS) global variables

11.2.18.1 cmldsLinkPortEnabled: A per-Link-Port Boolean that is TRUE if both the value of portDS.delayMechanism is Common_P2P and the value of portDS.ptpPortEnabled is TRUE, for at least one PTP Port that uses the CMLDS that is invoked on the Link Port; otherwise, the value is FALSE.

11.2.19 MDPdelayReq state machine

11.2.19.1 General

This state machine is either invoked as part of the Common Mean Link Delay Service (CMLDS) or by the transport-specific peer-to-peer delay mechanism. There is one instance of this state machine for all the domains (per port). As a result, there also is one instance of each of the state machine variables of 11.2.19.2, state machine functions of 11.2.19.3, and relevant global variables of 10.2.5, 11.2.13, and 11.2.18 for all the domains (per port). None of the variables used or functions invoked in this state machine are specific to a single domain. However, the single instances of all of these objects or entities are accessible to all the domains.

NOTE—This state machine uses the variable `asCapableAcrossDomains` (see 11.2.13.12). When only one domain is active, `asCapableAcrossDomains` is equivalent to the variable `asCapable`.

11.2.19.2 State machine variables

The following variables are used in the state diagram in Figure 11-9 (in 11.2.19.4):

11.2.19.2.1 pdelayIntervalTimer: A variable used to save the time at which the `Pdelay_Req` interval timer is started. A `Pdelay_Req` message is sent when this timer expires. The data type for `pdelayIntervalTimer` is `UScaledNs`.

11.2.19.2.2 rcvdPdelayResp: A Boolean variable that notifies the current state machine when a `Pdelay_Resp` message is received. This variable is reset by the current state machine.

11.2.19.2.3 rcvdPdelayRespPtr: A pointer to a structure whose members contain the values of the fields of the `Pdelay_Resp` message whose receipt is indicated by `rcvdPdelayResp` (see 11.2.19.2.2).

11.2.19.2.4 rcvdPdelayRespFollowUp: A Boolean variable that notifies the current state machine when a `Pdelay_Resp_Follow_Up` message is received. This variable is reset by the current state machine.

11.2.19.2.5 rcvdPdelayRespFollowUpPtr: A pointer to a structure whose members contain the values of the fields of the `Pdelay_Resp_Follow_Up` message whose receipt is indicated by `rcvdPdelayRespFollowUp` (see 11.2.19.2.4).

11.2.19.2.6 txPdelayReqPtr: A pointer to a structure whose members contain the values of the fields of a `Pdelay_Req` message to be transmitted.

11.2.19.2.7 rcvdMDTimestampReceiveMDPReq: A Boolean variable that notifies the current state machine when the `pdelayReqEventEgressTimestamp` (see 11.3.2.1) for a transmitted `Pdelay_Req` message is received. This variable is reset by the current state machine.

11.2.19.2.8 pdelayReqSequenceId: A variable that holds the `sequenceId` for the next `Pdelay_Req` message to be transmitted by this MD entity. The data type for `pdelayReqSequenceId` is `UInteger16`.

11.2.19.2.9 lostResponses: A count of the number of consecutive `Pdelay_Req` messages sent by the port, for which `Pdelay_Resp` and/or `Pdelay_Resp_Follow_Up` messages are not received. The data type for `lostResponses` is `UInteger16`.

11.2.19.2.10 nrrPdelayValid: A Boolean variable that indicates whether the function `computePdelayRateRatio()` (see 11.2.19.3.3) successfully computed `nrrPdelay` (see 11.2.13.13). The default value of `nrrPdelayValid` is TRUE.

11.2.19.2.11 detectedFaults: A count of the number of consecutive faults (see 11.5.4 for the definition of a fault).

11.2.19.2.12 portEnabled0: A Boolean variable whose value is equal to ptPPortEnabled (see 10.2.5.13) if this state machine is invoked by the *transport*-specific peer-to-peer delay mechanism and is equal to cmldsLinkPortEnabled (see 11.2.18.1) if this state machine is invoked by CMLDS.

11.2.19.2.13 s: A variable whose value is +1 if this state machine is invoked by the *transport*-specific peer-to-peer delay mechanism and -1 if this state machine is invoked by CMLDS. The data type for s is Integer8.

11.2.19.3 State machine functions

11.2.19.3.1 setPdelayReq(): Creates a structure containing the parameters (see 11.4) of a Pdelay_Req message to be transmitted, and returns a pointer, txPdelayReqPtr (see 11.2.19.2.6), to this structure. The parameters are set as follows:

- a) sourcePortIdentity is set equal to the port identity of the port corresponding to this MD entity (see 8.5.2).
- b) sequenceId is set equal to pdelayReqSequenceId (see 11.2.19.2.8).
- c) correctionField is set to
 - 1) 0 if this state machine is invoked by the *transport*-specific peer-to-peer delay mechanism, and
 - 2) -delayAsymmetry (i.e., the negative of delayAsymmetry) if this state machine is invoked by the CMLDS.
- d) The remaining parameters are set as specified in 11.4.2 and 11.4.5.

11.2.19.3.2 txPdelayReq(txPdelayReqPtr): Transmits a Pdelay_Req message from the MD entity, containing the parameters in the structure pointed to by txPdelayReqPtr (see 11.2.19.2.6).

11.2.19.3.3 computePdelayRateRatio(): Computes *nrrPdelay* (see 11.2.13.13) using the following information conveyed by successive Pdelay_Resp and Pdelay_Resp_Follow_Up messages:

- a) The pdelayRespEventIngressTimestamp (see 11.3.2.1) values for the respective Pdelay_Resp messages
- b) The correctedResponderEventTimestamp values, whose data type is UScaledNs, obtained by adding the following fields of the received Pdelay_Resp_Follow_Up message:
 - 1) The seconds field of the responseOriginTimestamp field, multiplied by 10^9
 - 2) The nanoseconds field of the responseOriginTimestamp field
 - 3) The correctionField, divided by 2^{16}
 - 4) delayAsymmetry (see 10.2.5.9), if and only if this state machine is invoked by CMLDS

NOTE 1—If delayAsymmetry does not change during the time interval over which *nrrPdelay* is computed, it is not necessary to add it if this state machine is invoked by CMLDS because in that case it will be canceled when computing the difference between earlier and later correctedResponderEventTimestamps.

Any scheme that uses the preceding information, along with any other information conveyed by the successive Pdelay_Resp and Pdelay_Resp_Follow_Up messages, to compute *nrrPdelay* is acceptable as long as the performance requirements specified in B.2.4 are met. If *nrrPdelay* is successfully computed, the Boolean *nrrPdelayValid* (see 11.2.19.2.10) is set to TRUE. If *nrrPdelay* is not successfully computed (e.g., if the MD entity has not yet exchanged a sufficient number of peer delay messages with its peer), the Boolean *nrrPdelayValid* is set to FALSE.

NOTE 2—As one example, `nrrPdelay` can be estimated as the ratio of the elapsed time of the LocalClock entity of the time-aware system at the other end of the PTP Link attached to this port, to the elapsed time of the LocalClock entity of this time-aware system. This ratio can be computed for the time interval between a set of received `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages and a second set of received `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages some number of `Pdelay_Req` message transmission intervals later, i.e.,

$$\frac{\text{correctedResponderEventTimestamp}_N - \text{correctedResponderEventTimestamp}_0}{\text{pdelayRespEventIngressTimestamp}_N - \text{pdelayRespEventIngressTimestamp}_0}$$

where N is the number of `Pdelay_Req` message transmission intervals separating the first set of received `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages and the second set, and the successive sets of received `Pdelay_Resp` and `Pdelay_Resp_Follow_Up` messages are indexed from 0 to N with the first set indexed 0.

NOTE 3—This function must account for non-receipt of `Pdelay_Resp` and/or `Pdelay_Resp_Follow_Up` for a `Pdelay_Req` message and also for receipt of multiple `Pdelay_Resp` messages within one `Pdelay_Req` message transmission interval.

11.2.19.3.4 computePropTime(): Computes the mean propagation delay on the PTP Link attached to this MD entity, D , and returns this value. D is given by Equation (11-5).

$$D = \frac{r \cdot (t_4 - t_1) - (t_3 - t_2)}{2} \quad (11-5)$$

where

- t_4 is `pdelayRespEventIngressTimestamp` (see 11.3.2.1) for the `Pdelay_Resp` message received in response to the `Pdelay_Req` message sent by the MD entity, in nanoseconds; the `pdelayRespEventIngressTimestamp` is equal to the timestamp value measured relative to the timestamp measurement plane, minus any `ingressLatency` (see 8.4.3)
- t_1 is `pdelayReqEventEgressTimestamp` (see 11.3.2.1) for the `Pdelay_Req` message sent by the P2PPort entity, in nanoseconds
- t_2 is the sum of (1) the ns field of the `requestReceiptTimestamp`, (2) the seconds field of the `requestReceiptTimestamp` multiplied by 10^9 , (3) the `correctionField` multiplied by s (see 11.2.19.2.13) and then divided by 2^{16} (i.e., the `correctionField` is expressed in nanoseconds plus fractional nanoseconds), of the `Pdelay_Resp` message received in response to the `Pdelay_Req` message sent by the MD entity, and (4) `delayAsymmetry` divided by 2^{16} if this state machine is invoked by CMLDS (i.e., `delayAsymmetry` is subtracted in the case of CMLDS)
- t_3 is the sum of (1) the ns field of the `responseOriginTimestamp`, (2) the seconds field of the `responseOriginTimestamp` multiplied by 10^9 , and (3) the `correctionField` divided by 2^{16} (i.e., the `correctionField` is expressed in nanoseconds plus fractional nanoseconds), of the `Pdelay_Resp_Follow_Up` message received in response to the `Pdelay_Req` message sent by the MD entity
- r is the current value of `nrrPdelay` for this MD entity (see 11.2.13.13)

When CMLDS is used, Equation (11-5) can be rewritten as shown in Equation (11-6) (see NOTE 3 below) using the definitions of t_2 and t_3 above for the CMLDS case.

Propagation delay averaging may be performed, as described in 11.1.2 by Equation (11-2), Equation (11-3), and Equation (11-4). In this case, the successive values of propagation delay computed using Equation (11-5) are input to either Equation (11-2) or Equation (11-4), and the computed average propagation delay is returned by this function.

NOTE 1—Equation (11-5) defines D as the mean propagation delay relative to the time base of the time-aware system at the other end of the attached PTP Link. It is divided by `nrrPdelay` (see 11.2.13.13) to convert it to the time base of the current time-aware system when subtracting from `syncEventIngressTimestamp` to compute `upstreamTxTime` [see item i in 11.2.14.2.1].

NOTE 2—The difference between mean propagation delay relative to the Grandmaster Clock time base and relative to the time base of the time-aware system at the other end of the attached PTP Link is usually negligible. To see this, note that the former can be obtained from the latter by multiplying the latter by the ratio of the Grandmaster Clock frequency to the frequency of the LocalClock entity of the time-aware system at the other end of the PTP Link attached to this port. This ratio differs from 1 by 200 ppm or less. For example, for a worst-case frequency offset of the LocalClock entity of the time-aware system at the other end of the PTP Link, relative to the Grandmaster Clock, of 200 ppm, and a measured propagation time of 100 ns, the difference in D relative to the two time bases is 20 ps.

NOTE 3—In IEEE Std 1588-2019, the computation of `meanLinkDelay` is organized differently from the organization used for `transport-specific peer delay` in the present standard. Using the definitions of t_2 and t_3 above, Equation (11-5) can be rewritten as shown in Equation (11-6).

$$D = [r \cdot (t_4 - t_1) - (\text{responseOriginTimestamp} - \text{requestReceiptTimestamp}) + s \times (\text{correctionField of Pdelay_Resp} - \text{correctionField of Pdelay_Resp_Follow_Up}) - \text{delayAsymmetry}] / 2 \quad (11-6)$$

where each term is expressed in units of nanoseconds as described in the definitions of t_1 , t_2 , t_3 , and t_4 above. In IEEE Std 1588-2019, the fractional nanoseconds portion of t_2 is subtracted from the correctionField of Pdelay_Resp, rather than added as in the present standard for transport-specific peer delay [see 11.2.20.3.1 d) 1)]; however, the correctionField of Pdelay_Resp is then subtracted in Equation (11-6) rather than added [in Equation (11-6) for transport-specific peer delay, where $s=1$], and the two minus signs result in the same sign for the fractional nanoseconds portion of t_2 and the seconds and nanoseconds portion of t_2 (`requestReceiptTimestamp`). The computations of D in this standard and IEEE Std 1588-2019 are mathematically equivalent. The organization of the computation with CMLDS needs to be used in the present standard for interoperability with IEEE Std 1588-2019 (see 11.2.17.2). This organization in the present standard, for the case where CMLDS is used, is consistent with the organization in IEEE Std 1588-2019.

11.2.19.4 State diagram

The MDPdelayReq state machine shall implement the function specified by the state diagram in Figure 11-9, the local variables specified in 11.2.19.2, the functions specified in 11.2.19.3, the messages specified in 11.4, and the relevant global variables and functions specified in 11.2.13 and 11.2.18 and 10.2.4 through 10.2.6. This state machine is responsible for the following:

- a) Sending Pdelay_Req messages and restarting the pdelayIntervalTimer.
- b) Detecting that the peer mechanism is running.
- c) Detecting if Pdelay_Resp and/or Pdelay_Resp_Follow_Up messages corresponding to a Pdelay_Req message sent are not received.
- d) Detecting whether more than one Pdelay_Resp is received within one Pdelay_Req message transmission interval (see 11.5.2.2).
- e) Computing propagation time on the attached PTP Link when Pdelay_Resp and Pdelay_Resp_Follow_Up messages are received.
- f) Computing the ratio of the frequency of the LocalClock entity of the time-aware system at the other end of the attached PTP Link to the frequency of the LocalClock entity of the current time-aware system.

NOTE 1—The ratio of the frequency of the LocalClock entity of the time-aware system at the other end of the attached PTP Link to the frequency of the LocalClock entity of the current time-aware system, `nrrPdelay`, retains its most recent value when a Pdelay_Resp and/or Pdelay_Resp_Follow_Up message is lost.

NOTE 2—Normally, Pdelay_Resp should be received within a time interval after sending Pdelay_Req that is less than or equal to the Pdelay turnaround time plus twice the mean propagation delay. However, while receiving Pdelay_Resp after a time interval that is longer than this can result in worse time-synchronization performance (see 11.1.2 and B.2.3), the peer delay protocol will still operate. It is expected that a peer delay initiator can receive and process Pdelay_Resp and Pdelay_Resp_Follow_Up messages within a time interval after sending Pdelay_Req that is as large as the current Pdelay Req message transmission interval (see 11.5.2.2).

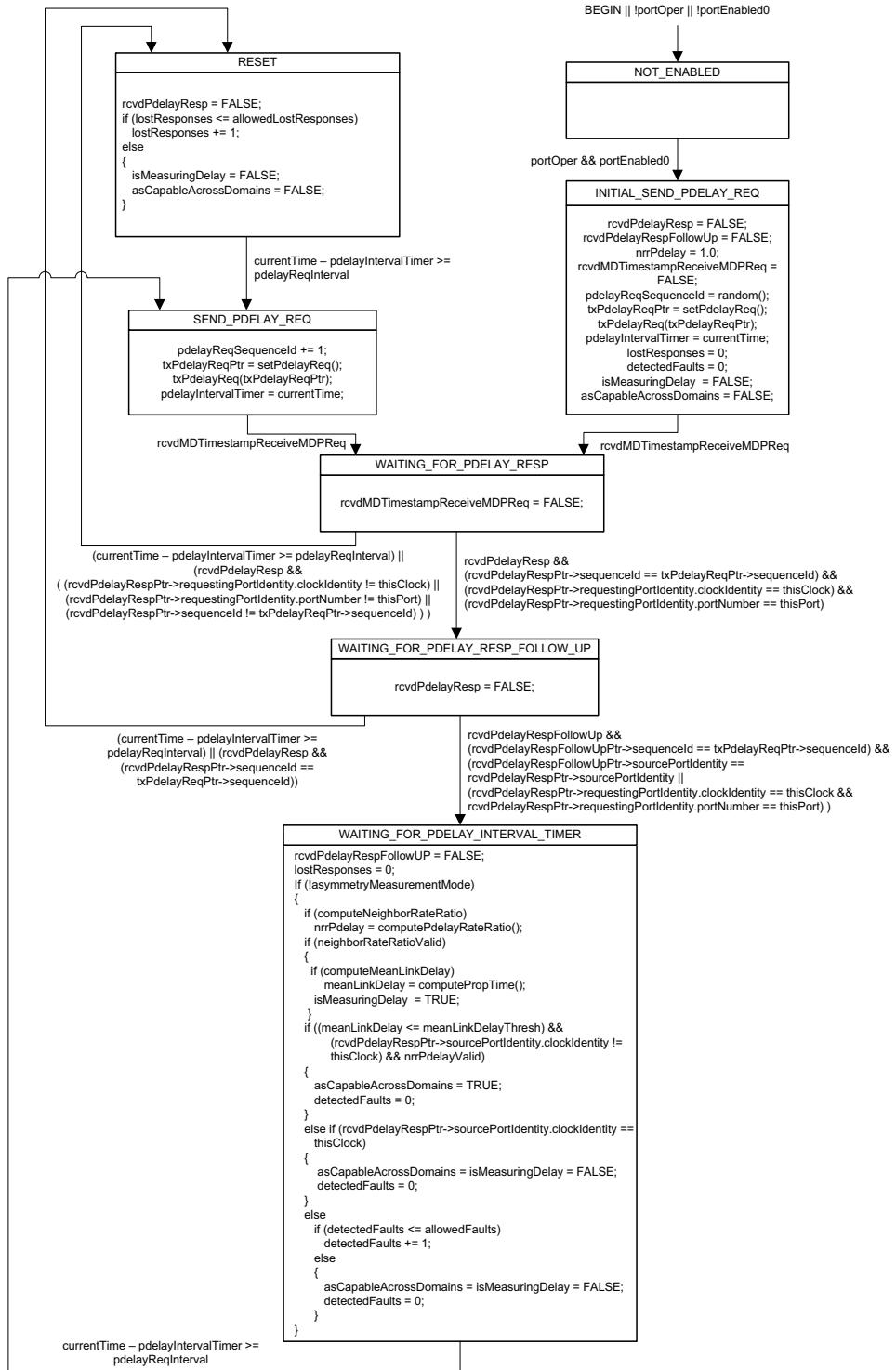


Figure 11-9—MDPdelayReq state machine

11.2.20 MDPdelayResp state machine

11.2.20.1 General

This state machine is either invoked as part of the Common Mean Link Delay Service (CMLDS) or by the transport-specific peer-to-peer delay mechanism. There is one instance of this state machine for all the domains (per port). As a result, there also is one instance of each of the state machine variables of 11.2.20.2, state machine functions of 11.2.20.3, and relevant global variables of 10.2.5, 11.2.13, and 11.2.18 for all the domains (per port). None of the variables used or functions invoked in this state machine are specific to a single domain. However, the single instances of all of these objects or entities are accessible to all the domains.

11.2.20.2 State machine variables

The following variables are used in the state diagram in Figure 11-10 (in 11.2.20.4):

11.2.20.2.1 rcvdPdelayReq: A Boolean variable that notifies the current state machine when a Pdelay_Req message is received. This variable is reset by the current state machine.

11.2.20.2.2 rcvdMDTimestampReceiveMDPResp: A Boolean variable that notifies the current state machine when the pdelayRespEventEgressTimestamp (see 11.3.2.1) for a transmitted Pdelay_Resp message is received. This variable is reset by the current state machine.

11.2.20.2.3 txPdelayRespPtr: A pointer to a structure whose members contain the values of the fields of a Pdelay_Resp message to be transmitted.

11.2.20.2.4 txPdelayRespFollowUpPtr: A pointer to a structure whose members contain the values of the fields of a Pdelay_Resp_Follow_Up message to be transmitted.

11.2.20.2.5 portEnabled1: A Boolean variable whose value is equal to ptPortEnabled (see 10.2.5.13) if this state machine is invoked by the transport-specific peer-to-peer delay mechanism and is equal to cmldsLinkPortEnabled (see 11.2.18.1) if this state machine is invoked by the CMLDS.

11.2.20.3 State machine functions

11.2.20.3.1 setPdelayResp(): Creates a structure containing the parameters (see 11.4) of a Pdelay_Resp message to be transmitted, and returns a pointer, txPdelayRespPtr (see 11.2.20.2.3), to this structure. The parameters are set as follows:

- a) sourcePortIdentity is set equal to the port identity of the port corresponding to this MD entity (see 8.5.2).
- b) sequenceId is set equal to the sequenceId field of the corresponding Pdelay_Req message.
- c) requestReceiptTimestamp is set equal to the pdelayReqEventIngressTimestamp (see 11.3.2) of the corresponding Pdelay_Req message, with any fractional nanoseconds portion truncated.
- d) correctionField is set equal to the following:
 - 1) The fractional nanoseconds portion of the pdelayReqEventIngressTimestamp of the corresponding Pdelay_Req message if this state machine is invoked by the transport-specific peer-to-peer delay mechanism and
 - 2) Minus the fractional nanoseconds portion of the pdelayReqEventIngressTimestamp of the corresponding Pdelay_Req message if this state machine is invoked by CMLDS.
- e) requestingPortIdentity is set equal to the sourcePortIdentity field of the corresponding Pdelay_Req message.
- f) The remaining parameters are set as specified in 11.4.2 and 11.4.6.

11.2.20.3.2 txPdelayResp(txPdelayRespPtr): Transmits a Pdelay_Resp message from the MD entity, containing the parameters in the structure pointed to by txPdelayRespPtr (see 11.2.20.2.3).

11.2.20.3.3 setPdelayRespFollowUp(): Creates a structure containing the parameters (see 11.4) of a Pdelay_Resp_Follow_Up message to be transmitted, and returns a pointer, txPdelayRespFollowUpPtr (see 11.2.20.2.4), to this structure. The parameters are set as follows:

- a) sourcePortIdentity is set equal to the port identity of the port corresponding to this MD entity (see 8.5.2).
- b) sequenceId is set equal to the sequenceId field of the corresponding Pdelay_Req message.
- c) responseOriginTimestamp is set equal to the pdelayRespEventEgressTimestamp (see 11.3.2) of the corresponding Pdelay_Resp message, with any fractional nanoseconds truncated.
- d) correctionField is set equal to the following:
 - 1) The fractional nanoseconds portion of the pdelayRespEventEgressTimestamp of the corresponding Pdelay_Resp message if this state machine is invoked by the *transport*-specific peer-to-peer delay mechanism and
 - 2) The sum of the correctionField of the corresponding Pdelay_Req message and the fractional nanoseconds portion of the pdelayRespEventEgressTimestamp of the corresponding Pdelay_Resp message if this state machine is invoked by CMLDS,
- e) requestingPortIdentity is set equal to the sourcePortIdentity field of the corresponding Pdelay_Req message.
- f) The remaining parameters are set as specified in 11.4.2 and 11.4.6.

11.2.20.3.4 txPdelayRespFollowUp(txPdelayRespFollowUpPtr): Transmits a Pdelay_Resp_Follow_Up message from the P2PPort entity containing the parameters in the structure pointed to by txPdelayRespFollowUpPtr (see 11.2.20.2.4).

11.2.20.4 State diagram

The MDPdelayResp state machine shall implement the function specified by the state diagram in Figure 11-10, the local variables specified in 11.2.20.2, the functions specified in 11.2.20.3, the messages specified in 11.4, and the relevant global variables and functions specified in 10.2.4 through 10.2.6, 11.2.13, and 11.2.18. This state machine is responsible for responding to Pdelay_Req messages, received from the MD entity at the other end of the attached PTP Link, with Pdelay_Resp and Pdelay_Resp_Follow_Up messages.

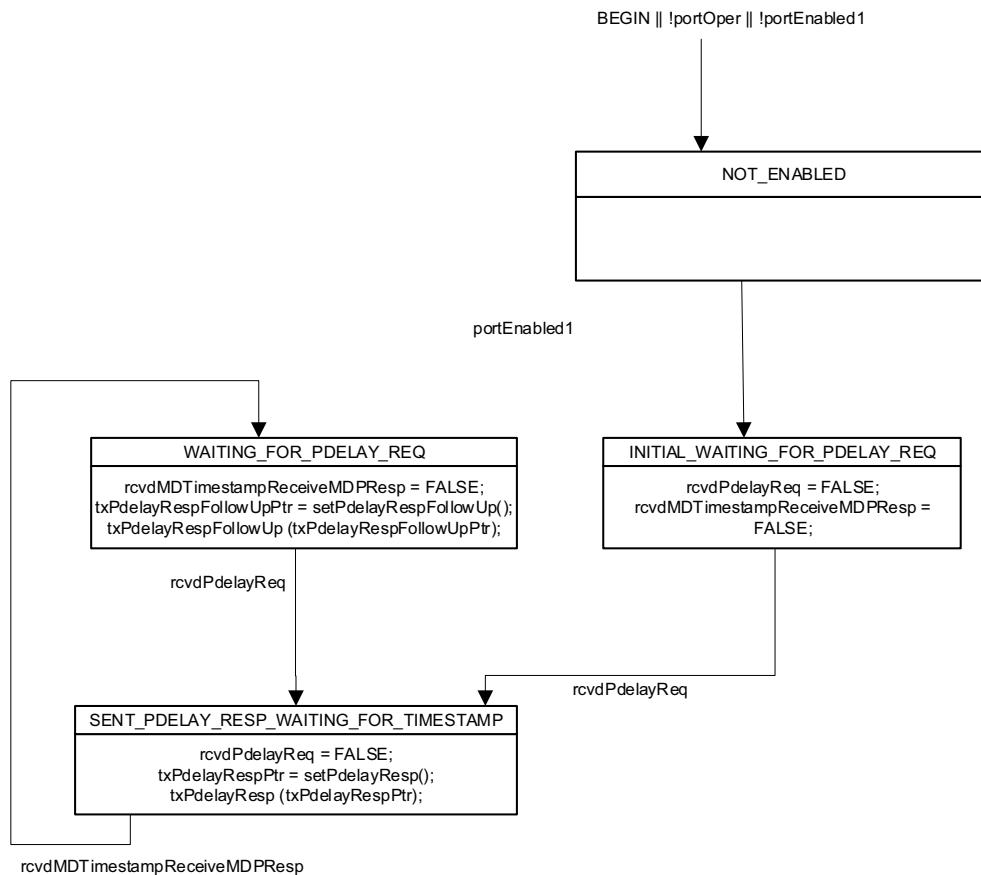


Figure 11-10—MDPdelayResp state machine

11.2.21 LinkDelayIntervalSetting state machine

11.2.21.1 General

This state machine is either invoked as part of the Common Mean Link Delay Service (CMLDS) or by the transport-specific peer-to-peer delay mechanism. There is one instance of this state machine per port, for the Common Service of the time-aware system.

11.2.21.2 State machine variables

The following variables are used in the state diagram in Figure 11-11 (in 11.2.21.4):

11.2.21.2.1 rcvdSignalingMsg1: A Boolean variable that notifies the current state machine when a Signaling message that contains a Message Interval Request TLV (see 10.6.4.3) is received. This variable is reset by the current state machine.

11.2.21.2.2 rcvdSignalingPtrLDIS: A pointer to a structure whose members contain the values of the fields of the received Signaling message that contains a Message Interval Request TLV (see 10.6.4.3).

11.2.21.2.3 portEnabled3: A Boolean variable whose value is equal to ptPortEnabled (see 10.2.5.13) if this state machine is invoked by the transport-specific peer-to-peer delay mechanism and is equal to cmldsLinkPortEnabled (see 11.2.18.1) if this state machine is invoked by the CMLDS.

11.2.21.2.4 logSupportedPdelayReqIntervalMax: The maximum supported logarithm to base 2 of the Pdelay_Req interval. The data type for logSupportedPdelayReqIntervalMax is Integer8.

11.2.21.2.5 logSupportedClosestLongerPdelayReqInterval: The logarithm to base 2 of the Pdelay_Req interval, such that $\text{logSupportedClosestLongerPdelayReqInterval} > \text{logRequestedPdelayReqInterval}$, that is numerically closest to $\text{logRequestedPdelayReqInterval}$, where $\text{logRequestedPdelayReqInterval}$ is the argument of the function computeLogPdelayReqInterval() (see 11.2.21.3.2). The data type for logSupportedClosestLongerPdelayReqInterval is Integer8.

11.2.21.2.6 computedLogPdelayReqInterval: A variable used to hold the result of the function computeLogPdelayReqInterval(). The data type for computedLogPdelayReqInterval is Integer8.

11.2.21.2.7 TEMP: A temporary variable used to reduce clutter in the state diagram (see Figure 11-11). The data type for TEMP is Integer16.

11.2.21.3 State machine functions

11.2.21.3.1 isSupportedLogPdelayReqInterval (logPdelayReqInterval): A Boolean function that returns TRUE if the Pdelay_Req interval given by the argument logPdelayReqInterval is supported by the PTP Port and FALSE if the Pdelay_Req interval is not supported by the PTP Port. The argument logPdelayReqInterval has the same data type and format as the field logLinkDelayInterval of the message interval request TLV (see 10.6.4.3.6).

11.2.21.3.2 computeLogPdelayReqInterval (logRequestedPdelayReqInterval): An Integer8 function that computes and returns the logPdelayReqInterval, based on the logRequestedPdelayReqInterval. This function is defined as indicated below. It is defined here so that the detailed code that it invokes does not need to be placed into the state machine diagram.

```
Integer8 computeLogPdelayReqInterval (logRequestedPdelayReqInterval)
Integer8 logRequestedPdelayReqInterval;
{
```

```

Integer8 logSupportedPdelayReqIntervalMax,
    logSupportedClosestLongerPdelayReqInterval;
if (isSupportedLogPdelayReqInterval (logRequestedPdelayReqInterval))
    // The requested Pdelay_Req Interval is supported and returned
    return (logRequestedPdelayReqInterval)
else
{
    if (logRequestedPdelayReqInterval > logSupportedPdelayReqIntervalMax)
        // Return the largest supported logPdelayReqInterval, even if smaller than
the requested interval
        return (logSupportedPdelayReqIntervalMax);
    else
        // Return the smallest supported logPdelayReqInterval that is still larger
than
        // the requested interval.
        return (logSupportedClosestLongerPdelayReqInterval);
}
}

```

11.2.21.4 State diagram

The LinkDelayIntervalSetting state machine shall implement the function specified by the state diagram in Figure 11-11, the local variables specified in 11.2.21.2, the functions specified in 11.2.21.3, the messages specified in 10.6 and 11.4, the relevant global variables specified in 10.2.5, 11.2.13, and 11.2.18, the relevant managed objects specified in 14.8 and 14.14, and the relevant timing attributes specified in 10.7 and 11.5. This state machine is responsible for setting the global variables that give the duration of the mean interval between successive Pdelay_Req messages and also the global variables that control whether meanLinkDelay and nrrPdelay are computed, both at initialization and in response to the receipt of a Signaling message that contains a Message Interval Request TLV (see 10.6.4.3).

NOTE—A Signaling message received by this state machine that carries the message interval request TLV (see 10.6.4.3) is ignored if multiple profiles are present and the Signaling message is directed to the CMLDS (i.e., if the value of sdoId of the Signaling message is 0x200).

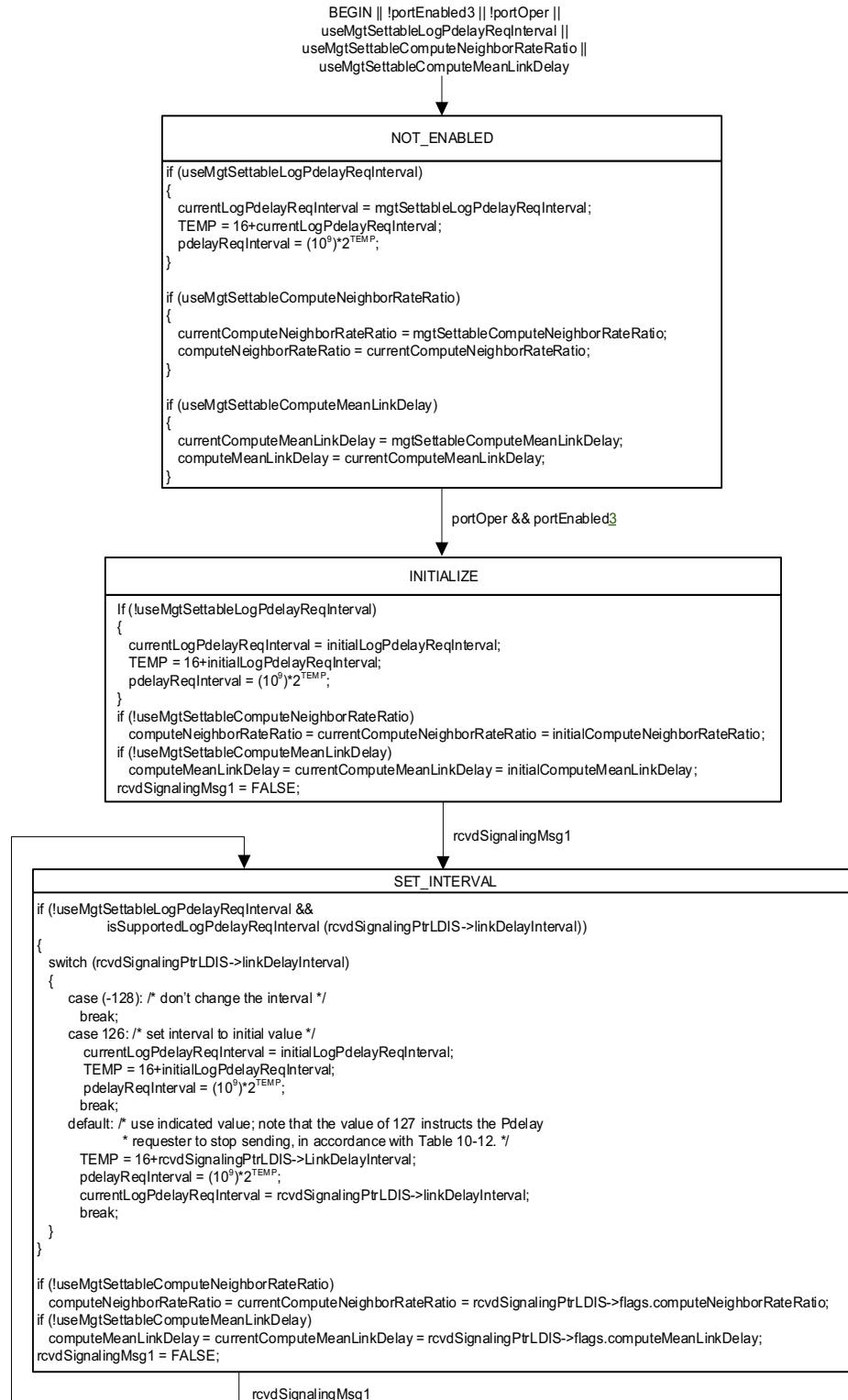


Figure 11-11—LinkDelayIntervalSetting state machine

11.3 Message attributes

11.3.1 General

This subclause describes attributes of the Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages that are not described in 8.4.2.

11.3.2 Message types contained in each message class

11.3.2.1 Event message class

The event message class contains the following message types:

- a) Sync: A Sync message contains time-synchronization information that originates at a ClockTimeTransmitter entity. The appearance of a Sync message at the reference plane of the PTP Port corresponding to an MD entity is an event to which the LocalClock assigns a timestamp, the syncEventIngressTimestamp or syncEventEgressTimestamp, based on the time of the LocalClock. The syncEventIngressTimestamp and syncEventEgressTimestamp are measured relative to the timestamp measurement plane; the MD entity corrects them for ingress and egress latencies, respectively (see 8.4.3). The Sync message is followed by a Follow_Up message containing synchronization information that is based in part on the sum of the syncEventEgressTimestamp and any egressLatency (see 8.4.3).
- b) Pdelay_Req: A Pdelay_Req message is transmitted by an MD entity to another MD entity as part of the peer-to-peer delay mechanism (see 11.2.19 and Figure 11-9) to determine the delay on the PTP Link between them. The appearance of a Pdelay_Req message at the reference plane of the port of an MD entity is an event to which the LocalClock assigns a timestamp, the pdelayReqEventIngressTimestamp or pdelayReqEventEgressTimestamp, based on the time of the LocalClock. The pdelayReqEventIngressTimestamp and pdelayReqEventEgressTimestamp are measured relative to the timestamp measurement plane; the MD entity corrects them for ingress and egress latencies, respectively (see 8.4.3).
- c) Pdelay_Resp: A Pdelay_Resp message is transmitted by an MD entity to another MD entity in response to the receipt of a Pdelay_Req message. The Pdelay_Resp message contains the pdelayReqEventIngressTimestamp of the Pdelay_Req message to which it is transmitted in response. The appearance of a Pdelay_Resp message at the reference plane of the port of an MD entity is an event to which the LocalClock assigns a timestamp, the pdelayRespEventIngressTimestamp or pdelayRespEventEgressTimestamp, based on the time of the LocalClock. The pdelayRespEventIngressTimestamp and pdelayRespEventEgressTimestamp are measured relative to the timestamp measurement plane; the MD entity corrects them for ingress and egress latencies, respectively (see 8.4.3). The Pdelay_Resp message is followed by a Pdelay_Resp_Follow_Up message containing the sum of the pdelayRespEventEgressTimestamp and any egressLatency (see 8.4.3).

Event messages shall be assigned the timestamps previously defined, in accordance with 8.4.3.

11.3.2.2 General message class

The general message class contains the following message types:

- a) Follow_Up: A Follow_Up message communicates the value of the syncEventEgressTimestamp for the associated Sync message.
- b) Pdelay_Resp_Follow_Up: A Pdelay_Resp_Follow_Up message communicates the value of the pdelayRespEventEgressTimestamp for the associated Pdelay_Resp message.

General messages are not required to be timestamped.

11.3.3 VLAN tag

A frame that carries an IEEE 802.1AS message shall not have a VLAN tag nor a priority tag (see 3.184 of IEEE Std 802.1Q-2018).

11.3.4 Addresses

The destination address of Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages shall be the reserved multicast address given in Table 11-3.

Table 11-3—Destination address for Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages

Destination address
01-80-C2-00-00-0E
NOTE—This address is taken from Table 8-1, Table 8-2, and Table 8-3 of IEEE Std 802.1Q-2018.

All Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages shall use the MAC address of the respective egress physical port as the source address.

11.3.5 EtherType

The EtherType of Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages shall be the EtherType given in Table 11-4.

Table 11-4—EtherType for Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages

EtherType
88-F7

11.3.6 Subtype

The subtype for the Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages is indicated by the majorSdId field (see 10.6.2.2.1).

11.3.7 Source port identity

The Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages each contain a sourcePortIdentity field (see 11.4.2.7) that identifies the egress port (see 8.5) on which the respective message is sent.

11.3.8 Sequence number

Each MD entity shall maintain a separate sequenceId pool for each of the message types Sync and Pdelay_Req.

Each Sync and Pdelay_Req message contains a sequenceId field (see 11.4.2.8), which carries the message sequence number. The sequenceId of a Sync message shall be one greater than the sequenceId of the previous Sync message sent by the transmitting port, subject to the constraints of the rollover of the UIInteger16 data type used for the sequenceId field. The sequenceId of a Pdelay_Req message shall be one greater than the sequenceId of the previous Pdelay_Req message sent by the transmitting port, subject to the constraints of the rollover of the UIInteger16 data type used for the sequenceId field.

Separate pools of sequenceId are not maintained for the following message types:

- a) Pdelay_Resp
- b) Follow_Up
- c) Pdelay_Resp_Follow_Up

For these exceptions, the sequenceId value is specified in Table 11-7 (in 11.4.2.8).

11.3.9 Event message timestamp point

The message timestamp point for a PTP event message shall be the beginning of the first symbol following the start of frame delimiter.

11.4 Message formats

11.4.1 General

The PTP messages Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up shall each have a header, body, and if present, a suffix that contains one or more TLVs (see 10.6.2, 11.4.3, 11.4.4, 11.4.5, 11.4.6, and 11.4.7 of this standard and Clause 14 of IEEE Std 1588-2019). Reserved fields shall be transmitted with all bits of the field 0 and ignored by the receiver, unless otherwise specified. The data type of the field shall be the type indicated in brackets in the title of each subclause.

Subclause 11.4.4.3 defines the Follow_Up information TLV, which is carried by the Follow_Up message if the corresponding Sync message is two-step (i.e., twoStepFlag of the Sync message is TRUE; see 10.6.2.2.8) and by the Sync message if the message is one-step (i.e., twoStepFlag is FALSE). The Follow_Up information TLV is the first TLV of the Follow_Up message or Sync message. The requirements for the parsing and forwarding of TLVs are given in 10.6.1.

NOTE—The standard Ethernet header and FCS (18 bytes total) must be added to each message.

11.4.2 Header

11.4.2.1 General

The common header for the PTP messages Sync, Follow_Up, Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up shall be as specified in 10.6.2, except as noted in the following subclauses.

11.4.2.2 messageType (Enumeration4)

The value indicates the type of the message, as defined in Table 11-5.

Table 11-5—Values for messageType field

Message type	Message class	Value
Sync	Event	0x0
Pdelay_Req	Event	0x2
Pdelay_Resp	Event	0x3
Follow_Up	General	0x8
Pdelay_Resp_Follow_Up	General	0xA
NOTE—Other values for the messageType field, except for 0xB that is used for the Announce message and 0xC that is used for the Signaling message (see 10.6.2.2.2), are not used in this standard.		

The most significant bit of the message ID field divides this field in half between event and general messages, i.e., it is 0 for event messages and 1 for general messages.

NOTE—The reserved nibble immediately following messageType is reserved for future expansion of the messageType field.

11.4.2.3 messageLength (UInteger16)

The value is the total number of octets that form the PTP message. The counted octets start with and include the first octet of the header and terminate with and include the last octet of the last TLV or, if there are no TLVs, with the last octet of the message as defined in this subclause.

NOTE—See 10.6.2.5 for an example.

11.4.2.4 domainNumber (UInteger8)

The domainNumber for Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages shall be 0. The domainNumber for all other PTP messages is as specified in 10.6.2.2.6.

11.4.2.5 flags (Octet2)

The value of the bits of the array are defined in Table 10-9.

11.4.2.6 correctionField (Integer64)

The correctionField is the value of the correction as specified in Table 11-6, measured in nanoseconds and multiplied by 2^{16} . For example, 2.5 ns is represented as 0x00000000000028000.

A value of one in all bits, except the most significant, of the field, indicates that the correction is too big to be represented.

Table 11-6—Value of correctionField

Message type	Value
Follow_Up Sync (sent by a one-step PTP Port; see 11.1.3 and 11.2.13.9)	Corrections for fractional nanoseconds (see 10.2.9 and Figure 10-5), difference between preciseOriginTimestamp field (if sent by a two-step PTP Port) or originTimestamp field (if sent by a one-step PTP Port) and current synchronized time (see 11.2.15.2.3 and Figure 11-7), and asymmetry (see 8.3, 11.2.14.2.1, and 11.2.15.2.3. The quantity delayAsymmetry is used in the computation of upstreamTxTime in 11.2.14.2.1, and upstreamTxTime is used in computing an addition to the correctionField in 11.2.15.2.3).
Pdelay_Resp, Pdelay_Resp_Follow_Up	Corrections for fractional nanoseconds (see Figure 11-9 and Figure 11-10) if the message is sent by the <i>transport</i> -specific peer-to-peer delay mechanism; or For Pdelay_Resp, minus the corrections for fractional nanoseconds (see 11.2.20.3.1, Figure 11-9, and Figure 11-10), and for Pdelay_Resp_Follow_Up, the sum of the correctionField of the corresponding Pdelay_Req message and the fractional nanoseconds portion of the pdelayRespEventEgressTimestamp of the corresponding Pdelay_Resp message, if this state machine is invoked by CMLDS.
Sync (sent by a two-step PTP Port), Pdelay_Req, Announce, Signaling	The value is 0 (see 10.6.2.2.9) if the message is sent by the <i>transport</i> -specific peer-to-peer delay mechanism, or The value is 0 for Sync (sent by a two-step PTP Port), Announce, and Signaling, and –delayAsymmetry for Pdelay_Req (see 11.2.19.3.1), if the message is sent by CMLDS.
NOTE—IEEE Std 1588-2019 describes asymmetry corrections for the Pdelay_Req and Pdelay_Resp messages. However, the peer-to-peer delay mechanism computes the mean propagation delay. Here, where the gPTP communication path is a full-duplex point-to-point PTP Link, these corrections cancel in the mean propagation delay computation and therefore are not needed.	

11.4.2.7 sourcePortIdentity (PortIdentity)

The value is the portIdentity of the egress port (see 8.5.2) on which the respective message is sent.

11.4.2.8 sequenceId (UInteger16)

The value is assigned by the originator of the message in conformance with 11.3.8, except for Follow_Up, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages. The sequenceId field values for these exceptions are defined in the state diagrams given in the figures referenced in Table 11-7.

Table 11-7—References for sequenceId value exceptions

Message type	Reference
Follow_Up	See 11.2.15 and Figure 11-7
Pdelay_Resp	See 11.2.20 and Figure 11-10
Pdelay_Resp_Follow_Up	See 11.2.20 and Figure 11-10

11.4.2.9 logMessageInterval (Integer8)

For Sync and Follow_Up messages, the value is the value of currentLogSyncInterval (see 10.2.5.4 and 10.7.2.3). For Pdelay_Req messages, the value is the value of currentLogPdelayReqInterval. For Pdelay_Resp and Pdelay_Resp_Follow_Up messages, the value is transmitted as 0x7F and ignored on reception.

11.4.3 Sync message

11.4.3.1 General Sync message specifications

If the twoStep flag of the PTP common header (see Table 10-9) of the Sync message is TRUE, the fields of the Sync message shall be as specified in Table 11-8. If the twoStep flag of the PTP common header of the Sync message is FALSE, the fields of the Sync message shall be as specified in Table 11-9 and 11.4.3.2. Carrying the Drift_Tracking TLV is optional

Table 11-8—Sync message fields if twoStep flag is TRUE

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
reserved								10	34

Table 11-9—Sync message fields if twoStep flag is FALSE

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
originTimestamp								10	34
Follow_Up information TLV								32	44
Drift_Tracking TLV (optional)								30	76

11.4.3.2 Sync message field specifications if twoStep flag is FALSE

11.4.3.2.1 originTimestamp (Timestamp)

The value of the originTimestamp field is the sourceTime of the ClockTimeTransmitter entity of the Grandmaster PTP Instance, when the Sync message was sent by that Grandmaster PTP Instance, with any fractional nanoseconds truncated (see 10.2.9).

The sum of the correctionField and the originTimestamp field of the Sync message is the value of the synchronized time corresponding to the syncEventEgressTimestamp at the PTP Instance that sent the Sync message, including any fractional nanoseconds.

11.4.3.2.1a Drift_Tracking TLV

The Sync message may carry the Drift_Tracking TLV, defined in 11.4.4.4.

11.4.3.2.2 Follow_Up information TLV

The Sync message carries the Follow_Up information TLV, defined in 11.4.4.3. This TLV shall be the first TLV after the fixed fields.

11.4.4 Follow_Up message

11.4.4.1 General Follow_Up message specifications

The fields of the Follow_Up message shall be as specified in Table 11-10 and 11.4.4.2. Carrying the Drift_Tracking TLV is optional.

Table 11-10—Follow_Up message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
preciseOriginTimestamp								10	34
Follow_Up information TLV								32	44
Drift_Tracking TLV (optional)								30	76

11.4.4.2 Follow_Up message field specifications

11.4.4.2.1 preciseOriginTimestamp (Timestamp)

The value of the preciseOriginTimestamp field is the sourceTime of the ClockTimeTransmitter entity of the Grandmaster PTP Instance, when the associated Sync message was sent by that Grandmaster PTP Instance, with any fractional nanoseconds truncated (see 10.2.9).

The sum of the correctionFields in the Follow_Up and associated Sync messages, added to the preciseOriginTimestamp field of the Follow_Up message, is the value of the synchronized time corresponding to the syncEventEgressTimestamp at the PTP Instance that sent the associated Sync message, including any fractional nanoseconds.

11.4.4.2.2 Follow_Up information TLV

The Follow_Up message carries the Follow_Up information TLV, defined in 11.4.4.3. This TLV shall be the first TLV after the fixed fields.

11.4.4.2.3 Drift_Tracking TLV

The Follow_Up message may carry the Drift_Tracking TLV, defined in 11.4.4.4.

11.4.4.3 Follow_Up information TLV definition

11.4.4.3.1 General

The fields of the Follow_Up information TLV shall be as specified in Table 11-11 and in 11.4.4.3.2 through 11.4.4.3.9. This TLV is a standard organization extension TLV for the Follow_Up message, as specified in 14.3 of IEEE Std 1588-2019.

Table 11-11—Follow_Up information TLV

7	6	5	4	3	2	1	0	Bits	Octets	Offset
				tlvType				2	0	
				lengthField				2	2	
				organizationId				3	4	
				organizationSubType				3	7	
				cumulativeScaledRateOffset				4	10	
				gmTimeBaseIndicator				2	14	
				lastGmPhaseChange				12	16	
				scaledLastGmFreqChange				4	28	

NOTE—The Follow_Up information TLV is different from the CUMULATIVE_RATE_RATIO TLV of IEEE Std 1588-2019 (see 16.10 and Table 52 of IEEE Std 1588-2019). While both TLVs carry cumulative rate offset information, the Follow_Up information TLV also carries information on the Grandmaster Clock time base, most recent phase change, and most recent frequency change. The CUMULATIVE_RATE_RATIO TLV is not used by gPTP.

11.4.4.3.2 tlvType (Enumeration16)

The value of the tlvType field is 0x3.

NOTE—This value indicates the TLV is a vendor and standard organization extension TLV, as specified in 14.3.2.1 and Table 52 of IEEE Std 1588-2019. The tlvType is specified in that standard as ORGANIZATION_EXTENSION with a value of 0x3.

11.4.4.3.3 lengthField (UInteger16)

The value of the lengthField is 28.

11.4.4.3.4 organizationId (Octet3)

The value of organizationId is 00-80-C2.

11.4.4.3.5 organizationSubType (Enumeration24)

The value of organizationSubType is 1.

11.4.4.3.6 cumulativeScaledRateOffset (Integer32)

The value of cumulativeScaledRateOffset is equal to $(\text{rateRatio} - 1.0) \times (2^{41})$, truncated to the next smaller signed integer, where rateRatio is the ratio of the frequency of the Grandmaster Clock to the frequency of the LocalClock entity in the PTP Instance that sends the message.

NOTE—The above scaling allows the representation of fractional frequency offsets in the range $[-(2^{-10} - 2^{-41}), 2^{-10} - 2^{-41}]$, with granularity of 2^{-41} . This range is approximately $[-9.766 \times 10^{-4}, 9.766 \times 10^{-4}]$.

11.4.4.3.7 gmTimeBaseIndicator (UInteger16)

The value of gmTimeBaseIndicator is the timeBaseIndicator of the ClockSource entity for the current Grandmaster PTP Instance (see 9.2.2.3).

NOTE—The timeBaseIndicator is supplied by the ClockSource entity to the ClockTimeTransmitter entity via the ClockSourceTime.invoke function (see 9.2.2.3).

11.4.4.3.8 lastGmPhaseChange (ScaledNs)

The value of lastGmPhaseChange is the time of the current Grandmaster Clock minus the time of the previous Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the Grandmaster PTP Instance. The value is copied from the lastGmPhaseChange member of the MDSyncSend structure whose receipt causes the MD entity to send the Follow_Up message (see 11.2.11).

11.4.4.3.9 scaledLastGmFreqChange (Integer32)

The value of scaledLastGmFreqChange is the fractional frequency offset of the current Grandmaster Clock relative to the previous Grandmaster Clock, at the time that the current Grandmaster PTP Instance became the Grandmaster PTP Instance, or relative to itself prior to the last change in gmTimeBaseIndicator, multiplied by 2^{41} and truncated to the next smaller signed integer. The value is obtained by multiplying the lastGmFreqChange member of MDSyncSend whose receipt causes the MD entity to send the Follow_Up message (see 11.2.11) by 2^{41} , and truncating to the next smaller signed integer.

NOTE—The above scaling allows the representation of fractional frequency offsets in the range $[-(2^{-10} - 2^{-41}), 2^{-10} - 2^{-41}]$, with granularity of 2^{-41} . This range is approximately $[-9.766 \times 10^{-4}, 9.766 \times 10^{-4}]$.

11.4.4.4 Drift_Tracking TLV definition

11.4.4.4.1 General

The fields of the Drift_Tracking TLV shall be as specified in Table 11-11 and in 11.4.4.4.2 through 11.4.4.4.9. This TLV is a standard organization extension TLV for the Sync or Follow_Up message, as specified in 14.3 of IEEE Std 1588-2019.

11.4.4.4.2 tlvType (Enumeration16)

The value of the tlvType is 0x3.

NOTE—This value indicates the TLV is a vendor and standard-organization extension TLV, as specified in 14.3.2.1 and Table 52 of IEEE Std 1588-2019. The tlvType is specified in that standard as ORGANIZATION_EXTENSION with a value of 0x3.

11.4.4.4.3 lengthField (UInteger16)

The value of lengthField is 26.

Table 11-11a—Drift_Tracking TLV

Bits	Octets	Offset
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
organizationId	3	4
organizationSubType	3	7
syncEgressTimestamp	12	10
syncGrandmasterIdentity	8	22
syncStepsRemoved	2	30
rateRatioDrift	4	32

11.4.4.4.4 organizationId (Octet3)

The value of organizationId is 00-80-C2.

11.4.4.4.5 organizationSubType (Enumeration24)

The value of organizationSubType is 6.

11.4.4.4.6 syncEgressTimestamp (ExtendedTimestamp)

The value of the syncEgressTimestamp field is the timestamp, based on the local clock, when the Sync message was sent by that PTP Relay Instance (see 11.4.3.2).

11.4.4.4.7 syncGrandmasterIdentity (ClockIdentity)

The value of the syncGrandmasterIdentity field is the value of the clockIdentity component of the rootSystemIdentity of the gmPriorityVector (see 10.3.5) of the PTP Instance that transmits the Sync message.

11.4.4.4.8 syncStepsRemoved (UInteger16)

The value of the syncStepsRemoved field is the value of syncTimeTransmitterStepsRemoved (see 10.3.9.3) for the PTP Instance that transmits the Sync message.

11.4.4.4.9 rateRatioDrift (Integer32)

The value of rateRatioDrift is equal to $(RRdrift - 1.0) \times (2^{41})$, truncated to the next smaller signed integer, where RRdrift is the measured estimate of the rate of change per second of the ratio of the frequency of the Grandmaster Clock to the frequency of the Local Clock entity in the PTP Instance that sends the message.

NOTE—The above scaling allows the representation of rates of change of fractional frequency offset in the range $[-(2^{-10} - 2^{-41}), 2^{-10} - 2^{-41}] \text{ s}^{-1}$, with granularity of 2^{-41} . This range is approximately $[-9.766 \times 10^{-4}, 9.766 \times 10^{-4}] \text{ s}^{-1}$.

11.4.5 Pdelay_Req message

The fields of the Pdelay_Req message shall be as specified in Table 11-12.

Table 11-12—Pdelay_Req message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
reserved								10	34
reserved								10	44

11.4.6 Pdelay_Resp message

11.4.6.1 General Pdelay_Resp message specifications

The fields of the Pdelay_Resp message shall be as specified in Table 11-13 and 11.4.6.2.

Table 11-13—Pdelay_Resp message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
requestReceiptTimestamp								10	34
requestingPortIdentity								10	44

11.4.6.2 Pdelay_Resp message field specifications

11.4.6.2.1 requestReceiptTimestamp (Timestamp)

The value is the seconds and nanoseconds portion of the pdelayReqEventIngressTimestamp of the associated Pdelay_Req message (see 11.2.19).

11.4.6.2.2 requestingPortIdentity (PortIdentity)

The value is the value of the sourcePortIdentity field of the associated Pdelay_Req message (see 11.4.5).

11.4.7 Pdelay_Resp_Follow_Up message

11.4.7.1 General Pdelay_Resp_Follow_Up message specifications

The fields of the Pdelay_Resp_Follow_Up message shall be as specified in Table 11-14 and 11.4.7.2.

Table 11-14—Pdelay_Resp_Follow_Up message fields

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (see 11.4.2)								34	0
responseOriginTimestamp								10	34
requestingPortIdentity								10	44

11.4.7.2 Pdelay_Resp_Follow_Up message field specifications

11.4.7.2.1 responseOriginTimestamp (Timestamp)

The value is the seconds and nanoseconds portion of the pdelayEventEgressTimestamp of the associated Pdelay_Resp message (see 11.4.6.2).

11.4.7.2.2 requestingPortIdentity (PortIdentity)

The value is the value of the sourcePortIdentity field of the associated Pdelay_Req message (see 11.4.5).

11.5 Protocol timing characterization

11.5.1 General

This subclause specifies timing attributes for the media-dependent sublayer specified in this clause.

11.5.2 Message transmission intervals

11.5.2.1 General interval specification

The mean time interval between successive Pdelay_Req messages is represented as the logarithm to the base 2 of this time interval measured in seconds. The value of this logarithmic attribute shall be as specified in 11.5.2.2.

The mean time interval between successive Sync messages shall be as specified in 10.7.2.1, 10.7.2.3, and 11.5.2.3.

11.5.2.2 Pdelay_Req message transmission interval

When useMgtSettableLogPdelayReqInterval (see 14.16.12) is FALSE, the initialLogPdelayReqInterval specifies the following:

- a) The mean time interval between successive Pdelay_Req messages sent over a PTP Link when the port is initialized, and
- b) The value to which the mean time interval between successive Pdelay_Req messages is set when a message interval request TLV is received with the logLinkDelayIntervalField set to 126 (see 11.2.21).

The currentLogPdelayReqInterval specifies the current value of the mean time interval between successive Pdelay_Req messages. The default value of initialLogPdelayReqInterval is 0. Every port supports the

value 127; the port does not send Pdelay_Req messages when currentLogPdelayReqInterval has this value (see 11.2.21). A port may support other values, except for the reserved values indicated in Table 10-15. A port shall ignore requests for unsupported values (see 11.2.21). The initialLogPdelayReqInterval and currentLogPdelayReqInterval are per-port attributes.

When useMgtSettableLogPdelayReqInterval is TRUE, currentLogPdelayReqInterval is set equal to mgtSettableLogPdelayReqInterval (see 14.16.13), and initialLogPdelayReqInterval is ignored.

Pdelay_Req messages shall be transmitted such that the value of the arithmetic mean of the intervals, in seconds, between Pdelay_Req message transmissions is not less than the value of $0.9 \times 2^{\text{currentLogPdelayReqInterval}}$.

NOTE 1—A minimum number of inter-message intervals is necessary to verify that a PTP Port meets these requirements. The arithmetic mean is the sum of the inter-message interval samples divided by the number of samples. For more detailed discussion of statistical analyses, see Papoulis [B25].

NOTE 2—If useMgtSettableLogPdelayReqInterval is FALSE, the value of initialLogPdelayReqInterval is the value of the mean time interval between successive Pdelay_Req messages when the port is initialized. The value of the mean time interval between successive Pdelay_Req messages can be changed, e.g., if the port receives a Signaling message that carries a message interval request TLV (see 10.6.4.3) and the current value is stored in currentLogPdelayReqInterval. The value of the mean time interval between successive Pdelay_Req messages can be reset to the initial value, e.g., by a message interval request TLV for which the value of the field logLinkDelayInterval is 126 (see 10.6.4.3.6).

NOTE 3—A port that requests (using a Signaling message that contains a message interval request TLV; see 10.6.4 and 11.2.21) that the port at the other end of the attached PTP Link set its currentLogPdelayReqInterval to a specific value can determine if the request was honored by examining the logMessageInterval field of subsequent received Pdelay_Req messages.

NOTE 4—The MDPdelayReq state machine ensures that the times between transmission of successive Pdelay_Req messages, in seconds, are not smaller than $2^{\text{currentLogPdelayReqInterval}}$. This is consistent with the requirements of the current subclause and IEEE Std 1588-2019, which require that the value of the arithmetic mean of the intervals, in seconds, between Pdelay_Req message transmissions is not less than the value of $0.9 \times 2^{\text{currentLogPdelayReqInterval}}$. The sending of Pdelay_Req messages is governed by the LocalClock and not the synchronized time (i.e., the estimate of the Grandmaster Clock time). Since the LocalClock frequency can be slightly larger than the Grandmaster Clock frequency (e.g., by 100 ppm, which is the specified frequency accuracy of the LocalClock; see B.1.1), it is possible for the time intervals between successive Pdelay_Req messages to be slightly less than $2^{\text{currentLogPdelayReqInterval}}$ when measured relative to the synchronized time. The factor 0.9 allow a margin of 10% for the arithmetic mean of the successive intervals between Pdelay_Req messages.

11.5.2.3 Sync message transmission interval default value

The default value of initialLogSyncInterval (see 10.7.2.3) is -3. Every PTP Port supports the value 127; the PTP Port does not send Sync messages when currentLogSyncInterval has this value (see 10.3.18). A PTP Port may support other values, except for the reserved values indicated in Table 10-16. A PTP Port ignores requests for unsupported values (see 10.3.18).

NOTE—A PTP Port that requests (using a Signaling message that contains a message interval request TLV; see 10.6.4 and 10.3.18) that the PTP Port at the other end of the attached PTP Link set its currentLogSyncInterval to a specific value can determine if the request was honored by examining the logMessageInterval field of subsequent received Sync messages.

11.5.3 allowedLostResponses

The variable allowedLostResponses (see 11.2.13.4) is the number of Pdelay_Req messages without valid responses above which a port is considered to be not exchanging peer delay messages with its neighbor. The default value of allowedLostResponses shall be 9. The range shall be 1 through 255.

11.5.4 allowedFaults

The variable allowedFaults (see 11.2.13.5) is the number of faults above which asCapableAcrossDomains is set to FALSE, i.e., the port is considered not capable of interoperating with its neighbor via the IEEE 802.1AS protocol (see 10.2.5.1). In this context, the term *faults* refers to instances where

- a) The computed mean propagation delay, i.e., meanLinkDelay (see 10.2.5.8), exceeds the threshold, meanLinkDelayThresh (see 11.2.13.7) and/or
- b) The computation of **nrrPdelay** is invalid (see 11.2.19.2.10).

The default value of allowedFaults shall be 9. The range shall be 1 through 255.

NOTE—The above description of allowedFaults uses the variable asCapableAcrossDomains (see 11.2.13.12). When only one domain is active, asCapableAcrossDomains is equivalent to the variable asCapable.

11.6 Control of computation of neighborRateRatio

If the variable driftTrackingTlvSupport (see 10.2.4.28) is TRUE and nrrCompMethod (see 11.2.13.15) is equal to Sync, the value of neighborRateRatio (see 10.2.5.7) is set equal to nrrSync (see 11.2.13.14).

If the variable driftTrackingTlvSupport (see 10.2.4.28) is FALSE or nrrCompMethod (see 11.2.13.15) is equal to Pdelay, the value of neighborRateRatio (see 10.2.5.7) is set equal to nrrPdelay (see 11.2.13.13). The computation of nrrPdelay is controlled as described below. The description below applies only to the computation of nrrPdelay, and not to the computation of nrrSync.

The variable computeNeighborRateRatio (see 10.2.5.10) indicates whether **nrrPdelay** is to be computed by this port when the peer-to-peer delay mechanism is invoked.

When useMgtSettableComputeNeighborRateRatio (see 14.16.16) is FALSE, computeNeighborRateRatio is initialized to the value of initialComputeNeighborRateRatio.

The currentComputeNeighborRateRatio specifies the current value of computeNeighborRateRatio. The default value of initialComputeNeighborRateRatio is TRUE. The initialComputeNeighborRateRatio and currentComputeNeighborRateRatio are per-port attributes.

When useMgtSettableComputeNeighborRateRatio is TRUE, currentComputeNeighborRateRatio is set equal to mgtSettableComputeNeighborRateRatio (see 14.16.17), and initialComputeNeighborRateRatio is ignored.

NOTE—If useMgtSettableComputeNeighborRateRatio is FALSE, the value of initialComputeNeighborRateRatio determines whether **nrrPdelay** is computed by the peer delay mechanism when the port is initialized. The value of computeNeighborRateRatio can be changed, e.g., if the port receives a Signaling message that carries a message interval request TLV (see 10.6.4.3) and the current value is stored in currentComputeNeighborRateRatio.

11.7 Control of computation of meanLinkDelay

The variable computeMeanLinkDelay (see 10.2.5.10) indicates whether meanLinkDelay is to be computed by this port when the peer-to-peer delay mechanism is invoked.

When useMgtSettableComputeMeanLinkDelay (see 14.16.20) is FALSE, computeMeanLinkDelay is initialized to the value of initialComputeMeanLinkDelay.

The currentComputeMeanLinkDelay specifies the current value of computeMeanLinkDelay. The default value of initialComputeMeanLinkDelay is TRUE. The initialComputeMeanLinkDelay and currentComputeMeanLinkDelay are per-port attributes.

When useMgtSettableComputeMeanLinkDelay is TRUE, currentComputeMeanLinkDelay is set equal to mgtSettableComputeMeanLinkDelay (see 14.16.21), and initialComputeMeanLinkDelay is ignored.

NOTE—If useMgtSettableComputeMeanLinkDelay is FALSE, the value of initialComputeMeanLinkDelay determines whether meanLinkDelay is computed by the peer delay mechanism when the port is initialized. The value of computeMeanLinkDelay can be changed, e.g., if the port receives a Signaling message that carries a message interval request TLV (see 10.6.4.3) and the current value is stored in currentComputeMeanLinkDelay.

1 12. Media-dependent layer specification for IEEE 802.11 links

2 12.1 Overview

3 12.1.1 General

4 Accurate synchronized time is distributed across a domain through time measurements between adjacent
5 PTP Instances in a packet network. Time is communicated from the root of the clock spanning tree (i.e., the
6 Grandmaster PTP Instance) to the leaves of the tree, by recursively propagating time from a leaf-facing
7 “timeTransmitter” port to some number of root-facing “timeReceiver” ports in PTP Instances at the next
8 level of the tree through measurements made across the links connecting the PTP Instances. While the time
9 semantics are consistent across the time-aware packet network, the method for communicating synchronized
10 time from a timeTransmitter port to the immediate downstream link partner varies depending on the type of
11 link interconnecting the two systems.

12 This clause specifies the interface primitives and state machines that provide accurate synchronized time
13 across wireless IEEE 802.11 links as part of a packet network. This clause builds upon time measurement
14 features defined in IEEE Std 802.11-2016 and makes no distinction between stations with an access point
15 function and stations without an access point function.

16 12.1.2 IEEE 802.11 Timing Measurement and Fine Timing Measurement procedures

17 12.1.2.1 General

18 IEEE Std 802.11-2016 defines a family of wireless measurements, including both “Timing Measurement”
19 (TM) and “Fine Timing Measurement” (FTM), which captures timestamps of the transmit time and receive
20 time of a round-trip message exchange between associated wireless local area network (WLAN) stations.

21 In contrast to the protocol defined for full-duplex point-to-point links, this clause does not define any new
22 frames nor the transmission of any frames. Rather, it makes use of a MAC layer management entity
23 (MLME) interface, which causes the IEEE 802.11 layer to not only take timestamps of measurement frames
24 as they are transmitted and received, but to also *generate* and *consume* the measurement frames, all within
25 the IEEE 802.11 MLME layer, and then to provide timestamp information from the MLME to this media-
26 dependent layer through a set of well-defined service primitives. However, as an aid to the reader, the
27 protocol and frames used by the IEEE 802.11 MLME for both Timing Measurement and Fine Timing
28 Measurement are described briefly as follows and illustrated in Figure 12-1 and Figure 12-2, respectively.

29 Both Timing Measurement and Fine Timing Measurement are accomplished through a round-trip frame
30 exchange. For Timing Measurement, the first frame of the round-trip measurement is generated by the
31 timeTransmitter within the IEEE 802.11 MLME when the MLME-TIMINGMSMT.request primitive is
32 invoked. For Fine Timing Measurement, an initial Fine Timing Measurement request frame is generated by
33 the timeReceiver within the IEEE 802.11 MLME when the MLME-FINETIMINGMSMTRQ.request
34 primitive is invoked. After this frame is successfully received by the timeTransmitter, the first frame of the
35 round-trip measurement is generated by the TimeTransmitter within the IEEE 802.11 MLME when the
36 MLME-FINETIMINGMSMT.request primitive is invoked. As defined by IEEE Std 802.11-2016, upon
37 receipt of the resulting Timing Measurement or Fine Timing Measurement frame, the timeReceiver station
38 transmits an IEEE 802.11 Ack control frame to the timeTransmitter station. Four timestamps are captured
39 during this two-frame exchange, as follows:

- 40 a) t1 is when (in the timeTransmitter station’s time base) the request frame is transmitted
41 b) t2 is when (in the timeReceiver station’s time base) the request frame is received
42 c) t3 is when (in the timeReceiver station’s time base) the Ack control frame is transmitted
43 d) t4 is when (in the timeTransmitter station’s time base) the Ack control frame is received

1 When the timeTransmitter sends either a Fine Timing Measurement or a Timing Measurement frame, it
 2 passes the t1 and t4 timestamps (and other end-to-end synchronization information) and
 3 FollowUpInformation, from the previous measurement to the timeReceiver. A pair of tokens is passed in
 4 each timing or Fine Timing Measurement frame, one to identify the current measurement and the other to
 5 allow the timeReceiver to associate the timestamp information with the previous measurement.

6 NOTE 1—TM also can include a Timing Measurement Request Frame; however, this frame type is not used by this
 7 standard.

8

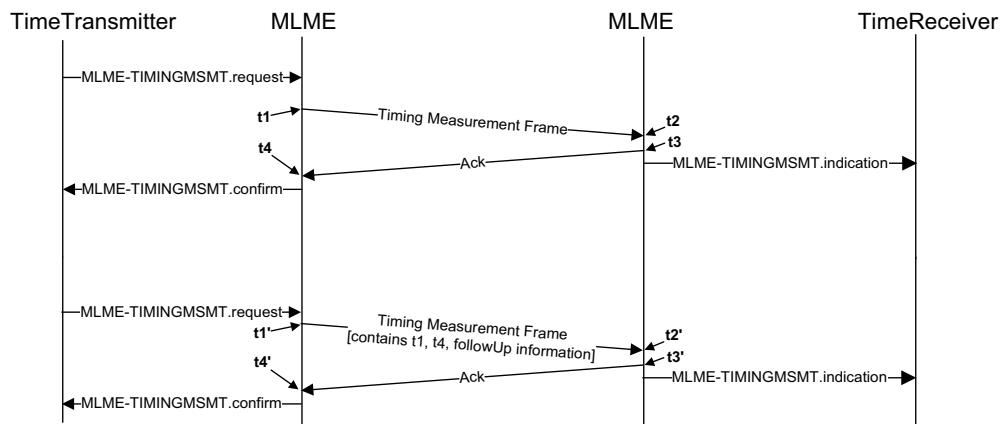


Figure 12-1—Timing measurement procedure for IEEE 802.11 links

1

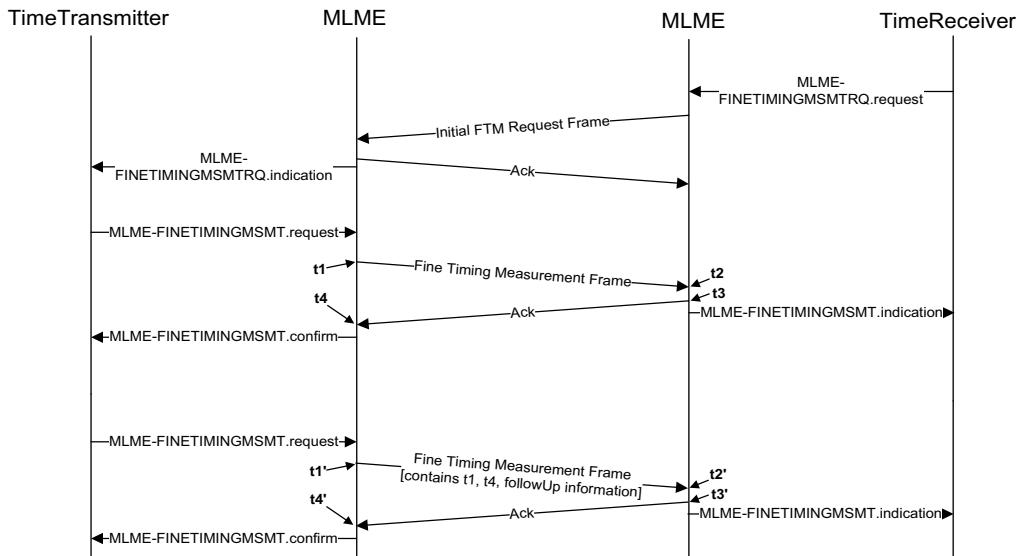


Figure 12-2—Fine Timing Measurement procedure for IEEE 802.11 links

2 Note that, unlike full-duplex point-to-point ports, IEEE 802.11 ports do not compute the link delay
3 measurements in both directions since only a PTP Port in the TimeReceiver state makes use of that
4 information. A PTP Port that transitions from the TimeTransmitter state to the TimeReceiver state (e.g., due
5 to selection of a new Grandmaster PTP Instance) can collect a number of link delay measurements and
6 perform averaging or other filtering to achieve the desired accuracy.

7 NOTE 2—Fine Timing Measurement can be used for time synchronization as described in this standard; however, it can
8 also be used for location services as defined in IEEE Std 802.11-2016. Since FTM supports only one configuration at a
9 time, it is possible that setting that single configuration for time synchronization might disable its previous configuration
10 for use by another application for location services or, conversely, an application that configures FTM for location
11 services could disable this standard’s use for time synchronization. Implementations that use FTM for time
12 synchronization and other applications need to coordinate usage of the FTM protocol.

13 The timeTransmitter generates MLME-TIMINGMSMT.request primitives for Timing Measurement, as
14 described in this standard, in a manner such that the requirements of 10.7.2.3 and 12.8 for the time
15 synchronization message interval are satisfied. Timing measurement frames are then sent from the
16 timeTransmitter to the timeReceiver continuously and at a rate that satisfies the requirements of those two
17 subclauses. It is not necessary for the timeReceiver to continually request timing information from the
18 timeTransmitter. In contrast, the timeReceiver must request timing information from the timeTransmitter for
19 Fine Timing Measurement. A Fine Timing Measurement frame carries timestamp information for a previous
20 measurement. The timeReceiver requests a burst of Fine Timing Measurement frames from the
21 timeTransmitter. Figure 12-2 shows an example of a burst of Fine Timing Measurement frames (there are
22 two frames in that example). The Fine Timing Measurement process is described in more detail in 12.1.2.2
23 and is illustrated in Figure 12-3. In that discussion, the focus is on the transmission of the frames. In the
24 simplified example, service primitives are omitted from Figure 12-3.

25 12.1.2.2 Detailed description of Fine Timing Measurement (FTM)

26 Figure 12-3 is adapted from Figure 11-37 of IEEE Std 802.11-2016. Additional details on the FTM
27 procedure are given in 11.24.6 of IEEE Std 802.11-2016. The example of this figure is for when the

1 initiating station (STA), i.e., the timeReceiver, requests a single burst of three FTM frames from the
 2 responding STA, i.e., the timeTransmitter, as soon as possible. The timeReceiver makes this request by
 3 sending an initial FTM Request to the timeTransmitter with respective parameters set to appropriate values.
 4 The FTM parameters that are relevant to time synchronization in this standard are described in 12.6, and all
 5 the FTM parameters are described in more detail in 9.4.2.168 of IEEE Std 802.11-2016. However, in the
 6 example here, the parameter ASAP is set to 1 to indicate to the timeTransmitter that the FTM frames are
 7 desired as soon as possible, and the Number of Bursts Exponent parameter is set to 0 to indicate a single
 8 burst. Figure 12-3 is a simplified view; the timeReceiver causes the frame to be sent by invoking the
 9 MLME-FINETIMINGMSMTRQ.request primitive, which includes the FTM parameter values. The
 10 timeTransmitter sends an acknowledgment (Ack) frame to the timeReceiver to indicate it received the initial
 11 FTM request. The timeTransmitter then sends an initial FTM frame at a time that is recommended to be no
 12 more than 10 ms later than the receipt of the initial FTM request. The initial FTM frame indicates to the
 13 timeReceiver whether the timeTransmitter was able to grant the values of the FTM parameters that the
 14 timeReceiver requested. If the requested parameters are granted, the procedure continues (Figure 12-3
 15 illustrates this case). If the requested parameters are not granted, the timeReceiver sends a new initial FTM
 16 request for a burst of two FTM frames. If the new request is granted, the procedure continues. If the new
 17 request is not granted, the timeReceiver and timeTransmitter use TM if they both support TM. If, at this
 18 point, the timeReceiver or the timeTransmitter, or both, do not support TM, the procedure terminates and
 19 asCapable is set to FALSE (see 12.4).

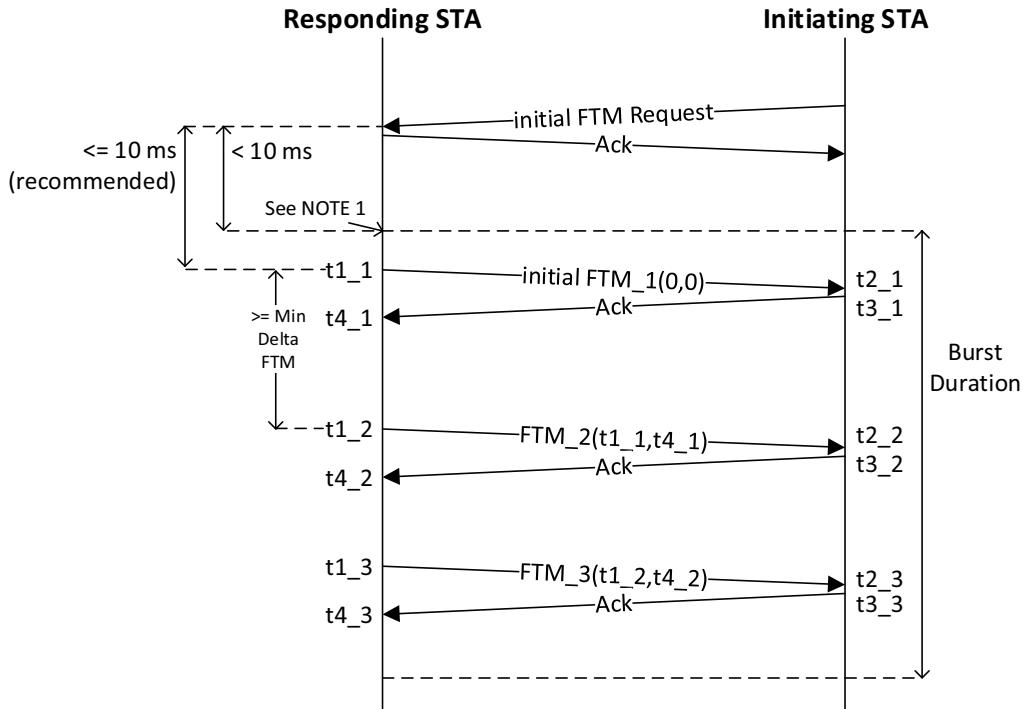


Figure 12-3—Illustration of Fine Timing Measurement burst

20 NOTE—IEEE Std 802.11-2016 allows various options in case the timeTransmitter does not grant the request. The above
 21 procedure is used in this subclause.

22 The initial FTM frame (initial FTM_1 in Figure 12-3) sent by the timeTransmitter is timestamped with the
 23 value $t_1_{_1}$ on transmission from the timeTransmitter and timestamped with the value $t_2_{_1}$ on receipt by the
 24 timeReceiver. The initial FTM frame has fields that carry the t_1 and t_4 timestamps of the previous FTM
 25 frame and corresponding Ack; however, since this is the first FTM frame of the burst, these fields are set to

1 zero. The timeReceiver responds to the timeTransmitter with an Ack, which is timestamped with the value
2 t₃_1 on transmission from the timeReceiver and with the value t₄_1 on receipt by the timeTransmitter.

3 The timeTransmitter sends the second FTM frame (FTM_2 in Figure 12-3) after a time interval since
4 sending the initial FTM frame that is greater than or equal to the Min Delta FTM parameter that was
5 requested by the timeReceiver. This FTM frame is timestamped with t₁_2 on transmission and t₂_2 on
6 reception. This FTM frame also carries the values of the timestamps t₁_1 and t₄_1 of the initial FTM frame
7 and corresponding Ack. On receipt of the second FTM frame, the timeReceiver sends an Ack frame to the
8 timeTransmitter; this frame is timestamped with t₃_2 on transmission and t₄_2 on reception. Finally, the
9 timeTransmitter sends the third FTM frame (FTM_3 in Figure 12-3) to the timeReceiver, also at a time
10 interval since sending FTM_2 that is greater than or equal to the Min Delta FTM parameter that was
11 requested by the timeReceiver. As with FTM_2, this frame is timestamped with t₁_3 on transmission and
12 t₂_3 on reception. This FTM frame also carries the values of the timestamps t₁_2 and t₄_2 of the second
13 FTM frame and corresponding Ack. On receipt of the third FTM frame, the timeReceiver sends an Ack
14 frame to the timeTransmitter; this frame is timestamped with t₃_3 on transmission and t₄_3 on reception.

15 On completion of the above exchanges of FTM frames and corresponding acknowledgments, the
16 timeReceiver knows the transmission and reception times for the initial FTM frame (t₁_1, t₂_1, t₃_1, and
17 t₄_1) and second FTM frame (t₁_2, t₂_2, t₃_2, and t₄_2) and the reception time for the third FTM frame
18 (t₂_3) and transmission time for the corresponding Ack (t₃_3). The timeReceiver can use this information
19 (along with FollowUpInformation contained in the VendorSpecific information element; see 12.7) to
20 synchronize to the timeTransmitter. In this standard, timestamps for the minimum delay FTM frames are
21 used. Specifically, the timeReceiver computes the quantities D₁ = t₂_1 - t₁_1, and D₂ = t₂_2 - t₁_2, and
22 uses the timestamps t₁_i and t₂_i, where i = 1 if D₁ < D₂ and i = 2 if D₁ ≥ D₂, to compute the respective
23 members of the MDSyncReceive structure. The timestamps of FTM_3 and its corresponding Ack are not
24 used; FTM_3 is used only to convey the timestamps of FTM_2 and its corresponding Ack.

25 If the timeTransmitter does not grant the parameters requested initially by the timeReceiver, i.e., for a burst
26 of three FTM frames, but it does grant the subsequent request for a burst of two FTM frames, the
27 timeReceiver has a full set of timestamps for only the initial FTM_1 frame. In this case, the timeReceiver
28 uses the timestamps t₁_1, t₂_1, t₃_1, and t₄_1 to compute the respective members of the MDSyncReceive
29 structure.

30 With the above procedure for FTM, the timeReceiver controls the rate at which time synchronization
31 information is sent from the timeTransmitter. This is different from TM, full-duplex IEEE 802.3, IEEE
32 802.3 EPON, and CSN transports. In those cases, the sending of time synchronization information from the
33 timeTransmitter to the timeReceiver is controlled by the timeTransmitter; this is true for syncLocked (see
34 10.2.5.15) TRUE, in which case the information is sent as soon as it is received from further upstream, and
35 syncLocked FALSE, in which case it is sent independently of information received from further upstream.
36 For FTM, the timeReceiver requests time synchronization information from the timeTransmitter at an
37 average rate equal to the inverse of the current synchronization message interval currentLogSyncInterval
38 (see 12.8 and 14.8.18). In addition, the actual intervals between successive requests by the timeReceiver for
39 time synchronization information meet the requirements of 10.7.2.3. Also, the value of syncLocked at the
40 timeTransmitter port will not affect the sending of time synchronization information from the
41 timeTransmitter to the timeReceiver; the requests for time synchronization information from the
42 timeReceiver are asynchronous to the receipt of time synchronization information from upstream at the node
43 that contains the timeTransmitter port.

44 12.1.3 Layering for IEEE 802.11 links

45 The *media-dependent* (MD) entity is tailored to the link technology and is responsible for translating the
46 PortSync entity's media-independent actions to media-dependent PDUs or primitives as necessary for
47 communicating synchronized time from the timeTransmitter port over the link to a single timeReceiver port.
48 For an IEEE 802.11 link, this one-to-one relationship between the MD entities of the timeTransmitter and
49 timeReceiver implies that if the one physical IEEE 802.11 port is associated with multiple stations, each
50 association requires its own instantiation of the IEEE 802.1AS PortSync entity and MD entity. The MLME-

1 TIMINGMSMT and MLME-FINETIMINGMSMT service primitives defined in IEEE Std 802.11-2016 are
 2 used to perform Timing Measurements and Fine Timing Measurements, respectively, between a
 3 timeTransmitter IEEE 802.11 station and associated IEEE 802.11 timeReceiver stations. Figure 12-4
 4 illustrates how the MD entity interacts with the higher and lower layers.

5

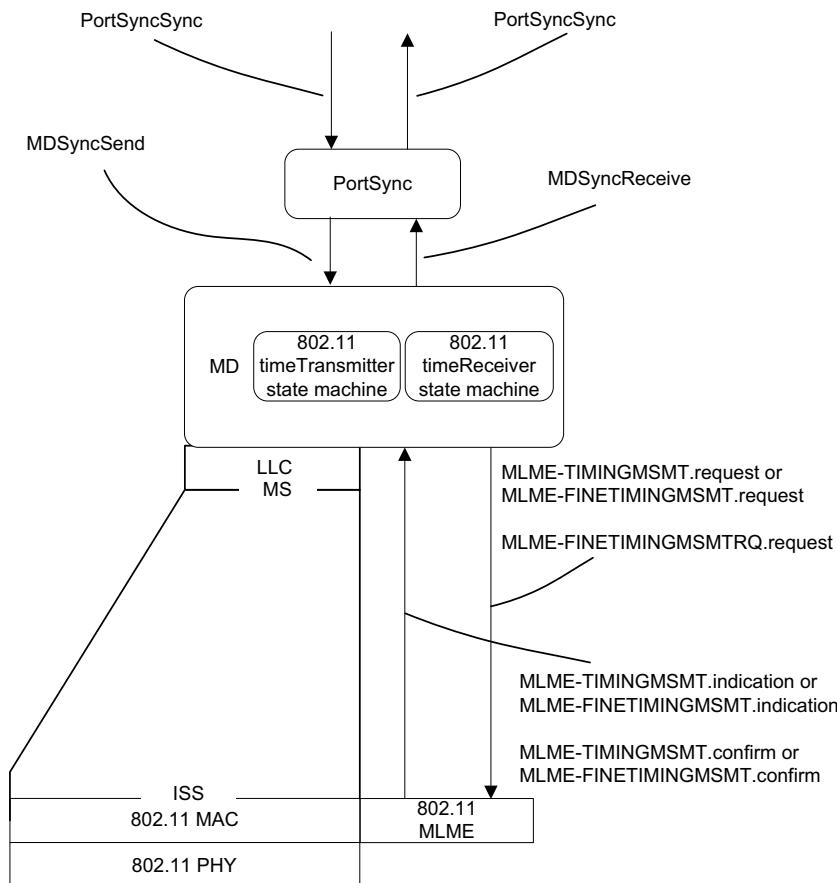


Figure 12-4—Media-dependent and lower entities in stations with IEEE 802.11 links

6

7 12.2 Messages

8 All media-dependent frames are generated and consumed by the lower-layer IEEE 802.11 MLME and thus
 9 none are defined here. Also, since the IEEE 802.11 event messages are timestamped by the MAC/PHY, the
 10 timestamp point is defined in IEEE Std 802.11-2016 as well. Media-independent messages, i.e., Announce
 11 and Signaling messages, are transmitted using the unicast address of the WLAN station instead of the group
 12 address defined in 10.5.3.

1 12.3 Determination of Timing Measurement and Fine Timing Measurement capability

3 The bits of the per-PTP Port global variable tmFtmSupport (see 12.5.1.5) shall be set as indicated in
 4 Table 12-1.

Table 12-1—Values of bits of tmFtmSupport

Bit	Value
0	TRUE if: a) The port supports Timing Measurement and b) The Timing Measurement bit in the Extended Capabilities information element defined in Table 9-135 of IEEE Std 802.11-2016 indicates that the peer IEEE 802.11 station is capable of participating in the Timing Measurement protocol. FALSE otherwise.
1	TRUE if: a) The port supports Fine Timing Measurement and b) The Fine Timing Measurement responder and initiator bits in the Extended Capabilities information element defined in Table 9-135 of IEEE Std 802.11-2016 indicate that the peer IEEE 802.11 station is capable of participating in the Fine Timing Measurement protocol. FALSE otherwise.
2–7	Reserved as FALSE.

5 12.4 Determination of asCapable

6 The per-PTP Port, per-domain instance of the global variable asCapable (see 10.2.5.1) is set to TRUE if the
 7 following conditions hold (see 12.5.1 and 12.5.2):

- 8 a) The value of tmFtmSupport is not zero.
- 9 b) neighborGptpCapable is TRUE.
- 10 c) At least one of the following conditions hold:
 - 11 1) Bit 0 of tmFtmSupport is TRUE.
 - 12 2) Bit 1 of tmFtmSupport is TRUE and, if the PTP Port is a timeTransmitter port, it can support (i.e., grant) the parameters requested by the timeReceiver with either FTMs per burst equal to 3 or FTMs per burst equal to 2.
 - 13 3) Bit 1 of tmFtmSupport is TRUE and, if the PTP Port is a timeReceiver port, the timeTransmitter port at the other end of the link can support (i.e., grant) the parameters requested by the timeReceiver with either FTMs per burst equal to 3 or FTMs per burst equal to 2.

19 If the value of domainNumber is zero (which is required for support of the 2011 edition of this standard) and
 20 bit 0 of tmFtmSupport is TRUE, asCapable can be set to TRUE. In all other instances, asCapable shall be set
 21 to FALSE.

1 NOTE—The above conditions ensure backward compatibility with the 2011 edition of this standard. A time-aware
 2 system that is compliant with the 2011 edition of this standard will not process the gPTP capable TLV, and asCapable
 3 will be determined as specified in the 2011 edition. A PTP Instance of a time-aware system compliant with the current
 4 edition of this standard that is attached, via an IEEE 802.11 link, to a node compliant with the 2011 edition of this
 5 standard will not receive Signaling messages that contain the gPTP capable TLV and will not set neighborGptpCapable
 6 to TRUE. However, item c) 1) in this subclause ensures that asCapable for this PTP Port and domain (i.e., domain 0) will
 7 still be set in a manner consistent with the 2011 edition of this standard.

8 12.5 State machines

9 12.5.1 Media-dependent timeTransmitter state machines

10 12.5.1.1 Overview

11 The MD entity of an IEEE 802.11 port whose port state is TimeTransmitterPort (see Table 10-2) shall
 12 behave in a manner that is indistinguishable, relative to an observer external to a system, from a strict
 13 implementation of the timeTransmitter state machines in Figure 12-5 and Figure 12-6 (denoted as
 14 timeTransmitter state machine A and timeTransmitter state machine B, respectively, in 12.5.1.2), the local
 15 variables specified in 12.5.1.3, the functions specified in 12.5.1.4, the shared variables specified in 12.5.1.5,
 16 and the primitives defined in 12.5.1.6.

17 For Timing Measurement, timeTransmitter state machine A is responsible for initiating a time measurement
 18 whenever the PortSync entity requests it do so, as indicated by the revdMDSyncDot11TimeTransmitterA
 19 Boolean (see 12.5.1.3.8). TimeTransmitter state machine A invokes the IEEE 802.11 MLME-
 20 TIMINGMSMT.request primitive and waits for the subsequent MLME-TIMINGMSMT.confirm primitive.
 21 It collects local timestamp information from the measurement (t1 and t4, provided by the confirm primitive)
 22 and includes the information in the subsequent request. See 8.4.3 for more information on timestamps.
 23 TimeTransmitter state machine B is not used for Timing Measurement.

24 For Fine Timing Measurement, timeTransmitter state machine A receives and stores information from the
 25 PortSync entity. TimeTransmitter state machine B receives the MLME-FINETIMINGMSMTRQ.indication
 26 caused by the initial FTM request from the timeReceiver. It sets asCapable as specified in 12.4. It then
 27 generates successive MLME-FINETIMINGMSMT.request primitives to indicate to the timeReceiver
 28 whether it can grant the requested parameters and also to cause information saved by timeTransmitter state
 29 machine A to be sent to the timeReceiver. It receives MLME-FINETIMINGMSMT.confirm primitives
 30 caused by Acknowledgments received from the timeReceiver. It collects local timestamp information from the current
 31 measurement (t1 and t4, provided by the confirm primitive) and includes the information in the subsequent
 32 MLME-FINETIMINGMSMT.request.

33 12.5.1.2 State diagrams

34 NOTE—In the computation of the burstDuration in timeTransmitter state machine B, the burst duration parameter from
 35 IEEE Std 802.11-2016 is converted to UScaledNs (i.e., units of 2^{-16} ns; see 6.4.3.2). The burst duration in UScaledNs
 36 (see 6.4.3.2) is related to the quantity $A = \text{initReqParamsDot11TimeTransmitterB.burstDuration} - 2$ by:

$$37 \quad \text{burst duration in UScaledNs} = 1000 \times 2^{16} \times 250 \times 2^A$$

38 i.e., A is the logarithm to base 2 of the burst duration, in microseconds, divided by 250. Also, it is assumed that the burst
 39 duration starts when the initial FTM request is received. In actuality, the timer begins by the partial TSF timer value
 40 indicated in the initial FTM frame, which is slightly after the initial FTM request is received.

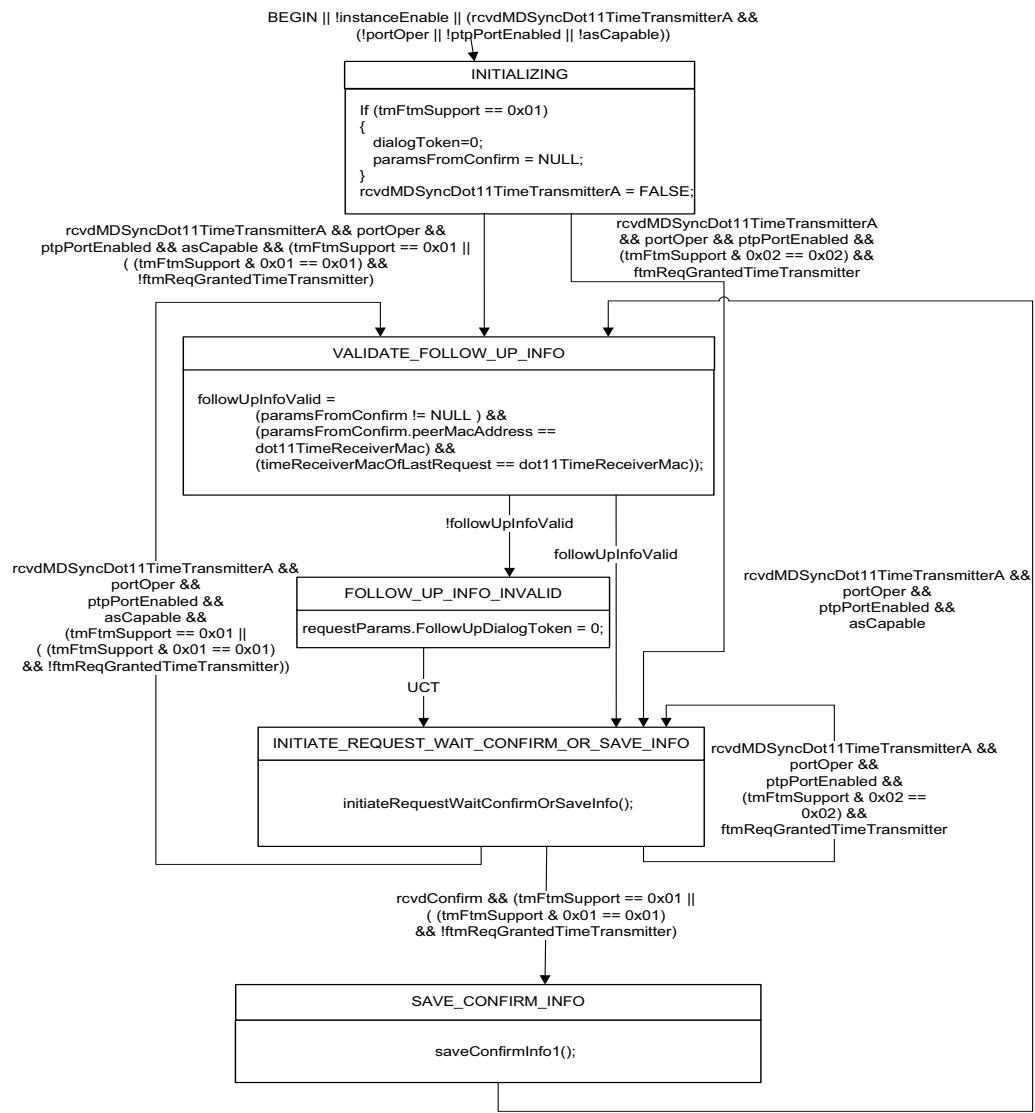


Figure 12-5—TimeTransmitter state machine A
(a) For TM, receives information from the PortSync entity and sends to timeReceiver, and
(b) for FTM, receives and stores information from the PortSync entity

1

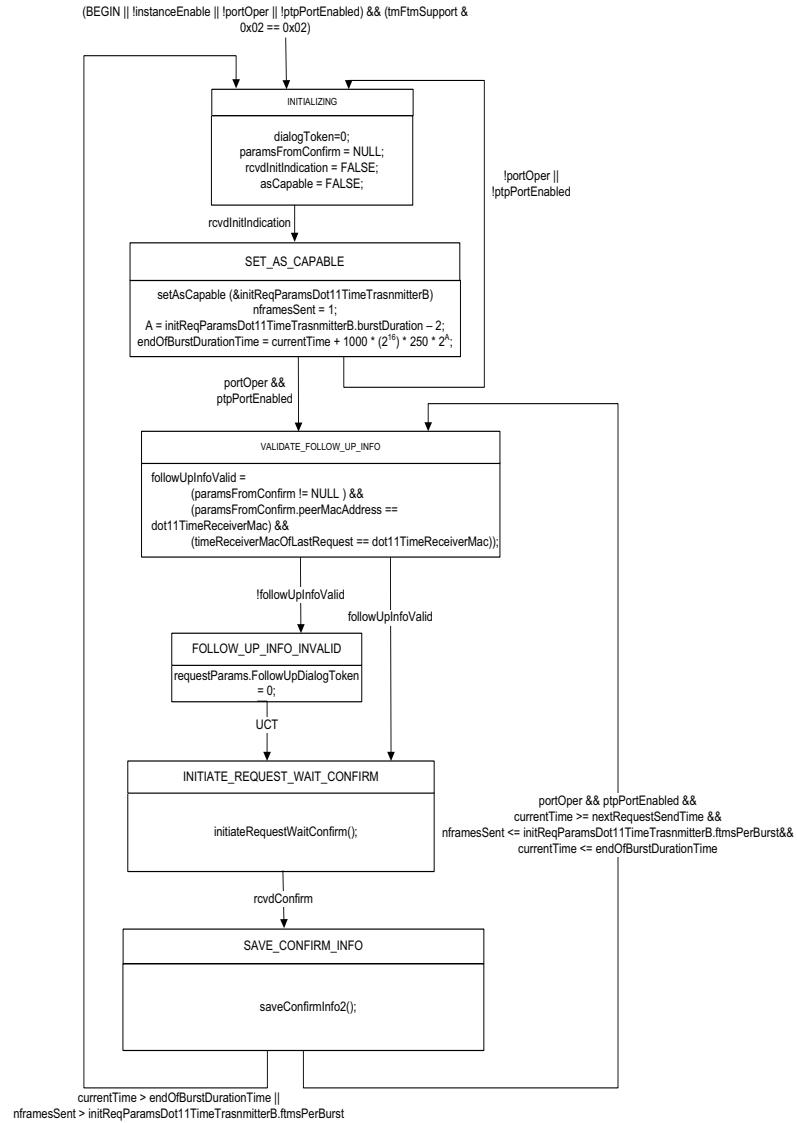


Figure 12-6—TimeTransmitter state machine B

(a) For TM, not invoked and

(b) for FTM, receives initial FTM request from timeReceiver and sends information received from upstream to timeReceiver in successive FTM frames

2 12.5.1.3 State machine local variables

3 **12.5.1.3.1 dialogToken:** An unsigned 8-bit integer used to identify a measurement from among those preceding and following it.

5 **12.5.1.3.2 followUpInfoValid:** A Boolean variable indicating whether the FollowUpInformation (e.g., timestamp and rateRatio; see 12.7) and the link partner are unchanged since the last Timing Measurement or Fine Timing Measurement.

1 **12.5.1.3.3 requestParams:** A structure whose members contain the values of the fields of either the
2 MLME-TIMINGMSMT.request or MLME-FINETIMINGMSMT.request primitive.

3 **12.5.1.3.4 paramsFromConfirm:** A structure whose members contain the values of the fields of either the
4 MLME-TIMINGMSMT.confirm or MLME-FINETIMINGMSMT.confirm primitive.

5 **12.5.1.3.5 dot11TimeReceiverMac:** The MAC address of the station associated with the current port.

6 **12.5.1.3.6 timeReceiverMacOfLastRequest:** The MAC address of the station of the previous request, used
7 to validate FollowUpInformation.

8 **12.5.1.3.7 residenceTime:** A temporary variable that holds the computation of the time between receipt of
9 the last synchronization information and transmission of synchronization information.

10 **12.5.1.3.8 rcvMDSyncDot11TimeTransmitterA:** A Boolean variable that is set to TRUE when an
11 MDSyncSend structure is provided by the PortSync entity.

12 **12.5.1.3.9 rcvDConfirm:** A Boolean variable that is set to TRUE when either the MLME-
13 TIMINGMSMT.confirm or MLME-FINETIMINGMSMT.confirm primitive is received.

14 **12.5.1.3.10 nframesSent:** An unsigned 8-bit integer used to count the number of frames sent by the
15 timeReceiver (the number of indications received is counted).

16 **12.5.1.3.11 rcvDInitIndication:** A Boolean variable that is set to TRUE when the initial MLME-
17 FINETIMINGMSMTRQ.indication primitive is received.

18 **12.5.1.3.12 initReqParamsDot11TimeTransmitterB:** A structure whose members contain the values of
19 the fields of an MLME-FINETIMINGMSMTRQ.indication primitive.

20 **12.5.1.3.13 endOfBurstDurationTime:** A UScaledNs variable whose value is the time at which the current
21 burst ends.

22 **12.5.1.3.14 nextRequestSendTime:** A UScaledNs variable whose value is the expected time that the next
23 MLME-FINETIMINGMSMTRQ.indication primitive, for the next request from the timeReceiver, will be
24 received.

25 **12.5.1.4 State machine functions**

26 **12.5.1.4.1 setRequestParams(&requestParams, MDSyncSend):** Assigns values to the parameters of the
27 request primitive of either MLME-TIMINGMSMT or MLME-FINETIMINGMSMT (see 12.5.1.6) as
28 follows:

29 a) Members of the FollowUpInformation member of the VendorSpecific information element, as
30 defined in 12.7, are assigned as defined in 11.4.4 with the exception of the correctionField, which is
31 assigned, for TM, by the function saveConfirmInfo1() in TimeTransmitter state machine A (see
32 Figure 12-5) and, for FTM, by the function saveConfirmInfo2() in TimeTransmitter state machine B
33 (see Figure 12-6).

34 b) The other fields of the VendorSpecific information element are assigned as follows:
35 1) ElementID is assigned the value 221 as defined in Table 9-77 (Element IDs) of
36 IEEE Std 802.11-2016, indicating that the information element is of type Vendor Specific.
37 2) The Length field is set to 80.

38 NOTE—This is equal to the length of the Follow_Up payload defined in 11.4.4 (including the common header) plus the
39 length of the OUI or CID field and the Type field (see Figure 12-8).

- 1 3) The OUI or CID field is set to 00-80-C2.
 - 2 4) The Type field is set to 0.
 - 3 c) Max t1 Error, Maxt4 Error are set to zero.
 - 4 d) For Fine Timing Measurement frames, the location configuration information (LCI) Report and
 - 5 Location Civic Report are not present. All other members are left unchanged.

12.5.1.4.2 setAsCapable (&initReqParamsDot11TimeTransmitterB): Determines the value of asCapable consistent with 12.4 and whether the timeTransmitter is able to grant the parameters requested by the timeReceiver. This function is used only for FTM.

9 12.5.1.4.3 initiateRequestWaitConfirmOrSaveInfo(): This function is defined as indicated below. It is used in TimeTransmitter state machine A. It is defined here so that the detailed code that it invokes does not need to be placed into the state machine diagram.

```
12 initiateRequestWaitConfirmOrSaveInfo()
13 {
14     rcvdMDSyncDot11TimeTransmitterA = FALSE;
15
16     If (tmFtmSupport == 0x01)
17     {
18         if ((++dialogToken % 256) == 0) dialogToken++;
19         requestParams.DialogToken=dialogToken;
20         requestParams.PeerMACAddress = dot11TimeReceiverMac;
21         setRequestParams(&requestParams, MDSyncSend);
22         MLME-TIMINGMSMT.request(requestParams);
23         requestParams.FollowUpDialogToken = 0;
24         //In case no confirm is received
25         timeReceiverMacOfLastRequest = dot11TimeReceiverMac;
26     }
27 }
```

28 **12.5.1.4.4 saveConfirmInfo1()**: This function is defined as indicated below. It is used in TimeTransmitter
29 state machine A. It is defined here so that the detailed code that it invokes does not need to be placed into the
30 state machine diagram.

```
31     saveConfirmInfo1()
32     {
33         MLME-TIMINGMSMT.confirm(&paramsFromConfirm);
34
35         requestParams.FollowUpDialogToken = paramsFromConfirm.DialogToken;
36         requestParams.T1 = paramsFromConfirm.T1;
37         requestParams.T4 = paramsFromConfirm.T4;
38
39         // NOTE: In Timing Measurement, T1 is in units of 10 ns.
40         // upstreamTxTime is units of 216 ns.
41
42         K = 1;
43         // K is 1 for Timing Measurement.
44         residenceTime = MDSyncSend.rateRatio *
45             (paramsFromConfirm.T1 * 10K*(216) - MDSyncSend.upstreamTxTime);
46
47         requestParams.VendorSpecific.correctionField =
48             residenceTime + MDSyncSend.followUpCorrectionField;
49         // NOTE: T1 and T4 are timestamps from a single
50         // local clock source. The roll-over of the 32-bit timestamps returned by
51         // MLME-TIMINGMSMT.request and MLME-TIMINGMSMT.indication
```

```

1           //      must be accounted for.
2       }

```

3 **12.5.1.4.5 initiateRequestWaitConfirm()**: This function is defined as indicated below. It is used in
4 TimeTransmitter state machine B. It is defined here so that the detailed code that it invokes does not need to
5 be placed into the state machine diagram.

```

6     initiateRequestWaitConfirm()
7     {
8         If ((++dialogToken % 256) == 0) dialogToken++;
9         If (nframesSent == initReqParamsDot11TimeTransmitterB.ftmsPerBurst
10    framesPerBurst)
11             dialogToken = 0;
12
13         requestParams.DialogToken=dialogToken;
14         requestParams.PeerMACAddress = dot11TimeReceiverMac;
15         setRequestParams(&requestParams, MDSyncSend);
16         // In the following statement, MinDeltaFTM, which is in units of 100
17         // microseconds, is converted to UScaledNs (i.e., units of  $2^{-16}$  ns; see 6.3.3.2)
18         nextRequestSendTime = currentTime +
19             initReqParamsDot11TimeTransmitterB.MinDeltaFTM * (65536 x 105);
20         MLME-FINETIMINGMSMT.request(requestParams);
21         requestParams.FollowUpDialogToken = 0; //In case no confirm is received
22         timeReceiverMacOfLastRequest = dot11TimeReceiverMac;
23     }

```

24 **12.5.1.4.6 saveConfirmInfo2()**: This function is defined as indicated below. It is used in TimeTransmitter
25 state machine B. It is defined here so that the detailed code that it invokes does not need to be placed into the
26 state machine diagram.

```

27     saveConfirmInfo2()
28     {
29         MLME-FINETIMINGMSMT.confirm(&paramsFromConfirm);
30
31         requestParams.FollowUpDialogToken = paramsFromConfirm.DialogToken;
32         requestParams.T1 = paramsFromConfirm.T1;
33         requestParams.T4 = paramsFromConfirm.T4;
34
35         // NOTE: In Fine Timing Measurement, T1 is in units of 0.1 ns.
36         // upstreamTxTime is units of  $2^{-16}$  ns.
37
38         K = -3;
39         // K is 1 for Timing Measurement and -3 for Fine Timing Measurement.
40         residenceTime = MDSyncSend.rateRatio *
41             (paramsFromConfirm.T1 * 10K( $2^{16}$ ) - MDSyncSend.upstreamTxTime);
42
43         requestParams.VendorSpecific.correctionField =
44             residenceTime + MDSyncSend.followUpCorrectionField;
45         // NOTE: T1 and T4 are timestamps from a single
46         // local clock source. The roll-over of the 48-bit timestamps returned by
47         // MLME-FINETIMINGMSMT.request and MLME-FINETIMINGMSMT.indication
48         // must be accounted for.
49
50         // A frame is only counted as being sent, for purposes of number of frames in a
51         // burst, if a confirm is received. It is up to IEEE Std 802.11-2016 to handle the
52         // case where a confirm is not received.
53         nframesSent += 1;
54     }

```

1 **12.5.1.5 Shared variables**

- 2 **12.5.1.5.1 MDSyncSend:** A structure as defined in 10.2.2.1.
- 3 **12.5.1.5.2 portOper:** A Boolean as defined in 10.2.5.12.
- 4 **12.5.1.5.3 ptpPortEnabled:** A Boolean as defined in 10.2.5.13.
- 5 **12.5.1.5.4 asCapable:** A Boolean whose value is specified in 12.4 and 10.2.5.1.
- 6 **12.5.1.5.5 tmFtmSupport:** An Octet whose bits are interpreted as Booleans and whose values are specified in Table 12-1.
- 8 **12.5.1.5.6 ftmReqGrantedTimeTransmitter:** A Boolean whose value is TRUE if the timeTransmitter grants the current initial FTM request and FALSE otherwise.

10 **12.5.1.6 TimeTransmitter primitives**

11 **12.5.1.6.1 MLME-TIMINGMSMT.request**

12 The MLME-TIMINGMSMT.request primitive is used by a timeTransmitter station to initiate a Timing Measurement and also communicates timestamps t1 and t4 captured by the timeTransmitter during a previous measurement. The primitive and its parameters are specified in 6.3.57.2 of IEEE Std 802.11-2016.

15 **12.5.1.6.2 MLME-TIMINGMSMT.confirm**

16 The MLME-TIMINGMSMT.confirm primitive indicates that a Timing Measurement request has completed. The primitive and its parameters are specified in 6.3.57.3 of IEEE Std 802.11-2016.

18 **12.5.1.6.3 MLME-FINETIMINGMSMT.request**

19 The MLME-FINETIMINGMSMT request primitive is used by a timeTransmitter station to initiate a Fine Timing Measurement and also communicates timestamps t1 and t4 captured by the timeTransmitter during a previous measurement. The primitive and its parameters are specified in 6.3.58.2 of IEEE Std 802.11-2016.

22 **12.5.1.6.4 MLME-FINETIMINGMSMT.confirm**

23 The MLME-FINETIMINGMSMT request primitive indicates that a Fine Timing Measurement request has completed. The primitive and its parameters are specified in 6.3.58.3 of IEEE Std 802.11-2016.

25 **12.5.1.6.5 MLME-FINETIMINGMSMTRQ.indication**

26 The MLME-FINETIMINGMSMTRQ.indication primitive indicates to a timeTransmitter station that the timeReceiver is requesting a burst of FTM frames with the indicated parameters. The primitive and its parameters are specified in 6.3.70.3 of IEEE Std 802.11-2016.

29 **12.5.2 Media-dependent timeReceiver state machine**

30 **12.5.2.1 Overview**

31 The MD entity of an IEEE 802.11 port whose PTP Port state is TimeReceiverPort or PassivePort (see 10.3.6) shall behave in a manner that is indistinguishable, relative to an observer external to a system, from a strict implementation of the timeReceiver state machine in 12.5.2.2, the local variables specified in 12.5.2.3, the

1 functions specified in 12.5.2.4, the shared variables specified in 12.5.2.5, and the primitives defined in
2 12.5.2.6.

3 The timeReceiver state machine is responsible for collecting information from the Timing measurement or
4 Fine Timing measurement indications, constructing an MDSyncReceive structure with the relevant
5 information, and passing the structure to the PortSync entity for further processing. In order to do this, the
6 state machine saves locally captured timestamps (i.e., t2 and t3) received in the indication and associates
7 them with the timestamps sent from the timeTransmitter port in a future indication (i.e., t1 and t4). In
8 addition, for Fine Timing measurement, the timeReceiver state machine is responsible for generating the
9 MLME-FINETIMINGMSMTRQ.request primitive, which causes the initial FTM request frame to be sent
10 to the timeTransmitter.

11 **12.5.2.2 State diagram**

12 Figure 12-7 presents the timeReceiver state machine. While quantities are shown to be computed from
13 information in consecutive indications, an implementation can choose to compute over longer intervals as
14 long as the clock performance requirements of Annex B are met.

1

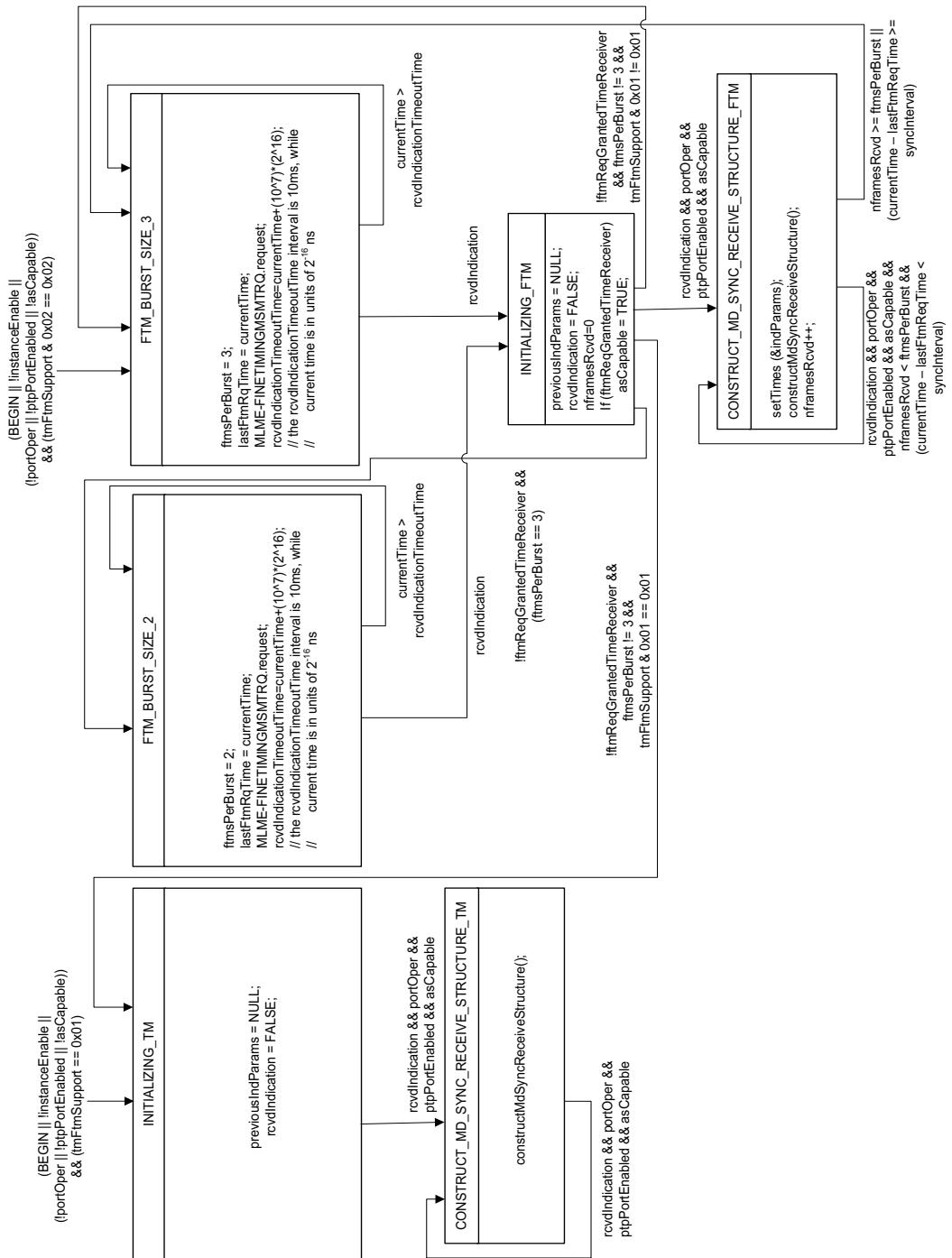


Figure 12-7—TimeReceiver state machine

1 12.5.2.3 State machine local variables

2 **12.5.2.3.1 indParams:** A structure whose members contain the values of the fields of the MLME-
 3 TIMINGMSMT.indication primitive or MLME-FINETIMINGMSMT.indication primitive, as defined in
 4 12.5.2.6, depending on whether TM or FTM, respectively, is used.

5 **12.5.2.3.2 previousIndParams:** A structure with members identical to those of indParams, used to save
 6 parameters from the previous indication.

7 **12.5.2.3.3 rcvIndication:** A Boolean that is set to TRUE when either the MLME-
 8 TIMINGMSMT.indication or MLME-FINETIMINGMSMT.indication primitive is received.

9 **12.5.2.3.4 RESTART:** A Boolean that indicates, when FTM is being used, that a new burst should be
 10 initiated.

11 **12.5.2.3.4 rcvIndicationTimeoutTime:** A UScaledNs variable whose value is the time after which the
 12 state machine will not wait any longer for the next MLME-FINETIMINGMSMT.indication, and a new burst
 13 is initiated.

14 **12.5.2.3.5 nframesRcvd:** An unsigned 8-bit integer used to count the number of frames received by the
 15 timeTransmitter in the burst (the number of indications received from the timeTransmitter are counted).

16 **12.5.2.3.6 initReqParamsDot11TimeReceiver:** A structure whose members contain the values of the fields
 17 of an MLME-FINETIMINGMSMTRQ.request~~indication~~ primitive.

18 **12.5.2.3.7 ftmsPerBurst:** The value of the FTM parameter ‘FTMs per burst’ (see 12.6), i.e., the number of
 19 FTM frames in the burst granted by the timeTransmitter.

20 **12.5.2.3.8 ftmReqGrantedTimeReceiver:** A Boolean that is TRUE if the timeTransmitter has granted the
 21 respective request for a burst and FALSE otherwise.

22 **12.5.2.3.9 t1_1, t1_2, t2_1, t2_2, t2_3, t3_1, t3_2, t3_3, t4_1, t4_2:** Temporary local variables used to hold
 23 the values of the timestamps t1, t2, t3, and t4 returned by the MLME-FINETIMINGMSMT.indication
 24 primitive in the structure indParams.

25 **12.5.2.3.11 D1, D2:** Temporary local variables used to hold the values of delay computed for the first two
 26 FTM frames and corresponding Ack when the timeTransmitter grants the request for three FTM frames.

27 12.5.2.4 State machine functions

28 **12.5.2.4.1 setMDSyncReceiveDot11TimeReceiver(indParams):** Creates an MDSyncReceive structure
 29 and returns the structure. All fields are assigned from FollowUpInformation (contained in the
 30 VendorSpecific information element) of indParams as in 11.2.14.2.1.

31 **12.5.2.4.2 passMDSyncReceiveToPortSync():** Passes an MDSyncReceive structure to the PortSync entity
 32 of this PTP Port.

33 **12.5.2.4.3 setTimes (&indParams):** extracts the timestamp values from the successive indParams
 34 structures returned by the multiple FTM frame exchanges of a burst, and places the correct times
 35 (corresponding to minimum delay frames and Ack) in the final indParams structure. This final indParams
 36 structure is then used in the function constructMdSyncReceiveStructure(). This procedure is needed when
 37 the timeTransmitter grants three FTM frames for the burst. If the timeTransmitter grants only two FTM
 38 frames for the burst, the timestamp values returned in the indication primitive of the second frame are used.

1 The function setTimes is used in the TimeReceiver state machine and is defined here so that the detailed
 2 code that it invokes does not need to be placed into the state machine diagram.

```

3      setTimes (&indParams)
4      {
5          if (nframesRcvd == 1)
6          {
7              t2_1 = indParams.T2;
8              t3_1 = indParams.T3;
9          }
10         if (nframesRcvd == 2)
11         {
12             t1_1 = indParams.T1;
13             t2_2 = indParams.T2;
14             t3_2 = indParams.T3;
15             t4_1 = indParams.T4
16         }
17         if (nframesRcvd == 3)
18         {
19             t1_2 = indParams.T1;
20             t2_3 = indParams.T2;
21             t3_3 = indParams.T3;
22             t4_2 = indParams.T4;
23             if (ftmsPerBurst == 3)
24             {
25                 D1 = t2_1 - t1_1;
26                 D2 = t2_2 - t1_2
27                 if (D2 <= D1)
28                 {
29                     indParams.T2 = t2_2;
30                     indParams.T1 = t1_2;
31                 }
32                 else
33                 {
34                     indParams.T2 = t2_1;
35                     indParams.T1 = t1_1;
36                 }
37                 D1 = t4_1 - t3_1;
38                 D2 = t4_2 - t3_2
39                 if (D2 <= D1)
40                 {
41                     indParams.T4 = t4_2;
42                     indParams.T3 = t3_2;
43                 }
44                 else
45                 {
46                     indParams.T4 = t4_1;
47                     indParams.T3 = t3_1;
48                 }
49             }
50         }
51     }

```

52 **12.5.2.4.4 constructMdSyncReceiveStructure():** This function constructs the MDSyncReceive structure
 53 and is defined as indicated below. It is used in the TimeReceiver state machine. It is defined here so that the
 54 detailed code that it invokes does not need to be placed into the state machine diagram.

```

55     constructMdSyncReceiveStructure()
56     {

```

```

1      if (tmFtmSupport == 0x01)
2          MLME-TIMINGMSMT.indication(&indParams);
3      else if (tmFtmSupport & 0x02 == 0x02)
4      {
5          MLME-FINETIMINGMSMT.indication(&indParams);
6          nframesRevd++;
7          if (nframesRevd == initReqParamsDot11TimeReceiver.framesPerBurst) ||
8              (currentTime > endofBurstDurationTime)
9              RESTART=1;
10     }
11
12     if ((previousIndParams != NULL) &&
13         (previousIndParams.PeerMacAddress == dot11TimeReceiverMac) &&
14         (indParams.FollowUpDialogToken != 0))
15     {
16
17         neighborRateRatio =
18             (indParams.T1-previousIndParams.T1) /
19             (indParams.T2-previousIndParams.T2);
20         //NOTE: Other methods of computing neighborRateRatio
21         can be used.
22
23         if (tmFtmSupport == 0x01)
24             K = 1;
25         else if (tmFtmSupport & 0x02 == 0x02)
26             K = -3;
27         //K = 1 for Timing Measurement and K = -3 for Fine Timing Measurement
28         meanLinkDelay =
29             (((indParams.T4 - indParams.T1) -
30                 neighborRateRatio * (indParams.T3 - indParams.T2)) /
31                 (2.0)) * (10K);
32
33         //NOTE: Other methods of computing meanLinkDelay
34         can be used.
35
36         MDSyncReceive = setMDSyncReceiveDot11TimeReceiver(indParams);
37         MDSyncReceive.VendorSpecific.rateRatio +=
38             (neighborRateRatio - 1);
39         MDSyncReceive.VendorSpecific.upstreamTxTime =
40             indParams.T2* (216) * (10K) -
41             meanLinkDelay*(216)/neighborRateRatio;
42         //NOTE: Actions performed with the timestampError
43         parameters of indParams are implementation independent.
44
45         passMDSyncReceiveToPortSync(&MDSyncReceive);
46     }
47     previousIndParams = indParams;
48     rcvdlIndication = FALSE;
49 }
```

50 12.5.2.5 State machine shared variables

51 **12.5.2.5.1 MDSyncReceive:** A structure used for passing information between MD and PortSync, as
 52 defined in 10.2.2.2.

53 **12.5.2.5.2 portOper:** A Boolean as defined in 10.2.5.12.

54 **12.5.2.5.3 ptpPortEnabled:** A Boolean as defined in 10.2.5.

- 1 **12.5.2.5.4 asCapable:** A Boolean as defined in 12.4 and 10.2.5.1.
- 2 **12.5.2.5.5 meanLinkDelay:** The delay over the link to the associated WLAN station as defined in 10.2.5.8.
- 3 **12.5.2.5.6 neighborRateRatio:** The measured ratio of the frequency of the LocalClock entity of the time-aware system at the other end of the link attached to this PTP Port, to the frequency of the LocalClock entity of this time-aware system, as defined in 10.2.5.7.
- 6 **12.5.2.5.7 tmFtmSupport:** An Octet whose value is specified in 12.3.

7 **12.5.2.6 TimeReceiver primitives**

8 **12.5.2.6.1 MLME-TIMINGMSMT.indication**

9 The MLME-TIMINGMSMT.indication primitive is received by a timeReceiver station as the natural result
 10 of the peer timeTransmitter station issuing the corresponding request primitive and carries the same
 11 parameters plus local timestamp information. The primitive and its parameters are specified in 6.3.57.4 of
 12 IEEE Std 802.11-2016.

13 **12.5.2.6.2 MLME-FINETIMINGMSMT.indication**

14 The MLME-FINETIMINGMSMT.indication primitive is received by a timeReceiver station as the natural
 15 result of the peer timeTransmitter station issuing the corresponding request primitive and carries the same
 16 parameters plus local timestamp information. The primitive and its parameters are specified in 6.3.58.4 of
 17 IEEE Std 802.11-2016.

18 **12.5.2.6.3 MLME-FINETIMINGMSMTRQ.request**

19 The MLME-FINETIMINGMSMTRQ.request primitive is used by the timeReceiver to request a burst of
 20 FTM frames from the timeTransmitter, with respective FTM parameters. The primitive and its parameters
 21 are specified in 6.3.70.2 of IEEE Std 802.11-2016.

22 **12.6 FTM parameters**

23 The values of the FTM parameters that are relevant to time synchronization transport in this standard are
 24 specified in Table 12-2, along with a brief description of each parameter. These parameter values are carried
 25 in the initial FTM Request. More detailed descriptions of these and other FTM parameters are given in
 26 9.4.2.168 of IEEE Std 802.11-2016. The parameters Burst Duration and Min Delta FTM depend on the time
 27 synchronization message interval, i.e., on currentLogSyncInterval (see 10.7.2.3 and 12.8). The values for
 28 these parameters are specified in Table 12-3. In this table, the Burst Duration encoding (i.e., value and
 29 corresponding duration in ms) is taken from Table 9-257 of IEEE Std 802.11-2016. The Min Delta FTM
 30 encoding values are in multiples of 100 μ s (see 9.4.2.168 of IEEE Std 802.11-2016).

31 The values of the FTM parameters given in Table 12-2 shall be used in the MLME-
 32 FINETIMINGMSMTRQ.request invoked by the timeReceiver STA (i.e., in the initial FTM Request). The
 33 values for Burst Duration and Min Delta FTM given in Table 12-3 shall be used in the MLME-
 34 FINETIMINGMSMTRQ.request invoked by the timeReceiver STA.

35 Background on the derivation of the FTM parameters of Table 12-2 and Table 12-3 is given in Garner [B4].

1

Table 12-2—FTM parameters relevant to time-synchronization transport

Parameter	Value	Description
Number of Bursts Exponent	0	Log to base 2 of the number of bursts requested by the timeReceiver (value of 0 indicates that one burst is requested)
Burst Duration	See Table 12-3	Duration of the burst of FTM frames and their corresponding Acknowledgments
Min Delta FTM	See Table 12-3	Minimum time between consecutive FTM frames
Partial TSF Timer	1	See 9.4.2.168 of IEEE Std 802.11-2016
Partial TSF Timer No Preference	Reserved in the initial FTM request	See 9.4.2.168 of IEEE Std 802.11-2016
ASAP	1	ASAP = 1 indicates that the timeReceiver would like the timeTransmitter to respond as soon as possible
ASAP Capable	Reserved in the initial FTM request	See 9.4.2.168 of IEEE Std 802.11-2016
FTMs per burst	3 in the first initial FTM Request 2 in the first retry, if the first initial FTM Request is not granted	Desired number of FTM frames and corresponding Acknowledgments in the requested burst
Burst Period	Reserved when Number of Bursts Exponent is zero	See 9.4.2.168 of IEEE Std 802.11-2016

2

Table 12-3—Values of Burst Duration and Min Delta FTM, for each value of currentLogSyncInterval

currentLogSyncInterval	Nominal message rate (messages/s)	Burst duration	Min delta FTM
-24 through -6, inclusive	64 through 16 777 216	6 (4 ms)	6 (0.6 ms)
-5	32	8 (16 ms)	25 (2.5 ms)
-4	16	9 (32ms)	50 (5 ms)
-3	8	10 (64 ms)	100 (10 ms)
-2 through 24, inclusive	4	11 (128 ms)	200 (20 ms)

1 12.7 Format of VendorSpecific information element

2 The IEEE 802.11 MLME request and indication primitives for Timing Measurement and Fine Timing
 3 Measurement support an ability to carry data transparently between stations using the VendorSpecific
 4 information element. The Type field within the VendorSpecific Content identifies the type of information
 5 that follows the Type field. See Figure 12-8 and Table 12-4.

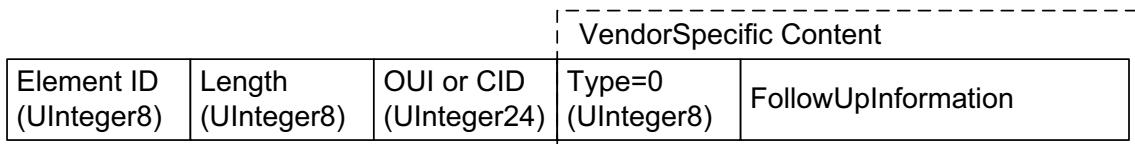


Figure 12-8—Format of VendorSpecific information element when Type = 0

Table 12-4—Values of the Type field in the VendorSpecific information element

Value	Description
0	The Type field is followed by FollowUp-Information
1–255	Reserved

6 This mechanism shall be used to carry end-to-end link-independent timing information from the
 7 timeTransmitter port to the associated timeReceiver port, including preciseOriginTimestamp, rateRatio,
 8 correctionField, and other fields of the Follow-Up message, as described in 12.5.1.4. For consistency, all of
 9 these fields are packed into the FollowUpInformation field using exactly the same format as used for full-
 10 duplex point-to-point links. In other words, the timeTransmitter state machine communicates an entire
 11 Follow_Up message [i.e., including all the fields of the common header (see 11.4.2 and 10.6.2), the
 12 preciseOriginTimestamp, and all the fields of the Follow_Up information TLV (see 11.4.4)] using this
 13 mechanism. The Type field, illustrated in Figure 12-8, identifies this use of the OUI or CID within the
 14 VendorSpecific information element. Table 12-4 lists values for the Type field.

15 12.8 Synchronization message interval

16 12.8.1 General synchronization message interval specification

17 The mean time interval between successive synchronization messages shall be as specified in 10.7.2.1,
 18 10.7.2.3, and 12.8.2.

19 12.8.2 Synchronization message interval default value

20 The default value of initialLogSyncInterval (see 10.7.2.3) is -3. Every PTP Port supports the value 127; the
 21 PTP Port does not send Sync messages when currentLogSyncInterval has this value (see 10.3.18). A PTP
 22 Port may support other values, except for the reserved values indicated in Table 10-16. A PTP Port ignores
 23 requests (see 10.3.18) for unsupported values.

24 Processing of the message interval request TLV carried in a Signaling message (see 10.6.4) shall be
 25 supported, as specified by the SyncIntervalSetting state machine of 10.3.18 (see Figure 10-20), except that

1 the logLinkDelayInterval (which is not relevant to IEEE 802.11 ports) is set to -128 by the sender of the
2 Signaling message, the logLinkDelayInterval is ignored by the receiver, and unsupported values of
3 logTimeSyncInterval are ignored by the receiver.

4 NOTE 1—For TM, a timeReceiver port that requests (using a Signaling message that contains a message interval
5 request TLV; see 10.6.4 and 10.3.18) that the PTP Port at the other end of the attached link set its
6 currentLogSyncInterval to a specific value can determine if the request was honored by examining the
7 logMessageInterval field of a FollowUpInformation contained in the VendorSpecific information element of a
8 subsequent MLME indication primitive.

9 NOTE 2—The time interval between every pair of adjacent timing or Fine Timing Measurements is not guaranteed to be
10 precisely the same. Some variation is expected, due to factors such as the MAC protocol (e.g., delay in accessing the
11 medium and/or packet retries) and the power state of the associated station. However, timestamp t1 is not captured when
12 either TIMINGMSMT.request or FINETIMINGMSMT.request is invoked, but only after the frame resulting from the
13 request is actually transmitted.

1 13. Media-dependent layer specification for interface to IEEE 802.3 Ethernet 2 passive optical network link

3 13.1 Overview

4 13.1.1 General

5 This clause specifies the service interface primitives, state machines, and message formats that provide
6 accurate synchronized time across IEEE 802.3 Ethernet passive optical network (EPON) links, through the
7 use of the timing process and measurements specified in 64.2.1.1, 64.3.2.4, 77.2.1.1, and 76.1.2 of
8 IEEE Std 802.3-2018. For purposes of this clause, an EPON link is an EPON that contains one optical line
9 terminal (OLT) and associated optical network units (ONUs).

10 A time-aware system may contain more than one OLT and/or ONU. Each PTP Instance of a time-aware
11 system uses at most one ONU port, but may serve, i.e., provide timing to, more than one OLT port (i.e., each
12 PTP Instance of a time-aware system is a clock timeReceiver to at most one EPON link, but can be clock
13 timeTransmitter to more than one EPON link). Two different PTP Instances of a time-aware system may use
14 different ONU ports.

15 13.1.2 Description of the EPON timing process

16 The timing process in EPON relies on the 32-bit counters (see 64.2.2.2 and 77.2.2.2 of IEEE Std
17 802.3-2018) at both the OLT and the ONU. The 32-bit counter used by EPON is the LocalClock entity of the
18 PTP Instance that uses the respective OLT or ONU. These counters increment every time_quantum, which is
19 equal to 16 ns (see 64.2.2.1 and 77.2.2.1 of IEEE Std 802.3-2018). IEEE Std 802.3-2018 defines multipoint
20 control protocol (MPCP), which is one of the protocols that enable MAC clients to communicate over a
21 point-to-multipoint optical network. When either the clock timeTransmitter (OLT) or the clock timeReceiver
22 (ONU) transmits an MPCP data unit (MPCPDU), its counter value is mapped into the timestamp field.
23 Clause 64 and Clause 77 of IEEE Std 802.3-2018 specify the EPON timing mechanism.

24 13.1.3 Best timeTransmitter selection

25 13.1.3.1 General

26 An EPON link contains one OLT and the associated ONUs. The OLT is the clock timeTransmitter and the
27 associated ONUs are clock timeReceivers. The OLT initiates the time synchronization as a requester. The
28 ONUs are the responders of the time synchronization. In other words, the invocation of the BTCA results in
29 the OLT having the PTP Port state TimeTransmitterPort and the ONU having the PTP Port state
30 TimeReceiverPort (see 10.3.1.1 and Table 10-2), for all PTP Instances using these PTP Ports, regardless of
31 the attributes of PTP Instances downstream from the ONU. This behavior is achieved using the acceptable
32 timeTransmitter table feature defined in 17.5 of IEEE Std 1588-2019.

33 A PTP Instance that contains an ONU port shall maintain a configured table, the
34 acceptableTimeTransmitterTable, and a per-PTP Port Boolean variable
35 acceptableTimeTransmitterTableEnabled. The data type of acceptableTimeTransmitterTable is
36 AcceptableTimeTransmitterTable (see 13.1.3.2).

13.1.3.2 AcceptableTimeTransmitterTable

2 The AcceptableTimeTransmitterTable type represents a table of AcceptableTimeTransmitter entries.

```
3 struct AcceptableTimeTransmitterTable{
4     UIInteger16 maxTableSize;
5     UIInteger16 actualTableSize;
6     AcceptableTimeTransmitter[actualTableSize] acceptableTimeTransmitter;
7 }
```

8 The maxTableSize member is the maximum size of the AcceptableTimeTransmitterTable. The
 9 actualTableSizeMember is the actual size of the AcceptableTimeTransmitterTable. The
 10 AcceptableTimeTransmitter array contains a list of AcceptableTimeTransmitter PTP Ports. The value of
 11 maxTableSize is implementation specific. actualTableSize shall be less than or equal to maxTableSize.

12 An AcceptableTimeTransmitterTable is configurable and may contain a number of
 13 AcceptableTimeTransmitter entries up to maxTableSize.

13.1.3.3 AcceptableTimeTransmitter

15 The AcceptableTimeTransmitter type represents a PTP Port that can be considered, in the execution of the
 16 BTCA, as a candidate for timeTransmitter.

```
17 struct AcceptableTimeTransmitter{
18     PortIdentity acceptablePortIdentity;
19     UIInteger8 alternatePriority1;
20 }
```

21 The acceptablePortIdentity member is the PortIdentity of an acceptable timeTransmitter port. The
 22 alternatePriority1 member contains an alternate value for the priority1 attribute of the acceptable
 23 timeTransmitter port (see 13.1.3.4).

13.1.3.4 Acceptable timeTransmitter table feature

25 The acceptable timeTransmitter table feature shall modify the operation of the BTCA (see 10.3) as follows:

26 a) If acceptableTimeTransmitterTableEnabled for a PTP Port is FALSE, the BTCA operates as
 27 described in 10.3.

28 b) If acceptableTimeTransmitterTableEnabled for a PTP Port is TRUE, then the following apply:

29 1) The function qualifyAnnounce() of the PortAnnounceReceive state machine (see 10.3.11.2.1)
 30 is replaced by the following:
 31

32 **qualifyAnnounce (rcvdAnnouncePtr):** qualifies the received Announce message pointed to
 33 by rcvdAnnouncePtr as follows:

34 i) if the Announce message was sent by the current PTP Instance, i.e., if
 35 sourcePortIdentity.clockIdentity (see 10.6.2.2.11 and 8.5.2) is equal to thisClock (see
 36 10.2.4.22), the Announce message is not qualified, and FALSE is returned;

37 ii) if the stepsRemoved field is greater than or equal to 255, the Announce message is not
 38 qualified, and FALSE is returned;

39 iii) if the sourcePortIdentity of the Announce message is not equal to the sourcePortIdentity
 40 of one of the entries of the acceptableTimeTransmitterTable, FALSE is returned;

41 iv) if a path trace TLV is present and one of the elements of the pathSequence array field of
 42 the path trace TLV is equal to thisClock (i.e., the clockIdentity of the current PTP
 43 Instance; see 10.2.4.22), the Announce message is not qualified, and FALSE is returned;

- otherwise, the Announce message is qualified, and TRUE is returned. If a path trace TLV is present, it is saved in the per port global variable receivedPathTrace. If a path trace TLV is not present, the per port global variable receivedPathTrace is set to the empty array.
- 2) If the alternatePriority1 member of the AcceptableTimeTransmitter array element that corresponds to the sourcePortIdentity of a received Announce message is 0, the alternatePriority1 member has no effect on the operation of the BTCA.
 - 3) If the alternatePriority1 member of the AcceptableTimeTransmitter array element that corresponds to the sourcePortIdentity of a received Announce message is greater than 0, the value of the grandmasterPriority1 field of the Announce message is replaced by the value of alternatePriority1 of this AcceptableTimeTransmitter array element for use in the invocation of the BTCA.

13 13.1.3.5 Default configuration of acceptable timeTransmitter table feature

14 The default configuration of the acceptable timeTransmitter table feature for a PTP Instance that is attached
15 to an IEEE 802.3 EPON link shall be as follows:

- 16 a) If the PTP Instance does not contain an ONU port, the default acceptableTimeTransmitterTable is
17 empty, i.e., the member actualTableSize is 0 and there are no AcceptableTimeTransmitter array
18 entries. The variable acceptableTimeTransmitterTableEnabled for each PTP Port is set to FALSE.
- 19 b) If the PTP Instance contains an ONU port, the default acceptableTimeTransmitterTable contains one
20 element in the AcceptableTimeTransmitter array. The member actualTableSize is 1. The
21 acceptablePortIdentity of that element is set equal to the portIdentity of the OLT port that the ONU
22 port is attached to, and alternatePriority1 set equal to 244. The variable
23 acceptableTimeTransmitterTableEnabled for each PTP Port is set to TRUE.

24 NOTE—These default settings ensure that, with the default priority1 values of 8.6.2.1, Table 8-1, used for all PTP
25 Instances, the PTP Instance that contains the ONU port will consider Announce messages only from the OLT that the
26 ONU port is attached to when invoking the BTCA. The alternatePriority1 value of 244 ensures that the OLT will be
27 considered better than the ONU in the sense of the BTCA, which will cause the OLT port state to be set to
28 TimeTransmitterPort and the ONU port state to be set to TimeReceiverPort. All other PTP Ports of this PTP Instance that
29 are not disabled and for which asCapable is TRUE will have PTP Port states of either TimeTransmitterPort or
30 PassivePort. If all PTP Instances downstream from the ONU have priority1 greater than 244, then the PTP Port at the
31 other end of each link attached to each non-ONU port that is not disabled and for which asCapable is TRUE will have
32 PTP Port states of either TimeReceiverPort or PassivePort; in this case, the downstream network portions will get their
33 timing through the EPON. However, if a downstream PTP Instance has priority1 less than 244, or priority1 equal to 244
34 and is better than the Grandmaster PTP Instance information contained in the Announce message received by the ONU
35 based on other attributes, then the portion of the network that is downstream of the ONU and includes that better PTP
36 Instance will get its timing from that better downstream PTP Instance. In this case, the endpoints of the link of that
37 network portion attached to the PTP Instance that contains the ONU will both have PTP Port states of
38 TimeTransmitterPort, and the PTP Ports at each end of the link will send Announce messages. However, the Announce
39 messages sent by the downstream PTP Instance will be ignored by the PTP Instance that contains the ONU because the
40 sourcePortIdentity of those Announce messages will not be contained in the acceptableTimeTransmitterTable. The
41 Announce messages sent by the PTP Instance that contains the ONU will be used in the invocation of the BTCA at the
42 downstream PTP Instance; however, those Announce messages will not reflect the best timeTransmitter because one of
43 the downstream PTP Instances is better.

44 13.1.4 Time synchronization in EPON

45 Transmission in the EPON downstream direction (from OLT to ONUs) utilizes time division multiplexing
46 (TDM). In the upstream direction (from ONUs to OLT), time division multiple access (TDMA) is employed.
47 Due to the frame queuing in TDMA, the downstream delay is different from the upstream delay.
48 Asymmetric delay also occurs in the EPON physical layer due to upstream and downstream transmission
49 using different wavelengths. The index of refraction is frequency dependent, which results in the upstream
50 and downstream delays being asymmetric. The accurate time synchronization across the EPON links is

1 operated as follows. It is assumed that the clock timeTransmitter (the OLT) has an accurate synchronized
 2 time. The clock timeTransmitter informs the clock timeReceiver (the ONU) what the accurate synchronized
 3 time will be when the counter of the clock timeReceiver reaches a certain value. The information transfer
 4 can be accomplished using the organization-specific slow protocol (OSSP) message (see Clause 57 of IEEE
 5 Std 802.3-2018).

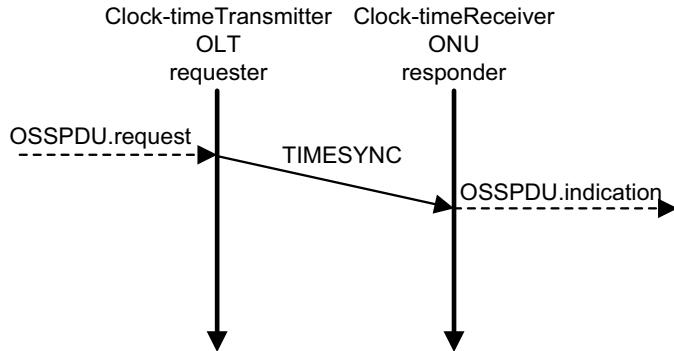


Figure 13-1—IEEE 802.3 EPON time-synchronization interfaces

6 The following reference process, illustrated schematically in Figure 13-1, will result in the clock
 7 timeReceiver of an ONU being synchronized to the clock timeTransmitter of the OLT:

- 8 a) The clock timeTransmitter selects a value X of the local MPCP counter that is used as the timing
 9 reference. Any value can be chosen, provided it is relative to the current epoch of the MPCP counter.
 10 b) The clock timeTransmitter calculates the $ToD_{X,i}$ based on $ToD_{X,o}$ using Equation (13-1).

$$11 \quad ToD_{X,i} = ToD_{X,o} + RTT_i \times \frac{ndown}{(nup + ndown)} \times rateRatio \quad (13-1)$$

12 where $ToD_{X,i}$ is the synchronized time when the MPCP counter at the clock timeReceiver i reaches a
 13 value equal to the timestamp X minus the $onuLatencyFactor$; $ToD_{X,o}$ is the synchronized time when
 14 the MPCP counter at the clock timeTransmitter reaches a value equal to the timestamp X plus the
 15 $oltLatencyFactor$; RTT_i is the round-trip time measured by the clock timeTransmitter for clock
 16 timeReceiver i , i.e., ONU i ; nup is the effective refraction index of the light propagating in the
 17 upstream channel; $ndown$ is the effective refraction index of the light propagating in the downstream
 18 channel; and $rateRatio$ is the $rateRatio$ member of the most recently received MDSyncSend
 19 structure. The $onuLatencyFactor$ and $oltLatencyFactor$ are given in Equation (13-2) and
 20 Equation (13-3), respectively. The impact of the worst-case variation in the transmission wavelength
 21 for the clock timeTransmitter and clock timeReceiver transmitters is examined in appendix VII of
 22 ITU-T G.984.3, Amendment 2 (11/2009).

$$23 \quad onuLatencyFactor = onuIngressLatency - \\ (onuIngressLatency + onuEgressLatency) \times \frac{ndown}{(nup + ndown)} \times rateRatio \quad (13-2)$$

$$24 \quad oltLatencyFactor = oltEgressLatency - \\ (oltIngressLatency + oltEgressLatency) \times \frac{ndown}{(nup + ndown)} \times rateRatio \quad (13-3)$$

- 1 c) The clock timeTransmitter sends the pair of values (X , $ToD_{X,i}$) to clock timeReceiver i via the
2 downstream TIMESYNC message.

3 NOTE—After the clock timeReceiver receives the downstream TIMESYNC message, it can compute the synchronized
4 time, ToD , when the value of the local MPCP counter is equal to S ; ToD is given by the following equation:

$$5 \quad ToD = ToD_{X,i} + [(S - X) \bmod (2^{32})] (16 \text{ ns}) (\text{rateRatio})$$

6 where $(A) \bmod (B)$ is A modulo B .

7 The OSSP message is a general message (see 3.10), analogous to Follow_Up. Note that the preceding
8 synchronized time values correspond to timestamps that are referenced to the MAC control sublayer. Both
9 the clock timeTransmitter and clock timeReceiver are responsible for compensating their processing delays
10 (e.g., the ingressLatency and egressLatency, as described in 8.4.3). RTT_i is measured using MPCPDU
11 timestamps, inserted into the frame structure as specified by 64.2.1.1 and 77.2.1.1 of IEEE Std 802.3-2018.

12 **13.2 Message attributes**

13 **13.2.1 Message class**

14 The TIMESYNC message is a general message (see 3.10 and 8.4.2.2). It is transmitted in the downstream
15 direction, from OLT to ONU.

16 **13.3 Message format**

17 **13.3.1 TIMESYNC message**

18 **13.3.1.1 General TIMESYNC message specifications**

19 The fields of the body of the TIMESYNC message shall be as specified in Table 13-1 and 13.3.1.2.

20 **13.3.1.2 TIMESYNC message field specifications**

21 **13.3.1.2.1 Destination address (Octet6)**

22 The destination address field is equal to 01-80-C2-00-00-02 (see 57A.3 of IEEE Std 802.3-2018).

23 **13.3.1.2.2 Source address (Octet6)**

24 The source address field is the individual MAC address associated with the port through which the
25 TIMESYNC message is transmitted (see 57B.1.1 of IEEE Std 802.3-2018).

26 **13.3.1.2.3 Length/Type (Octet2)**

27 The value of this field is equal to 0x8809 (see 57A.4 of IEEE Std 802.3-2018).

Table 13-1—TIMESYNC message fields

Bits								Octets	Offset				
7	6	5	4	3	2	1	0						
Destination Address								6	0				
Source Address								6	6				
Length/Type								2	12				
Subtype								1	14				
OUI or CID								3	15				
Message Identifier								2	18				
<i>X</i>								4	20				
$ToD_{X,i}$								10	24				
sourcePortIdentity								10	34				
logMessageInterval								1	44				
rateRatio								8	45				
gmTimeBaseIndicator								2	53				
lastGmPhaseChange								12	55				
scaledLastGmFreqChange								4	67				
domainNumber								1	71				
majorSdoId				reserved				1	72				
minorSdoId								1	73				
reserved								0	74				
FCS								4					

1 13.3.1.2.4 Subtype (Octet)

2 The value of this field is equal to 0x0A (see 57A.4 of IEEE Std 802.3-2018).

3 13.3.1.2.5 OUI or CID (Octet3)

4 This field contains the OUI or CID that identifies the Organization-Specific Data. The value is 00-80-C2,
 5 i.e., the OUI assigned to IEEE 802.1.

6 13.3.1.2.6 Message identifier (Octet2)

7 This field is the TIMESYNC message identifier. The value of this field is 1.

8 13.3.1.2.7 X (UInteger32)

9 The *X* field is the selected timestamp that will be used as the timing reference as specified in 13.1.4.

1 13.3.1.2.8 *ToD_{X,i}* (Timestamp)

2 *ToD_{X,i}* is the synchronized time when the MPCP counter at the clock timeReceiver *i* reaches a value equal to
 3 *X* minus the *onuLatencyFactor* (see 13.1.4). *X* is carried in the respective TIMESYNC message.
 4 Synchronization of the MPCP clock is described in detail in 64.2.1.1 and 77.2.1.1 in IEEE Std 802.3-2018,
 5 for 1G-EPON and 10G-EPON, respectively.

6 NOTE—Any subnanosecond portion of synchronized time (in this case, time of day), normally transported in a
 7 correction field (see 10.2.2.1.2, 10.2.2.2.2, and 10.2.2.3.5), is not transported over EPON.

8 13.3.1.2.9 sourcePortIdentity (PortIdentity)

9 This field is specified as the sourcePortIdentity member of the MDSyncSend structure most recently
 10 received from the PortSync entity of the OLT (see 10.2.2.1.4).

11 13.3.1.2.10 logMessageInterval (Integer8)

12 This field is specified as the logMessageInterval member of the MDSyncSend structure most recently
 13 received from the PortSync entity of the OLT (see 10.2.2.1.5). It is the value of the currentLogSyncInterval
 14 for this PTP Port (see 10.7.2.3).

15 13.3.1.2.11 rateRatio (Float64)

16 This field is specified as the rateRatio member of the MDSyncSend structure most recently received from
 17 the PortSync entity of the OLT (see 10.2.2.1.8).

18 13.3.1.2.12 gmTimeBaseIndicator (UInteger16)

19 This field is specified as the gmTimeBaseIndicator member of the MDSyncSend structure most recently
 20 received from the PortSync entity of the OLT (see 10.2.2.1.9).

21 13.3.1.2.13 lastGmPhaseChange (ScaledNs)

22 This field is specified as the lastGmPhaseChange member of the MDSyncSend structure most recently
 23 received from the PortSync entity of the OLT (see 10.2.2.1.10).

24 13.3.1.2.14 scaledLastGmFreqChange (Integer32)

25 The value of scaledLastGmFreqChange is the fractional frequency offset of the current Grandmaster Clock
 26 relative to the previous Grandmaster Clock, at the time that the current Grandmaster PTP Instance became
 27 the Grandmaster PTP Instance, or relative to itself prior to the last change in gmTimeBaseIndicator,
 28 multiplied by 2^{41} and truncated to the next smaller signed integer. The value is obtained by multiplying the
 29 lastGmFreqChange member of MDSyncSend (see 10.2.2.1) whose receipt causes the MD entity to send the
 30 TIMESYNC message by 2^{41} , and truncating to the next smaller signed integer.

31 NOTE—The above scaling allows the representation of fractional frequency offsets in the range $[-(2^{-10} - 2^{-41}), 2^{-10} -$
 32 $2^{-41}]$, with granularity of 2^{-41} . This range is approximately $[-9.766 \times 10^{-4}, 9.766 \times 10^{-4}]$.

33 13.3.1.2.15 domainNumber (UInteger8)

34 This field is specified as the gPTP domainNumber (see 8.1).

1 13.3.1.2.16 majorSdId (Nibble)

2 The value is the same as the value specified in 8.1 for all transmitted PTP messages of a gPTP domain. Any
 3 TIMESYNC message received for which the value is not one of the values specified in 8.1 shall be ignored.

4 NOTE—The nibble that immediately follows majorSdId is reserved (see 10.6.1).

5 13.3.1.2.17 minorSdId (UInteger8)

6 The value is the same as the value specified in 8.1 for all transmitted PTP messages of a gPTP domain. Any
 7 TIMESYNC message received for which the value is not one of the values specified in 8.1 shall be ignored.

8 13.3.1.2.18 reserved

9 The reserved field that follows minorSdId has variable length. The field shall have zero length on
 10 transmission. Any bytes between the minorSdId and the FCS field shall be ignored on reception.

11 13.3.1.2.19 FCS (Octet4)

12 This field is the frame check sequence (see 57B.1.1 of IEEE Std 802.3-2018).

13 13.4 Determination of asCapable

14 The default value of the per-PTP Port, per-domain global variable asCapable shall be TRUE.

15 The per-PTP Port, per-domain global variable asCapable shall be set to TRUE if the value of
 16 neighborGptpCapable for this PTP Port is TRUE.

17 NOTE—The above conditions ensure backward compatibility with the 2011 edition of this standard. A time-aware
 18 system that is compliant with the 2011 edition of this standard will not process the gPTP capable TLV, and asCapable
 19 will be determined as specified in the 2011 edition. A PTP Instance of a time-aware system compliant with the current
 20 edition of this standard that is attached, via an EPON link, to a node compliant with the 2011 edition of this standard will
 21 not receive Signaling messages that contain the gPTP capable TLV and will not set neighborGptpCapable to TRUE.
 22 However, the above ensures that asCapable for this PTP Port and domain (i.e., domain 0) will still be set in a manner
 23 consistent with that of the 2011 edition of this standard because the default value of asCapable is TRUE in that edition.

24 13.5 Layering for IEEE 802.3 EPON links

25 The MD entity is media-dependent and is responsible for translating the media-independent layer to media-
 26 dependent PDUs or primitives as necessary for communicating synchronized time over the EPON link from
 27 the OLT to a single ONU. This implies that if one OLT port is associated with multiple ONUs, it will require
 28 one IEEE 802.1AS PortSync entity and one MD entity per associated ONU. The OSSPDU primitives are
 29 used to communicate synchronized time information. Figure 13-2 illustrates how the MD entity interacts
 30 with the OSSP sublayer.

31

32

33

34

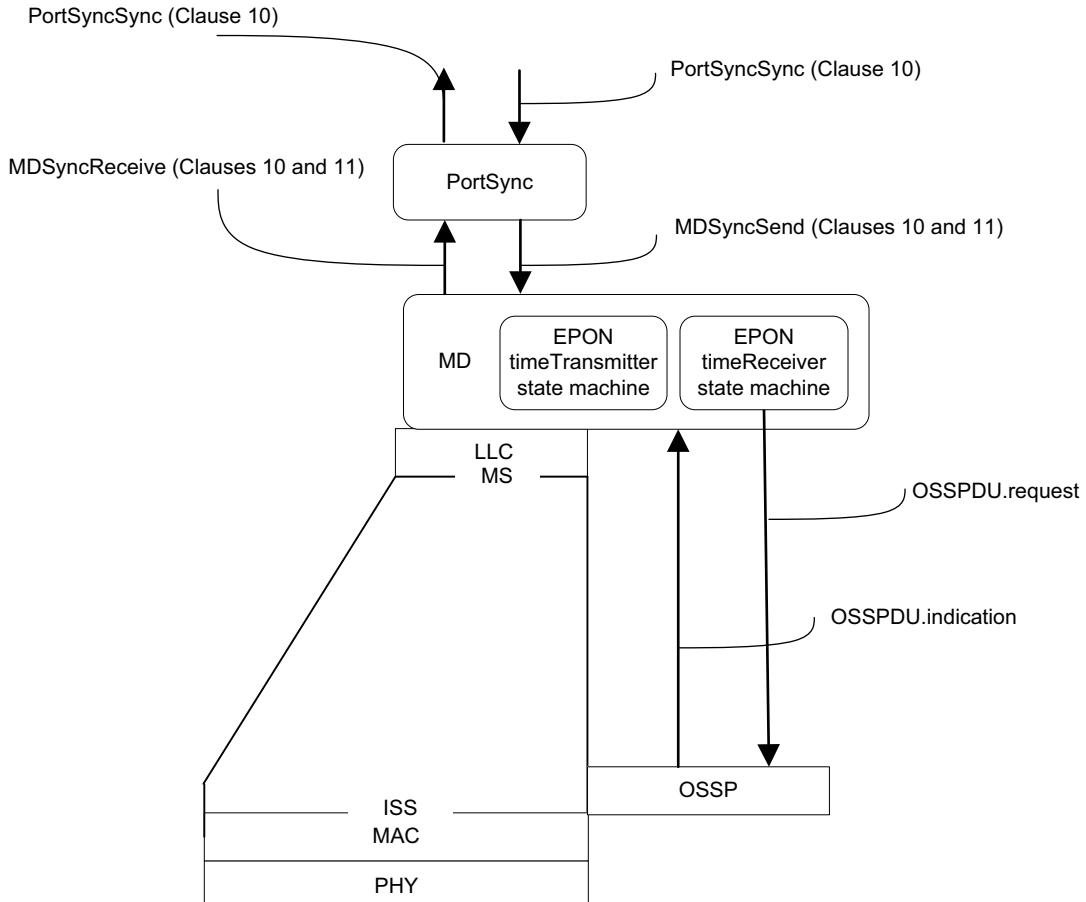


Figure 13-2—IEEE 802.3 EPON interface model

1

2

3 13.6 Service interface definitions

4 13.6.1 OSSPDU.request

5 13.6.1.1 General

6 This service interface primitive is generated periodically by the MD entity of the clock timeTransmitter
 7 every sync interval (see 10.7.2.1). It triggers transmission of a TIMESYNC message from the clock
 8 timeTransmitter to the clock timeReceiver. The values of the parameters of the primitive are sent to the
 9 clock timeReceiver via the TIMESYNC message.

1 13.6.1.2 OSSPDU.request parameters

2 13.6.1.2.1 Syntax of the primitive

```

3 OSSPDU.request {
4     X
5     ToDX,i
6     sourcePortIdentity
7     logMessageInterval
8     rateRatio
9     gmTimeBaseIndicator
10    lastGmPhaseChange
11    scaledLastGmFreqChange
12    domainNumber
13    sdold
14 }
```

15 13.6.1.2.2 X (Integer32)

16 The *X* field is the selected timestamp that will be used as the timing reference as specified in 13.1.4.

17 13.6.1.2.3 ToD_{X,i} (Timestamp)

18 *ToD_{X,i}* is the synchronized time when the MPCP counter at the clock *timeReceiver i* reaches a value equal to 19 *X* minus the *onuLatencyFactor* (see 13.1.4). *X* is carried in the respective TIMESYNC message. 20 Synchronization of the MPCP clock is described in detail in 64.2.1.1 and 77.2.1.1 in IEEE Std 802.3-2018, 21 for 1G-EPON and 10G-EPON, respectively.

22 13.6.1.2.4 sourcePortIdentity (PortIdentity)

23 This parameter identifies the sourcePortIdentity value for this PTP Port (see 13.3.1.2.9).

24 13.6.1.2.5 logMessageInterval (Integer8)

25 This parameter identifies the currentLogSyncInterval value for this PTP Port (see 13.3.1.2.10).

26 13.6.1.2.6 rateRatio (Float64)

27 This parameter identifies the rateRatio value for this PTP Port (see 13.3.1.2.11).

28 13.6.1.2.7 gmTimeBaseIndicator (UInteger16)

29 This parameter identifies the gmTimeBaseIndicator value for this PTP Port (see 13.3.1.2.12).

30 13.6.1.2.8 lastGmPhaseChange (ScaledNs)

31 This parameter identifies the lastGmPhaseChange value for this PTP Port (see 13.3.1.2.13).

32 13.6.1.2.9 scaledLastGmFreqChange (Integer32)

33 This parameter identifies the scaledLastGmFreqChange value for this PTP Port (see 13.3.1.2.14).

1 13.6.1.2.10 domainNumber

2 This parameter identifies the domainNumber for this instance of gPTP (see 13.3.1.2.15).

3 13.6.1.2.11 sdoId

4 This parameter identifies the sdoId for this instance of gPTP (see 13.3.1.2.16 and 13.3.1.2.17).

5 13.6.1.3 When generated

6 This primitive is generated by the clock timeTransmitter every $2^{\text{currentLogSyncInterval}}$ seconds when it is in the
7 TIME_TRANSMITTER state, as the first phase of synchronized time information transfer.

8 13.6.1.4 Effect of receipt

9 Upon receipt of this primitive, a TIMESYNC message is enqueued for transmission.

10 NOTE—Arrival of the TIMESYNC message at the ONU after the selected time X does not impede proper operation of
11 the synchronization mechanism defined in this clause.

12 13.6.2 OSSPDU.indication

13 13.6.2.1 General

14 This service interface primitive is generated on receipt of a TIMESYNC message by the responder, and
15 provides the values contained in the corresponding OSSPDU.request primitive to the clock timeReceiver.

16 13.6.2.2 OSSPDU.indication parameters

```
17 OSSPDU.indication {
18     X
19     ToDX,i
20     sourcePortIdentity
21     logMessageInterval
22     rateRatio
23     gmTimeBaseIndicator
24     lastGmPhaseChange
25     scaledLastGmFreqChange
26     domainNumber
27     sdoId
28 }
```

29 The parameters of the OSSPDU.indication are set equal to the corresponding fields of the most recently
30 received TIMESYNC message. Their definitions are given in 13.3.1.2.7 through 13.3.1.2.17, respectively.

31 13.6.2.3 When generated

32 This primitive is generated by the receipt of a TIMESYNC message during the phase of synchronized time
33 information transfer.

34 13.6.2.4 Effect of receipt

35 Upon receipt, the OSSPDU.indication parameters are used by the MD entity to compute the parameters of
36 the MDSyncReceive structure that will be transmitted to the PortSync entity of this PTP Port.

1 13.7 MD entity global variables

2 **13.7.1 RTT_i :** Is used only by the OLT MD entity. RTT_i is the RTT between the clock timeTransmitter and
3 clock timeReceiver. The data type for RTT_i is UInteger32.

4 NOTE—RTT is measured and updated by the MPCP using the mechanism specified in IEEE Std 802.3-2018 and stored
5 in RTT_i when measured and updated. RTT_i is not used by the ONU and is set to zero in an ONU MD entity.

6 13.8 State machines

7 13.8.1 Requester state machine

8 13.8.1.1 Function

9 This state machine generates and consumes primitives, at the requester, used to provide accurate
10 synchronized time across EPON links to the responder.

11 13.8.1.2 State machine variables

12 The following variables are used in the state diagram in Figure 13-3 (in 13.8.1.4):

13 **13.8.1.2.1 ndown:** The effective index of the light propagating in the downstream channel. The data type for
14 ndown is Float64.

15 **13.8.1.2.2 nup:** The effective index of the light propagating in the upstream channel. The data type for
16 ndown is Float64.

17 **13.8.1.2.3 recvMDSyncEponReq:** A Boolean variable that notifies the current state machine when an
18 MDSyncSend structure is received. This variable is reset by the current state machine.

19 **13.8.1.2.4 recvMDSyncPtrEponReq:** A pointer to the received MDSyncSend structure.

20 **13.8.1.2.5 registered:** A Boolean variable that indicates an ONU has registered to EPON.

21 **13.8.1.2.6 $ToD_{X,i}$:** The synchronized time when the MPCP counter at the clock timeReceiver i reaches a
22 value equal to X (see 13.8.1.2.8) minus the *onuLatencyFactor* (see 13.1.4). The data type for $ToD_{X,i}$ is
23 Timestamp.

24 **13.8.1.2.7 $ToD_{X,o}$:** The synchronized time when the MPCP counter at the clock timeTransmitter reaches a
25 value equal to X (see 13.8.1.2.8) plus the *oltLatencyFactor* (see 13.1.4). The data type for $ToD_{X,o}$ is
26 Timestamp.

27 **13.8.1.2.8 X:** The value of the timestamp [see item a) of 13.1.4] that is selected as the reference time. The
28 data type for X is UInteger32.

29 13.8.1.3 State machine functions

30 **13.8.1.3.1 setToDXo():** Computes the state machine variable $ToD_{X,o}$ (see 13.8.1.2.7) as the sum of the
31 following:

- 32 a) The preciseOriginTimestamp member of the most recently received MDSyncSend structure
- 33 b) The followUpCorrectionField of the most recently received MDSyncSend structure

1 c) The quantity in Equation (13-4)

2 $\text{rateRatio} \times (X \times (16 \text{ ns}) - \text{upstreamTxTime})$ (13-4)

3 where

4 rateRatio is the rateRatio member of the most recently received MDSyncSend structure
 5 upstreamTxTime is the upstreamTxTime member of the most recently received MDSyncSend
 6 structure
 7 X is defined in 13.8.1.2.8

8 13.8.1.4 State diagram

9 The requester state machine shall implement the function specified by the state diagram in Figure 13-3, the local variables specified in 13.8.1.2, the function specified in 13.8.1.3, the service interface primitives specified in 13.6, the structure specified in 10.2.2.1, the message specified in 13.3, and the relevant global variables specified in 10.2.5 and 13.7. The state machine receives an MDSyncSend structure from the PortSyncSyncSend state machine of the PortSync entity of this PTP Port and transmits an OSSPDU.request primitive to cause a TIMESYNC message to be sent to the responder (ONU).

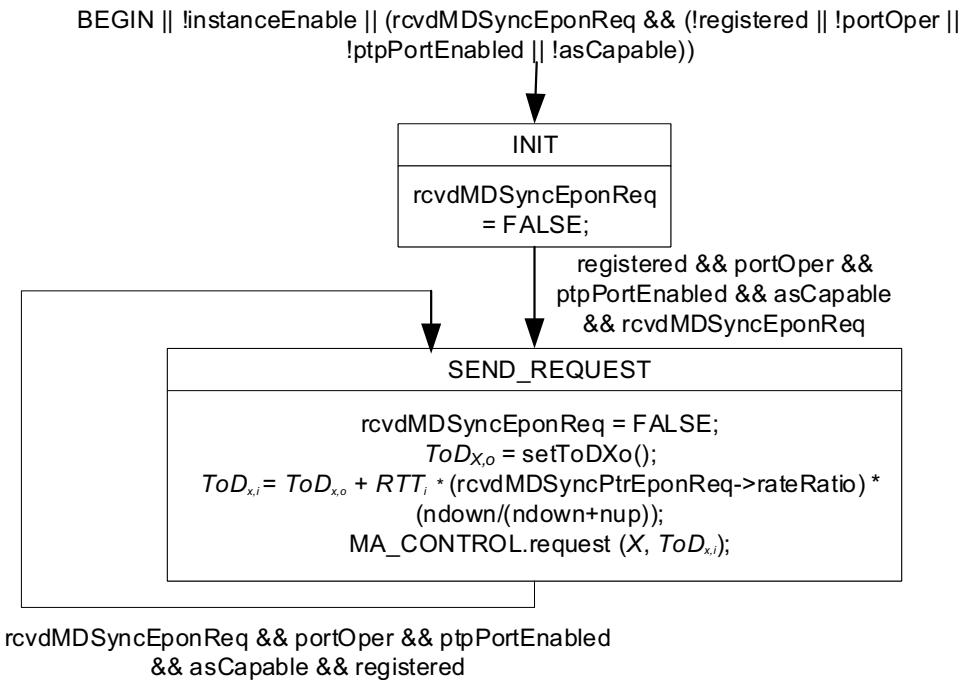


Figure 13-3—State machine for IEEE 802.3 EPON requester

15 13.8.2 Responder state machine

16 13.8.2.1 Function

17 This state machine responds to EPON-specific primitives generated by receipt of a TIMESYNC message
 18 from the requester.

1 13.8.2.2 State machine variables

2 The following variables are used in the state diagram in Figure 13-4 (in 13.8.2.4):

3 **13.8.2.2.1 rcvdOSSPDUind:** a Boolean variable that notifies the responder state machine when a
4 TIMESYNC message is received and the OSSPDU.indication primitive is generated.

5 **13.8.2.2.2 txMDSyncReceivePtrEponResp:** a pointer to a structure whose members contain the values of
6 the parameters of an MDSyncReceive structure to be transmitted.

7 **13.8.2.2.3 rcvdOSSPDUptr:** a pointer to a structure whose members contain the values of the parameters of
8 the OSSPDU.indication primitive whose receipt is indicated by rcvdOSSPDUind (see 13.8.2.2.1).

9 13.8.2.3 State machine functions

10 **13.8.2.3.1 setMDSyncReceiveEponResp():** creates an MDSyncReceive structure (see 10.2.2.2) using
11 members of the structure pointed to by rcvdOSSPDUptr (see 13.8.2.2.3), and returns a pointer to this
12 structure. The members of this structure are set as follows:

- 13 a) followUpCorrectionField is set equal to 0.
- 14 b) sourcePortIdentity is set equal to the sourcePortIdentity field of the most recently received
15 TIMESYNC message (see 13.3.1.2.9).
- 16 c) logMessageInterval is set equal to the logMessageInterval of the most recently received
17 TIMESYNC message (see 13.3.1.2.10).
- 18 d) preciseOriginTimestamp is set equal to the $ToD_{X,i}$ field of the most recently received TIMESYNC
19 message (see 13.3.1.2.8).
- 20 e) rateRatio is set to the rateRatio of the most recently received TIMESYNC message (see
21 13.3.1.2.11).
- 22 f) upstreamTxTime is set equal to X multiplied by 16 ns, where X is the value of the X field of the most
23 recently received TIMESYNC message (see 13.3.1.2.7).
- 24 g) gmTimeBaseIndicator is set equal to the gmTimeBaseIndicator of the most recently received
25 TIMESYNC message (see 13.3.1.2.12).
- 26 h) lastGmPhaseChange is set equal to the lastGmPhaseChange of the most recently received
27 TIMESYNC message (see 13.3.1.2.13).
- 28 i) lastGmFreqChange is set equal to the scaledLastGmFreqChange of the most recently received
29 TIMESYNC message (see 13.3.1.2.14), divided by 2^{41} .
- 30 j) domainNumber is set equal to the domainNumber of the most recently received TIMESYNC
31 message (see 13.3.1.2.15).

32 **13.8.2.3.2 txMDSyncReceive (txMDSyncReceivePtrEponResp):** transmits an MDSyncReceive structure
33 to the PortSyncSyncReceive state machine of the PortSync entity of this PTP Port.

34 13.8.2.4 State diagram

35 The responder state machine shall implement the function specified by the state diagram in Figure 13-4, the
36 local variables specified in 13.8.2.2, the functions specified in 13.8.2.3, the service interface primitives
37 specified in 13.6, the structure specified in 10.2.2.2, the message specified in 13.3, and the relevant global
38 variables specified in 10.2.5 and 13.7. The state machine receives an OSSPDU.indication primitive in
39 response to its having received a TIMESYNC message from the requester (OLT), and transmits an
40 MDSyncReceive structure to the PortSync entity of this PTP Port.

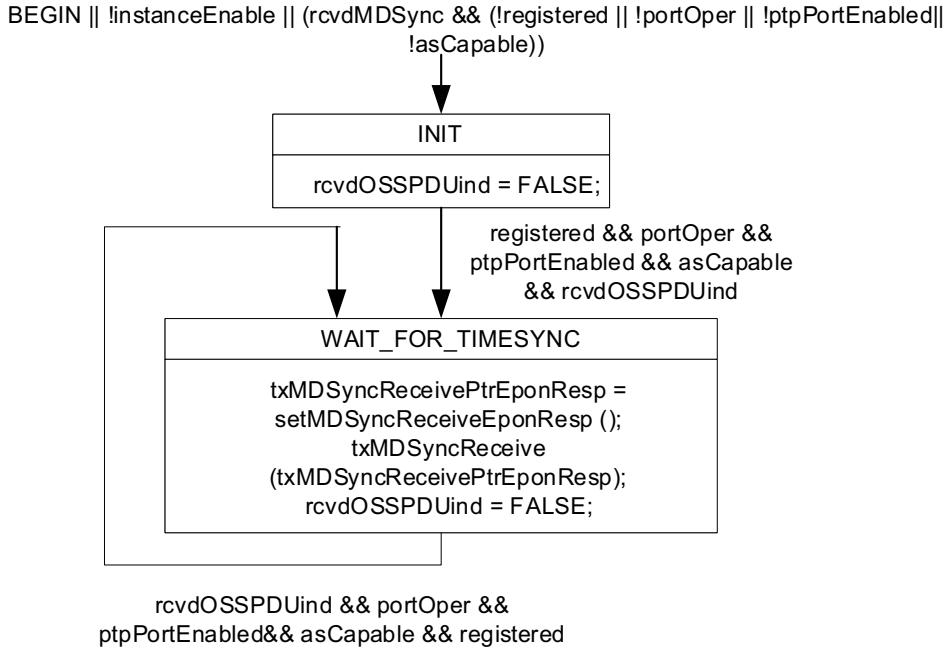


Figure 13-4—State machine for IEEE 802.3 EPON responder

1 13.9 Message transmission intervals

2 13.9.1 General interval specification

3 The mean time interval between successive TIMESYNC messages shall be as specified in 10.7.2.1, 10.7.2.3, 4 and 13.9.2.

5 13.9.2 TIMESYNC message transmission interval default value

6 The default value of initialLogSyncInterval (see 10.7.2.4) is -3. Every PTP Port supports the value 127; the 7 PTP Port does not send TIMESYNC messages when currentLogSyncInterval has this value. A PTP Port 8 may support other values, except for the reserved values indicated in Table 10-16.

9 Processing of the message interval request TLV carried in a Signaling message (see 10.6.4) shall be 10 supported, as specified by the SyncIntervalSetting state machine of 10.3.18 (see Figure 10-20), except that 11 the logLinkDelayInterval (which is not relevant to IEEE 802.3 EPON ports) is set to 128 by the sender of the 12 Signaling message, the logLinkDelayInterval is ignored by the receiver, and unsupported values of 13 logTimeSyncInterval are ignored by the receiver.

1 14. Timing and synchronization management

2 14.1 General

3 14.1.1 Data set hierarchy

4 This clause defines the set of managed objects, and their functionality, that allow administrative
5 configuration of clock parameters and timing and synchronization protocols.

6 Management data models typically represent data for the physical device (i.e., time-aware system). The
7 specifications for discovery, management address, and security for the physical device are typically covered
8 by standards of the management mechanism, which are outside the scope of this standard. For the
9 management information model of this standard, the scope of work is the data contained within a time-aware
10 system. From a management perspective, the time-aware system contains a list of one or more PTP
11 Instances. Each entry in the list is a set of managed data sets for the respective PTP Instance.

12 Conformance for each managed object is optional. This standard operates correctly using default values;
13 therefore, management is not essential. Since the management mechanism is not limited to remote protocols
14 (e.g., SNMP, NETCONF), management can use a local mechanism with a simple interface (e.g., DIP
15 switches). Therefore, each product can determine the support of managed objects as appropriate for its
16 management mechanism.

17 The following hierarchy summarizes the managed data sets within a gPTP Node:

- 18 a) instanceList[]
 - 19 1) defaultDS
 - 20 2) currentDS
 - 21 3) parentDS
 - 22 4) timePropertiesDS
 - 23 5) pathTraceDS
 - 24 6) acceptableTimeTransmitterTableDS
 - 25 7) **ptpInstanceSyncDS**
 - 26 8) **driftTrackingDS**
 - 27 9) portList[]
 - 28 i) portDS
 - 29 ii) descriptionPortDS
 - 30 iii) portStatisticsDS
 - 31 iv) acceptableTimeTransmitterPortDS
 - 32 v) externalPortConfigurationPortDS
 - 33 vi) asymmetryMeasurementModeDS
 - 34 vii) commonServicesPortDS
- 35 b) commonServices
 - 36 1) commonMeanLinkDelayService
 - 37 i) cmldsDefaultDS
 - 38 ii) cmldsLinkPortList[]
 - 39 — cmldsLinkPortDS
 - 40 — cmldsLinkPortStatisticsDS
 - 41 — cmldsAsymmetryMeasurementModeDS
 - 42 2) **hotStandbyService**

- 1 i) **hotStandbySystemList[]**
 2 — **hotStandbySystemDS**
 3 — **hotStandbySystemDescriptionDS**
 4 3) Future common services can follow.

5 The instanceList is indexed using a number that is unique per PTP Instance within the time-aware system,
 6 applicable to the management context only (i.e., not used in PTP messages). The domainNumber of the PTP
 7 Instance must not be used as the index to instanceList since it is possible for a time-aware system to contain
 8 multiple PTP Instances using the same domainNumber. The portList is indexed using a number that is
 9 unique per logical port (i.e., PTP Port) in the PTP Instance (see 8.5.1). Since the portNumber of a logical
 10 port can have any value in the range 1, 2, 3, ..., 0xFFFF (see 8.5.2.3), the portList index and portNumber
 11 values for a logical port **are** not necessarily the same. PTP Instances and logical ports may be created or
 12 deleted dynamically in implementations that support dynamic create/delete of devices. Unless otherwise
 13 indicated, the data sets and managed objects under the instanceList[] are maintained separately for each PTP
 14 Instance supported by the time-aware system.

15 Following the instanceList[] and all the data sets of each instanceList[] member is an overall structure for
 16 common services. That structure contains one sub-structure for each common service. At present there **are**
 17 **two** common services, namely the Common Mean Link Delay Service (CMLDS) **and the Hot Standby**
 18 **Service**. The corresponding sub-structures **are** the commonMeanLinkDelayService structure **and the**
 19 **hotStandbyService structure, respectively**. The item “Future common services can follow” is a placeholder
 20 for any common services that might be defined in the future. The commonMeanLinkDelayService structure
 21 **and the hotStandbyService structure** contain the data sets and lists that are needed by the Common Mean
 22 Link Delay Service and the Hot Standby Service, **respectively**.

23 The commonMeanLinkDelayService structure contains the cmldsLinkPortList, which is a list of CMLDS
 24 logical ports, i.e., Link Ports (see 11.2.17), of the time-aware system that **runs** the common service. The
 25 CMLDS must be implemented (i.e., a CMLDS executable must be present) on every physical port for which
 26 there is a PTP Port of a PTP Instance that can use the CMLDS (i.e., where portDS.delayMechanism of that
 27 PTP Instance can have the value COMMON_P2P). Therefore, the cmldsLinkPortList[] must include Link
 28 Ports that correspond to all such physical ports. As is the case for the portList of a PTP Instance, the
 29 cmldsLinkPortList is indexed using a number that is unique per Link Port that invokes the CMLDS
 30 (see 8.5.1). Since the portNumber of a logical port (i.e., PTP Port or CMLDS Link Port) can have any
 31 value in the range 1, 2, 3, ..., 0xFFFF (see 8.5.2.3), the cmldsLinkPortList index and
 32 cmldsLinkPortDS.portIdentity.portNumber values for a logical port of the Common Mean Link Delay
 33 Service **are** not necessarily the same. CMLDS Link Ports may be created or deleted dynamically in
 34 implementations that support dynamic create/delete of devices.

35 The Common Mean Link Delay Service Data Sets are not maintained separately for each PTP Instance.
 36 Rather, a single copy of the commonServices.cmldsDefaultDS is maintained for the time-aware system, and
 37 a single copy of each data set under the cmldsLinkPortList[] is maintained per Link Port of the time-aware
 38 system.

39 A PTP Instance can use the commonServicesPortDS to determine which Link Port it must use when it
 40 obtains information provided by the Common Mean Link Delay Service (see 14.14).

41 The hotStandbyService structure contains the hotStandbySystemList, which is a list of instances of the Hot
 42 Standby Service. Since a Hot Standby Service Instance is associated with two PTP Instances, but a
 43 time-aware system can have more than two PTP Instances, there can, in general, be multiple instances of the
 44 Hot Standby Service (i.e., one Hot Standby Service Instance associated with each set of two distinct PTP
 45 Instances). A hotStandbySystemDS and a hotStandbySystemDescriptionDS are maintained for each
 46 instance of the Hot Standby Service.

1 Each instance of the Hot Standby Service (i.e., each hotStandbySystemList member) is associated with two
 2 PTP Instances, i.e., the primary PTP Instance and the secondary PTP Instance (see 18.1 and 18.2). The
 3 indices of these PTP Instances in the instanceList are contained in the members primaryPtpInstanceIndex
 4 and secondaryPtpInstanceIndex, respectively, of the hotStandbySystemDS for this instance of the Hot
 5 Standby Service.

6 NOTE—This hierarchy is intended to support a wide variety of time-aware system implementations. Examples include
 7 the following:

- 8 a) A time-aware system containing four PTP Relay Instances, each of which use the same physical ports, but
 9 different domainNumber values.
- 10 b) A time-aware system containing four PTP Relay Instances and two hotStandbySystem entities, with two PTP
 11 Instances associated with one of the hotStandbySystem entities and the other two PTP Instances associated with
 12 the other hotStandbySystem entity.
- 13 c) A time-aware system that represents a chassis with slots for switch/router cards, where each switch/router card is
 14 represented as a PTP Instance using distinct physical ports and all PTP Instances can use the same
 15 domainNumber.

16 14.1.2 Data set descriptions

17 This management resource comprises the following objects:

- 18 a) The Default Parameter Data Set (defaultDS in 14.1.1; see Table 14-1), which represents the native
 19 capabilities of a PTP Instance, i.e., a PTP Relay Instance or a PTP End Instance station.
- 20 b) The Current Parameter Data Set (currentDS in 14.1.1; see Table 14-2), which represents the
 21 topological position of a local PTP Instance and other information, relative to the Grandmaster PTP
 22 Instance.
- 23 c) The Parent Parameter Data Set (parentDS in 14.1.1; see Table 14-3), which represents capabilities of
 24 the upstream PTP Instance toward the Grandmaster PTP Instance, as measured at a local PTP
 25 Instance.
- 26 d) The Time Properties Parameter Data Set (timePropertiesDS in 14.1.1; see Table 14-4), which
 27 represents capabilities of the Grandmaster PTP Instance, as measured at a local PTP Instance.
- 28 e) The Path Trace Parameter Data Set (pathTraceDS in 14.1.1; see Table 14-5), which represents the
 29 current path trace information (see 10.3.9.23) available at the PTP Instance.
- 30 f) The Acceptable TimeTransmitter Table Parameter Data Set (acceptableTimeTransmitterTableDS in
 31 14.1.1; see Table 14-6), which represents the acceptable timeTransmitter table used when an EPON
 32 port is used by a PTP Instance of a time-aware system.
- 33 g) The PTP Instance Synchronization Parameter Data Set (ptpInstanceSyncDS in 14.1.1; see Table 14-
 34 6a), which represents time synchronization status information for the PTP Instance.
- 35 h) The Drift Tracking Parameter Data Set (driftTrackingDS in 14.1.1; see Table 14-6b), which contains
 36 a manged object used to enable or disable the Drift_Tracking TLV.
- 37 i) The Port Parameter Data Set (portDS in 14.1.1; see Table 14-10), which represents time-aware
 38 capabilities at a given PTP Relay Instance or PTP End Instance port.
- 39 j) The Description Port Parameter Data Set (descriptionPortDS in 14.1.1; see Table 14-11), which
 40 contains the profileIdentifier for this PTP profile as specified in F.2.
- 41 k) The Port Parameter Statistics Data Set (portStatisticsDS in 14.1.1; see Table 14-12), which
 42 represents statistics and counters associated with time-aware capabilities at a given PTP Relay
 43 Instance or PTP End Instance port.
- 44 l) The Acceptable TimeTransmitter Port Parameter Data Set (acceptableTimeTransmitterPortDS in
 45 14.1.1; see Table 14-13), which represents the capability to enable/disable the acceptable
 46 timeTransmitter table feature on a PTP Port.

- 1 m) The External Port Configuration Port Parameter Data Set (externalPortConfigurationPortDS in
2 14.1.1; see Table 14-14), which is used with the external port configuration option to indicate the
3 desired state of a PTP Port.
- 4 n) The Asymmetry Measurement Mode Parameter Data Set (asymmetryMeasurementModeDS in
5 14.1.1; see Table 14-15), which represents the capability to enable/disable the Asymmetry
6 Compensation Measurement Procedure on a PTP Port (see Annex G) and is used instead of the
7 cmldsAsymmetryMeasurementModeDS when CMLDS is not used and there is a single gPTP
8 domain.
- 9 o) The Common Services Port Parameter Data Set (commonServicesPortDS in 14.1.1; see Table 14-
10 16), which enables a PTP Port of a PTP Instance to determine which Link Port of the respective
11 common service corresponds to that PTP Port.
- 12 p) The Common Mean Link Delay Service Default Parameter Data Set (cmldsDefaultDS in 14.1.1; see
13 Table 14-18), which describes the per-time-aware-system attributes of the Common Mean Link
14 Delay Service.
- 15 q) The Common Mean Link Delay Service Link Port Parameter Data Set (cmldsLinkPortDS in 14.1.1;
16 see Table 14-18), which represents time-aware Link Port capabilities for the Common Mean Link
17 Delay Service of a time-aware system.
- 18 r) The Common Mean Link Delay Service Link Port Parameter Statistics Data Set
19 (cmldsLinkPortStatisticsDS in 14.1.1; see Table 14-19), which represents statistics and counters
20 associated with Link Port capabilities at a given time-aware system.
- 21 s) The Common Mean Link Delay Service Asymmetry Measurement Mode Parameter Data Set
22 (cmldsAsymmetryMeasurementModeDS in 14.1.1; see Table 14-20), which represents the
23 capability to enable/disable the Asymmetry Compensation Measurement Procedure on a Link Port
24 (see Annex G).
- 25 t) The Hot Standby Service Hot Standby System Parameter Data Set (hotStandbySystemDS in 14.1.1;
26 see Table 14-17), which describes the attributes of the respective instance of the Hot Standby
27 Service.
- 28 u) The Hot Standby Service Hot Standby System Description Parameter Data Set
29 (hotStandbySystemDescriptionDS, see Table 14-22), which represents descriptive information for
30 the instance of the Hot Standby Service.

31 NOTE—portDS, descriptionPortDS, portStatisticsDS, and acceptableTimeTransmitterPortDS correspond to a logical
32 PTP Port of a PTP Instance; a PTP Relay Instance or PTP End Instance physical port can contain one or more logical
33 ports (see 8.5.1). For example, a PTP Relay Instance physical port can be connected to a full-duplex point-to-point link
34 that contains one logical port. As another example, a PTP Relay Instance physical port can be connected to a CSN link
35 that contains more than one logical port.

36 **14.2 Default Parameter Data Set (defaultDS)**

37 **14.2.1 General**

38 The defaultDS represents the native capabilities of a PTP Instance, i.e., a PTP Relay Instance or a PTP End
39 Instance.

40 **14.2.2 clockIdentity**

41 The value is the clockIdentity (see 8.5.2.2) of the PTP Instance.

42 **14.2.3 numberPorts**

43 The value is the number of PTP Ports of the PTP Instance (see 8.6.2.8). For an end station the value is 1.

1 **14.2.4 clockQuality**

2 **14.2.4.1 General**

3 This is a structure whose data type is ClockQuality (see 6.4.3.8).

4 **14.2.4.2 clockQuality.clockClass**

5 The value is the clockClass of the PTP Instance, which implements the clockClass specifications of 8.6.2.2.

6 **14.2.4.3 clockQuality.clockAccuracy**

7 The value is the clockAccuracy of the PTP Instance, which implements the clockAccuracy specifications of 8.6.2.3.

9 **14.2.4.4 clockQuality.offsetScaledLogVariance**

10 The value is the offsetScaledLogVariance of the PTP Instance, which implements the offsetScaledLogVariance specifications of 8.6.2.4.

12 **14.2.5 priority1**

13 The value is the priority1 attribute of the PTP Instance (see 8.6.2.1).

14 **14.2.6 priority2**

15 The value is the priority2 attribute of the PTP Instance (see 8.6.2.5).

16 **14.2.7 gmCapable**

17 The value is TRUE if the PTP Instance is capable of being a Grandmaster PTP Instance and FALSE if the PTP Instance is not capable of being a Grandmaster PTP Instance.

19 **14.2.8 currentUtcOffset**

20 The value is the offset between TAI and UTC, relative to the ClockTimeTransmitter entity of this PTP Instance. It is equal to the global variable sysCurrentUtcOffset (see 10.3.9.18). The value is in units of seconds.

23 The default value is selected as follows:

- 24 a) The value is the value obtained from a primary reference if the value is known at the time of initialization, else
- 25 b) The value is the current IERS defined value of TAI – UTC (see IERS Bulletin C) when the PTP Instance is designed.

28 **14.2.9 currentUtcOffsetValid**

29 The value is TRUE if the currentUtcOffset, relative to the ClockTimeTransmitter entity of this PTP Instance, is known to be correct. It is equal to the global variable sysCurrentUtcOffsetValid (see 10.3.9.14).

31 The default value is TRUE if the value of currentUtcOffset is known to be correct; otherwise, it is set to FALSE.

1 14.2.10 leap59

2 A TRUE value indicates that the last minute of the current UTC day, relative to the ClockTimeTransmitter
 3 entity of this PTP Instance, will contain 59 s. It is equal to the global variable sysLeap59 (see 10.3.9.13).

4 The value is selected as follows:

- 5 a) The value is obtained from a primary reference if known at the time of initialization, else
- 6 b) The value is set to FALSE.

7 14.2.11 leap61

8 A TRUE value indicates that the last minute of the current UTC day, relative to the ClockTimeTransmitter
 9 entity of this PTP Instance, will contain 61 s. It is equal to the global variable sysLeap61 (see 10.3.9.12).

10 The value is selected as follows:

- 11 a) The value is obtained from a primary reference if known at the time of initialization, else
- 12 b) The value is set to FALSE.

13 14.2.12 timeTraceable

14 The value is set to TRUE if the timescale and the value of currentUtcOffset, relative to the
 15 ClockTimeTransmitter entity of this PTP Instance, are traceable to a primary reference standard; otherwise
 16 the value is set to FALSE. It is equal to the global variable sysTimeTraceable (see 10.3.9.16).

17 The value is selected as follows:

- 18 a) If the time and the value of currentUtcOffset are traceable to a primary reference standard at the time
 of initialization, the value is set to TRUE, else
- 19 b) The value is set to FALSE.

21 14.2.13 frequencyTraceable

22 The value is set to TRUE if the frequency determining the timescale of the ClockTimeTransmitter Entity of
 23 this PTP Instance is traceable to a primary standard; otherwise the value is set to FALSE. It is equal to the
 24 global variable sysFrequencyTraceable (see 10.3.9.17).

25 The value is selected as follows:

- 26 a) If the frequency is traceable to a primary reference standard at the time of initialization the value is
 set to TRUE, else
- 27 b) The value is set to FALSE.

29 14.2.14 ptpTimescale

30 The value is set to TRUE if the clock timescale of the ClockTimeTransmitter Entity of this PTP Instance is
 31 PTP (see 8.2) and FALSE otherwise.

32 14.2.15 timeSource

33 The value is the source of time used by the Grandmaster Clock (see 8.6.2.7).

1 14.2.16 domainNumber

2 The value is the **domainNumber** of the gPTP domain for this instance of gPTP supported by the time-aware system (see 8.1).

4 NOTE—The PTP Instance for which domainNumber is 0 has constraints applied to it, e.g., timescale (see 8.2.1).

5 14.2.17 sdoId

6 The value is the sdoId of the gPTP domain for this instance of gPTP supported by the time-aware system (see 8.1).

8 NOTE—The attribute sdoId is specified as a 12-bit unsigned integer in 8.1. The data type for the managed object sdoId is UIInteger16 in Table 14-1, for compatibility with IEEE Std 1588-2019. The range of the managed object is limited to 10 12 bits; in addition, only the single value 0x100 is specified in this standard for the gPTP domain of a PTP Instance.

11 14.2.18 externalPortConfigurationEnabled

12 The value is the externalPortConfigurationEnabled attribute of the PTP Instance (see 10.3.9.24).

13 14.2.19 instanceEnable

14 The value is the instanceEnable attribute of the PTP Instance (see 10.2.4.24).

15 14.2.20 defaultDS table

16 There is one defaultDS table per PTP Instance of a time-aware system, as detailed in Table 14-1.

Table 14-1—defaultDS table

Name	Data type	Operations supported ^a	References
clockIdentity	ClockIdentity	R	14.2.2
numberPorts	UIInteger16	R	14.2.3
clockQualtiy.clockClass	UIInteger8	R	14.2.4.2; 7.6.2.5 of IEEE Std 1588-2019
clockQuality.clockAccuracy	Enumeration8	R	14.2.4.3; 7.6.2.6 of IEEE Std 1588-2019
clockQuality.offsetScaledLogVa riance	UIInteger16	R	14.2.4.4
priority1	UIInteger8	RW	14.2.5
priority2	UIInteger8	RW	14.2.6
gmCapable	Boolean	R	14.2.7
currentUtcOffset	Integer16	R	14.2.8
currentUtcOffsetValid	Boolean	R	14.2.9
leap59	Boolean	R	14.2.10
leap61	Boolean	R	14.2.11

Table 14-1—defaultDS table (continued)

Name	Data type	Operations supported ^a	References
timeTraceable	Boolean	R	14.2.12
frequencyTraceable	Boolean	R	14.2.13
ptpTimescale	Boolean	R	14.2.14
timeSource	TimeSource	R	14.2.15 and Table 8-2
domainNumber	UIInteger8	RW	14.2.16
sdold	UIInteger16	R	14.2.17
externalPortConfigurationEnabled	Boolean	RW	14.2.18
instanceEnable	Boolean	RW	14.2.19

^a R = Read only access; RW = Read/write access.

1 14.3 Current Parameter Data Set (currentDS)

2 14.3.1 General

3 The currentDS represents the position of a local system and other information, relative to the Grandmaster
 4 PTP Instance.

5 14.3.2 stepsRemoved

6 The value is the number of gPTP communication paths traversed between this PTP Instance and the
 7 Grandmaster PTP Instance, as specified in 10.3.3.

8 14.3.3 offsetFromTimeTransmitter

9 The value is an implementation-specific or profile standard-specific representation of the current value of
 10 the time difference between a timeReceiver and the Grandmaster Clock, as computed by the timeReceiver,
 11 and as specified in 10.2.10. The value is computed by an algorithm that is implementation specific or profile
 12 standard specific. The data type shall be TimeInterval. The default value is implementation specific.

13 NOTE—For example, the inputs to this implementation-specific algorithm could be the successive values of
 14 clockSourcePhaseOffset (see 10.2.4.7) of the ClockSyncOffset state machine (see 10.2.10 and Figure 10-6).

15 14.3.4 lastGmPhaseChange

16 The value (see 10.2.4.16) is the phase change that occurred on the most recent change in either Grandmaster
 17 PTP Instance or gmTimeBaseIndicator (see 9.2.2.3).

18 14.3.5 lastGmFreqChange

19 The value (see 10.2.4.17) is the frequency change that occurred on the most recent change in either
 20 Grandmaster PTP Instance or gmTimeBaseIndicator (see 9.2.2.3).

1 14.3.6 gmTimebaseIndicator

2 The value is the value of timeBaseIndicator of the current Grandmaster PTP Instance (see 9.2.2.3 and
 3 9.6.2.3).

4 14.3.7 gmChangeCount

5 This statistics counter tracks the number of times the Grandmaster PTP Instance has changed in a gPTP
 6 domain. This counter increments when the PortAnnounceInformation state machine enters the
 7 SUPERIOR_TIME_TRANSMITTER_PORT state or the
 8 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state (see 10.3.12 and Figure 10-14).

9 14.3.8 timeOfLastGmChangeEvent

10 This timestamp takes the value of sysUpTime (see IETF RFC 3418) when the most recent Grandmaster PTP
 11 Instance change occurred in a gPTP domain. This timestamp is updated when the PortAnnounceInformation
 12 state machine enters the SUPERIOR_TIME_TRANSMITTER_PORT state or the
 13 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state (see 10.3.12 and Figure 10-14).

14 14.3.9 timeOfLastGmPhaseChangeEvent

15 This timestamp takes the value of sysUpTime (see IETF RFC 3418) when the most recent change in
 16 Grandmaster Clock phase occurred due to a change of either the Grandmaster PTP Instance or the
 17 Grandmaster Clock time base. This timestamp is updated when one of the following occurs:

- 18 a) The PortAnnounceInformation state machine enters the
 19 SUPERIOR_TIME_TRANSMITTER_PORT state or the
 20 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state (see 10.3.12 and Figure 10-14), or
 21 b) The gmTimebaseIndicator managed object (see 14.3.6) changes and the lastGmPhaseChange field
 22 of the most recently received Follow_Up information TLV is nonzero.

23 14.3.10 timeOfLastGmFreqChangeEvent

24 This timestamp takes the value of sysUpTime (see IETF RFC 3418) when the most recent change in
 25 Grandmaster Clock frequency occurred due to a change of either the Grandmaster PTP Instance or the
 26 Grandmaster Clock time base. This timestamp is updated when one of the following occurs:

- 27 a) The PortAnnounceInformation state machine enters the
 28 SUPERIOR_TIME_TRANSMITTER_PORT state or the
 29 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state (see 10.3.12 and Figure 10-14), or
 30 b) The gmTimebaseIndicator managed object (see 14.3.6) changes, and the lastGmFreqChange field of
 31 the most recently received Follow_Up information TLV is nonzero.

32 14.3.11 currentDS table

33 There is one currentDS table per PTP Instance of a time-aware system, as detailed in Table 14-2.

34 14.4 Parent Parameter Data Set (parentDS)

35 14.4.1 General

36 The parentDS represents capabilities of the upstream system, toward the Grandmaster PTP Instance, as
 37 measured at a local system.

Table 14-2—currentDS table

Name	Data type	Operations supported ^a	References
stepsRemoved	UInteger16	R	14.3.2
offsetFromTimeTransmitter	TimeInterval	R	14.3.3
lastGmPhaseChange	ScaledNs	R	14.3.4
lastGmFreqChange	Float64	R	14.3.5
gmTimebaseIndicator	UInteger16	R	14.3.6
gmChangeCount	UInteger32	R	14.3.7
timeOfLastGmChangeEvent	UInteger32 (sysUp Time, IETF RFC 3418)	R	14.3.8
timeOfLastGmPhaseChangeEvent	UInteger32 (sysUp Time, IETF RFC 3418)	R	14.3.9
timeOfLastGmFreqChangeEvent	UInteger32 (sysUp Time, IETF RFC 3418)	R	14.3.10

^a R = Read only access; RW = Read/write access.

1 14.4.2 parentPortIdentity

2 If this PTP Instance is the Grandmaster PTP Instance, the value is a portIdentity whose clockIdentity is the
 3 clockIdentity of this PTP Instance, and whose portNumber is 0.

4 If this PTP Instance is not the Grandmaster PTP Instance, the value is the portIdentity of the
 5 TimeTransmitterPort (see Table 10-7) of the gPTP communication path attached to the single timeReceiver
 6 port of this PTP Instance.

7 The default value is a portIdentity for which the following apply:

- 8 a) The clockIdentity member is the value of the clockIdentity member of the default data set.
- 9 b) The portNumber member is 0.

10 14.4.3 cumulativeRateRatio

11 The value is an estimate of the ratio of the frequency of the Grandmaster Clock to the frequency of the
 12 LocalClock entity of this PTP Instance. cumulativeRateRatio is expressed as the fractional frequency offset
 13 multiplied by 2^{41} , i.e., the quantity $(\text{rateRatio} - 1.0)(2^{41})$, where rateRatio is computed by the
 14 PortSyncSyncReceive state machine (see 10.2.8.1.4).

15 14.4.4 grandmasterIdentity

16 The value is the clockIdentity attribute (see 8.5.2.2) of the Grandmaster PTP Instance.

17 The default value is the value of defaultDS.clockIdentity (14.2.2).

1 **14.4.5 grandmasterClockQuality**

2 **14.4.5.1 General**

3 This is a structure whose data type is ClockQuality (see 6.4.3.8).

4 **14.4.5.2 grandmasterClockQuality.clockClass**

5 The value is the clockClass (see 8.6.2.2) of the Grandmaster PTP Instance.

6 The default value is the clockClass member of the default data set.

7 **14.4.5.3 grandmasterClockQuality.clockAccuracy**

8 The value is the clockAccuracy (see 8.6.2.3) of the Grandmaster PTP Instance.

9 The default value is the clock accuracy member of the default data set.

10 **14.4.5.4 grandmasterClockQuality.offsetScaledLogVariance**

11 The value is the offsetScaledLogVariance (see 8.6.2.4) of the Grandmaster PTP Instance.

12 The default value is the offsetScaledLogVariance member of the default data set.

13 **14.4.6 grandmasterPriority1**

14 The value is the priority1 attribute (see 8.6.2.1) of the Grandmaster PTP Instance.

15 The default value is the priority1 value of the default data set.

16 **14.4.7 grandmasterPriority2**

17 The value is the priority2 attribute (see 8.6.2.5) of the Grandmaster PTP Instance.

18 The default value is the priority2 value of the default data set.

19 **14.4.7a gmPresent**

20 The value is the value of the per PTP Instance global variable gmPresent (see 10.2.4.13).

21 **14.4.8 parentDS table**

22 There is one parentDS table per PTP Instance of a time-aware system, as detailed in Table 14-3.

Table 14-3—parentDS table

Name	Data type	Operations supported ^a	References
parentPortIdentity	PortIdentity (see 6.4.3.7)	R	14.4.2
cumulativeRateRatio	Integer32	R	14.4.3
grandmasterIdentity	ClockIdentity	R	14.4.4
grandmasterClockQuality.clock Class	UIInteger8	R	14.4.5.2; 7.6.2.5 of IEEE Std 1588-2019
grandmasterClockQuality.clock Accuracy	Enumeration8	R	14.4.5.3; 7.6.2.6 of IEEE Std 1588-2019
grandmasterClockQuality.offset ScaledLog Variance	UIInteger16	R	14.4.5.4
grandmasterPriority1	UIInteger8	R	14.4.6
grandmasterPriority2	UIInteger8	R	14.4.7
gmPresent	Boolean	R	14.4.7a

^a R = Read only access; RW = Read/write access.

1 **14.5 Time Properties Parameter Data Set (timePropertiesDS)**

2 **14.5.1 General**

3 The timePropertiesDS represents capabilities of the Grandmaster PTP Instance, as measured at a local
4 system.

5 **14.5.2 currentUtcOffset**

6 The value is currentUtcOffset for the current Grandmaster PTP Instance (see 14.2.8). It is equal to the value
7 of the global variable currentUtcOffset (see 10.3.9.10). The value is in units of seconds.

8 **14.5.3 currentUtcOffsetValid**

9 The value is currentUtcOffsetValid for the current Grandmaster PTP Instance (see 14.2.9). It is equal to the
10 global variable currentUtcOffsetValid (see 10.3.9.6).

11 **14.5.4 leap59**

12 The value is leap59 for the current Grandmaster PTP Instance (see 14.2.10). It is equal to the global variable
13 leap59 (see 10.3.9.5).

14 **14.5.5 leap61**

15 The value is leap61 for the current Grandmaster PTP Instance (see 14.2.11). It is equal to the global variable
16 leap61 (see 10.3.9.4).

17 **14.5.6 timeTraceable**

18 The value is timeTraceable for the current Grandmaster PTP Instance (see 14.2.12). It is equal to the global
19 variable timeTraceable (see 10.3.9.8).

20 **14.5.7 frequencyTraceable**

21 The value is frequencyTraceable for the current Grandmaster PTP Instance (see 14.2.13). It is equal to the
22 global variable frequencyTraceable (see 10.3.9.9).

23 **14.5.8 ptptimescale**

24 The value is ptptimescale for the current Grandmaster PTP Instance (see 14.2.14).

25 **14.5.9 timeSource**

26 The value is timeSource for the current Grandmaster PTP Instance (see 14.2.15). It is equal to the global
27 variable timeSource (see 10.3.9.11).

28 **14.5.10 timePropertiesDS table**

29 There is one timePropertiesDS table per PTP Instance of a time-aware system, as detailed in Table 14-4.

Table 14-4—timePropertiesDS table

Name	Data type	Operations supported ^a	References
currentUtcOffset	Integer16	R	14.5.2
currentUtcOffsetValid	Boolean	R	14.5.3
leap59	Boolean	R	14.5.4
leap61	Boolean	R	14.5.5
timeTraceable	Boolean	R	14.5.6
frequencyTraceable	Boolean	R	14.5.7
ptpTimescale	Boolean	R	14.5.8
timeSource	TimeSource	R	14.5.9 and Table 8-2

^a R = Read only access; RW = Read/write access.

1 14.6 Path Trace Parameter Data Set (pathTraceDS)

2 14.6.1 General

3 The pathTraceDS represents the current path trace information available at the PTP Instance.

4 14.6.2 list

5 The value is the array of ClockIdentity values contained in the pathTrace array (see 10.3.9.23), which
 6 represents the current path trace information and which is carried in the path trace TLV (see 10.6.3.3).

7 The initialization value shall be the empty list (i.e., an array of length 0).

8 14.6.3 enable

9 The value is TRUE.

10 NOTE—This member is included for compatibility with IEEE Std 1588-2019. In IEEE Std 1588-2019, the path trace
 11 mechanism is optional, and the pathTraceDS.enable member is configurable (its value in IEEE Std 1588-2019 is TRUE
 12 or FALSE, depending on whether the path trace mechanism is operational or not operational, respectively. However, the
 13 pathTrace mechanism is mandatory in this standard, and the value of enable is always TRUE.

14 14.6.4 pathTraceDS table

15 There is one pathTraceDS table per PTP Instance, as detailed in Table 14-5.

16

17

18

19

Table 14-5—pathTraceDS table

Name	Data type	Operations supported ^a	References
list	ClockIdentity[N], where N is defined in 10.3.9.23	R	14.6.2
enable	Boolean	R	14.6.3

^a R = Read only access; RW = Read/write access.

1 14.7 Acceptable TimeTransmitter Table Parameter Data Set

2 (acceptableTimeTransmitterTableDS)

3 14.7.1 General

4 The acceptableTimeTransmitterTableDS represents the acceptable timeTransmitter table used when an EPON port is used by a PTP Instance of a time-aware system.

6 14.7.2 maxTableSize

7 The value is the maximum size of the AcceptableTimeTransmitterTable. It is equal to the maxTableSize member of the AcceptableTimeTransmitterTable structure (see 13.1.3.2).

9 14.7.3 actualTableSize

10 The value is the actual size of the AcceptableTimeTransmitterTable. It is equal to the actualTableSize member of the AcceptableTimeTransmitterTable structure (see 13.1.3.2 and 13.1.3.5), i.e., the current number of elements in the acceptable timeTransmitter array. The actual table size is less than or equal to the maxTableSize.

14 14.7.4 acceptableTimeTransmitterArray

15 Each element of this array is an AcceptableTimeTransmitter structure (see 13.1.3.3 and 13.1.3.5).

16 14.7.5 acceptableTimeTransmitterTableDS table

17 There is one acceptableTimeTransmitterTableDS table per PTP Instance of a time-aware system, as detailed in Table 14-6.

Table 14-6—acceptableTimeTransmitterTableDS table

Name	Data type	Operations supported ^a	References
maxTableSize	UInteger16	R	14.7.2
actualTableSize	UInteger16	RW	14.7.3
acceptableTimeTransmitterArray	AcceptableTimeTransmitter[actualTableSize] (see 13.1.3.3)	RW	14.7.4

^a R = Read only access; RW = Read/write access.

1 14.7a PTP Instance Synchronization Parameter Data Set (ptpInstanceSyncDS)

2 14.7a.1 General

3 The ptpInstanceSyncDS describes the synchronization status of the PTP Instance. The ptpInstanceSyncDS
 4 shall be implemented if the optional hot standby feature (see Clause 18) is implemented and may be
 5 implemented otherwise.

6 14.7a.2 isSynced

7 The value of the global variable isSynced (see 18.4.1.1).

8 14.7a.3 offsetFromTimeTransmitterMax

9 The value is the threshold for offsetFromTimeTransmitter (see 18.4.1.2), below which the PTP Instance is
 10 considered to be synchronized. For values less than or equal to zero, the PTP Instance is considered
 11 synchronized if and only if offsetFromTimeTransmitter is zero. For values greater than zero, the PTP
 12 Instance is considered synchronized if and only if the equation:

$$13 \quad -\text{offsetFromTimeTransmitterThreshold} \leq \text{offsetFromTimeTransmitter} \leq \\ \boxed{\text{offsetFromTimeTransmitterThreshold}}$$

14 is satisfied.

15 14.7a.4 rxSyncCountTimeReceiverPThresh

16 The value of rxSyncCountTimeReceiverPThresh is the threshold for rxSyncCountTimeReceiverP
 17 (see 18.4.1.4), above which the PTP Instance is considered to be synchronized.

18 14.7a.5 offsetMaxExceededCountThresh

19 The value of offsetMaxExceededCountThresh (see 18.4.1.7) is the threshold for the number of consecutive
 20 exceedances of offsetFromTimeTransmitterMax (see 18.4.1.3) by offsetFromTimeTransmitter
 21 (see 18.4.1.2), at which isSynced (see 18.4.1.1) is no longer TRUE.

22 14.7a.6 offsetMaxMetCountThresh

23 The value of offsetMaxMetCountThresh (18.4.1.9) is the threshold for the number of consecutive
 24 occurrences of offsetFromTimeTransmitter (see 18.4.1.2) being within offsetFromTimeTransmitterMax
 25 (see 18.4.1.3), at which isSynced (see 18.4.1.1) is changed to TRUE if it currently is FALSE.

26 14.7a.7 ptpInstanceSyncDS table

27 There is one ptpInstanceSyncDS table per PTP Instance, as detailed in Table 14-6a.

28 14.7b Drift Tracking Parameter Data Set (driftTrackingDS)

29 14.7b.1 General

30 The driftTrackingDS contains a managed object that is used to enable or disable the optional Drift_Tracking
 31 TLV.

Table 14-6a—ptpInstanceSyncDS table

Name	Data type	Operations supported ^a	References
isSynced	Boolean	R	14.7a.2
offsetFromTimeTransmitterMax	TimeInterval	RW	14.7a.3
rxSyncCountTimeReceiverPThresh	UInteger32	RW	14.7a.4
offsetMaxExceededCountThresh	UInteger32	RW	14.7a.5
offsetMaxMetCountThresh	UInteger32	RW	14.7a.6

^a R = Read only access; RW = Read/write access.

1 14.7b.2 driftTrackingTlvSupport

2 The value of driftTrackingTlvSupport indicates whether the Drift_Tracking TLV is enabled or disabled. If
 3 the value is TRUE, the TLV is enabled, i.e., the global variable driftTrackingTlvSupport (see 10.2.4.28) is
 4 set to TRUE. If the value is FALSE, the TLV is disabled, i.e., the global variable driftTrackingTlvSupport is
 5 set to FALSE.

6 14.7b.3 driftTrackingDS table

7 There is one driftTrackingDS table per PTP Instance, as detailed in Table 14-6b.

Table 14-6b—driftTrackingDS table

Name	Data type	Operations supported ^a	References
driftTrackingTlvSupport	Boolean	RW	14.7b.2

^a R = Read only access; RW = Read/write access.

8 14.8 Port Parameter Data Set (portDS)

9 14.8.1 General

10 The portDS represents PTP Port time-aware capabilities for a PTP Instance of a time-aware system.

11 For the single PTP Port of a PTP End Instance and for each PTP Port of a PTP Relay Instance, the portDS is
 12 maintained as the basis for making protocol decisions and providing values for message fields. The number
 13 of such data sets is the same as the value of defaultDS.numberPorts.

14 14.8.2 portIdentity

15 The value is the portIdentity attribute of the local PTP Port (see 8.5.2).

1 14.8.3 portState

2 The value is the value of the PTP Port state of this PTP Port (see Table 10-2) and is taken from the
 3 enumeration in Table 14-7. It is equal to the value of the global variable selectedState (see 10.2.4.20) for this
 4 PTP Port.

Table 14-7—portState enumeration

State	Value
DisabledPort	3
TimeTransmitterPort	6
PassivePort	7
TimeReceiverPort	9
	All other values reserved

NOTE—The enumeration values are consistent with Table 20 in IEEE Std 1588-2019.

5 The default value is 3 (DisabledPort).

6 14.8.4 ptpNetEnabled

7 The value is equal to the value of the Boolean ptpNetEnabled (see 10.2.5.13). Setting the managed object
 8 ptpNetEnabled causes the Boolean ptpNetEnabled to have the same value.

9 14.8.5 delayMechanism

10 The value of **delayMechanism** indicates the mechanism for measuring mean propagation delay and neighbor
 11 rate ratio on the link attached to this PTP Port and is taken from the enumeration in Table 14-8.
 12 **portDS.delayMechanism** may be P2P (see 11.2.17) only if the only domainNumber active on the physical
 13 port is 0.

Table 14-8—delayMechanism enumeration

Delay mechanism	Value	Specification
P2P	02	The PTP Port uses the transport-specific peer-to-peer delay mechanism
COMMON_P2P	03	The PTP Port uses the CMLDS
SPECIAL	04	The PTP Port uses a transport that has a native time transfer mechanism and, therefore, does not use the peer-to-peer delay mechanism (e.g., IEEE 802.11, IEEE 802.3 EPON)
All other values reserved		

NOTE—The enumeration values are consistent with Table 21 in IEEE Std 1588-2019.

1 14.8.6 isMeasuringDelay

2 The value is equal to the value of the per-port, per PTP Instance global variable isMeasuringDelay (see
 3 11.2.13.6 and 16.4.3.3).

4 14.8.7 asCapable

5 The value is equal to the value of the Boolean asCapable (see 10.2.5.1).

6 14.8.8 meanLinkDelay

7 The value is equal to the value of the per-PTP Port global variable meanLinkDelay (see 10.2.5.8). It is an
 8 estimate of the current one-way propagation time on the link attached to this PTP Port, measured as
 9 specified for the respective medium (see 11.2.17, 12.5.2, and 16.4). The value is zero for PTP Ports attached
 10 to IEEE 802.3 EPON links and for the timeTransmitter port of an IEEE 802.11 link, because one-way
 11 propagation delay is not measured on the latter and not directly measured on the former. The data type shall
 12 be TimeInterval. The default value is zero.

13 NOTE—The underlying per-port global variable meanLinkDelay is of type UScaledNS, which is a 96-bit value
 14 (see 6.4.3.2). meanLinkDelay values that are larger than the maximum value that can be represented by the TimeInterval
 15 data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this managed object are set
 16 to this largest value.

17 14.8.9 meanLinkDelayThresh

18 The value is equal to the value of the per-PTP Port global variable meanLinkDelayThresh (see 11.2.13.7). It
 19 is the propagation time threshold above which a PTP Port is considered not capable of participating in the
 20 IEEE 802.1AS protocol. Setting the managed object meanLinkDelayThresh causes the per PTP Port global
 21 variable meanLinkDelayThresh to have the same value.

22 NOTE—The underlying per-port global variable meanLinkDelayThresh is of type UScaledNS, which is a 96-bit value
 23 (see 6.4.3.2). meanLinkDelayThresh values that are larger than the maximum value that can be represented by the
 24 TimeInterval data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this managed
 25 object are set to this largest value.

26 14.8.10 delayAsymmetry

27 The value is the asymmetry in the propagation delay on the link attached to this PTP Port relative to the
 28 Grandmaster Clock time base, as defined in 10.2.5.9 and 8.3. If propagation delay asymmetry is not
 29 modeled, then delayAsymmetry is 0.

30 NOTE—The underlying per-port global variable delayAsymmetry is of type ScaledNS, which is a 96-bit value
 31 (see 6.4.3.1). delayAsymmetry values that are larger than the maximum value that can be represented by the
 32 TimeInterval data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this
 33 managed object are set to this largest value. delayAsymmetry values that are less than the minimum value that can be
 34 represented by the TimeInterval data type, i.e., 0x8000 0000 0000 0001 written in twos complement form (where the
 35 units are 2^{-16} ns; see 6.4.3.3), used for this managed object are set to this smallest value.

36 14.8.11 neighborRateRatio

37 The value is an estimate of the ratio of the frequency of the LocalClock entity of the PTP Instance at the
 38 other end of the link attached to this PTP Port, to the frequency of the LocalClock entity of this PTP Instance
 39 (see 10.2.5.7). neighborRateRatio is expressed as the fractional frequency offset multiplied by 2^{41} , i.e., the
 40 quantity ($\text{neighborRateRatio} - 1.0)(2^{41})$.

1 14.8.12 initialLogAnnounceInterval

2 If useMgtSettableLogAnnounceInterval is FALSE, the value is the logarithm to base 2 of the announce
 3 interval used when:

- 4 a) The PTP Port is initialized or
- 5 b) A message interval request TLV is received with the logAnnounceInterval field set to 126
 6 (see 10.7.2.2 and the AnnounceIntervalSetting state machine in 10.3.17).

7 14.8.13 currentLogAnnounceInterval

8 The value is the logarithm to the base 2 of the current announce interval (see 10.7.2.2).

9 14.8.14 useMgtSettableLogAnnounceInterval

10 The managed object is a Boolean that determines the source of the announce interval. If the value is TRUE,
 11 the value of currentLogAnnounceInterval is set equal to the value of mgtSettableLogAnnounceInterval (see
 12 14.8.15). If the value is FALSE, the value of currentLogAnnounceInterval is determined by the
 13 AnnounceIntervalSetting state machine (see 10.3.17). The default value of
 14 useMgtSettableLogAnnounceInterval is FALSE for domain 0 and TRUE for domains other than domain 0.

15 14.8.15 mgtSettableLogAnnounceInterval

16 The value is the logarithm to base 2 of the announce interval used if useMgtSettableLogAnnounceInterval is
 17 TRUE. The value is not used if useMgtSettableLogAnnounceInterval is FALSE.

18 14.8.16 announceReceiptTimeout

19 The value is the number of Announce message transmission intervals that a timeReceiver port waits without
 20 receiving an Announce message before assuming that the timeTransmitter is no longer transmitting
 21 Announce messages and the BTCA needs to be run, if appropriate (see 10.7.3.2).

22 14.8.17 initialLogSyncInterval

23 If useMgtSettableLogSyncInterval is FALSE, the value is the logarithm to base 2 of the sync interval used
 24 when

- 25 a) The PTP Port is initialized or
- 26 b) A message interval request TLV is received with the logTimeSyncInterval field set to 126
 27 (see 10.7.2.3, 11.5.2.3, 12.8.2, 13.9.2, and the SyncIntervalSetting state machine in 10.3.18).

28 14.8.18 currentLogSyncInterval

29 The value is the logarithm to the base 2 of the current time-synchronization transmission interval
 30 (see 10.7.2.3).

31 14.8.19 useMgtSettableLogSyncInterval

32 The managed object is a Boolean that determines the source of the sync interval. If the value is TRUE, the
 33 value of currentLogSyncInterval is set equal to the value of mgtSettableLogSyncInterval (see 14.8.20). If
 34 the value of the managed object is FALSE, the value of currentLogSyncInterval is determined by the
 35 SyncIntervalSetting state machine (see 10.3.18). The default value of useMgtSettableLogSyncInterval is
 36 FALSE for domain 0 and TRUE for domains other than domain 0.

1 14.8.20 mgtSettableLogSyncInterval

2 The value is the logarithm to base 2 of the sync interval if useMgtSettableLogSyncInterval is TRUE. The
 3 value is not used if useMgtSettableLogSyncInterval is FALSE.

4 14.8.21 syncReceiptTimeout

5 The value is the number of time-synchronization transmission intervals that a timeReceiver port waits
 6 without receiving synchronization information before assuming that the timeTransmitter is no longer
 7 transmitting synchronization information and that the BTCA needs to be run, if appropriate (see 10.7.3.1).

8 14.8.22 syncReceiptTimeoutTimeInterval

9 The value is equal to the value of the per-PTP Port global variable syncReceiptTimeoutTimeInterval
 10 (see 10.2.5.3). It is the time interval after which sync receipt timeout occurs if time-synchronization
 11 information has not been received during the interval.

12 14.8.23 initialLogPdelayReqInterval

13 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 14 path delay (see 16.4.3.2), the value is the logarithm to base 2 of the Pdelay_Req message transmission
 15 interval used when:

- 16 a) The PTP Port is initialized or
- 17 b) A message interval request TLV is received with the logLinkDelayInterval field set to 126 (see
 18 11.5.2.2 and the LinkDelayIntervalSetting state machine in 11.2.21).

19 For all other media, the value is 127.

20 14.8.24 currentLogPdelayReqInterval

21 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 22 path delay (see 16.4.3.2), the value is the logarithm to the base 2 of the current Pdelay_Req message
 23 transmission interval (see 11.5.2.2).

24 For all other media, the value is 127.

25 14.8.25 useMgtSettableLogPdelayReqInterval

26 The managed object is a Boolean that determines the source of the mean time interval between successive
 27 Pdelay_Req messages. If the value is TRUE, the value of currentLogPdelayReqInterval is set equal to the
 28 value of mgtSettableLogPdelayReqInterval (see 14.8.26). If the value of the managed object is FALSE,
 29 the value of currentLogPdelayReqInterval is determined by the LinkDelayIntervalSetting state machine
 30 (see 11.2.21). The default value of useMgtSettableLogPdelayReqInterval is FALSE.

31 14.8.26 mgtSettableLogPdelayReqInterval

32 The value is the logarithm to base 2 of the mean time interval between successive Pdelay_Req
 33 messages if useMgtSettableLogPdelayReqInterval is TRUE. The value is not used if
 34 useMgtSettableLogPdelayReqInterval is FALSE.

35 14.8.27 initialLogGtpCapableMessageInterval

36 The value is the logarithm to base 2 of the gPTP capable message interval used when:

- 1 a) The PTP Port is initialized or
- 2 b) A gPtpCapableMessage interval request TLV is received with the logGptpCapableMessageInterval field set to 126 (see 10.6.4.5 and the GptpCapableIntervalSetting state machine in 10.4.3).

4 14.8.28 currentLogGptpCapableMessageInterval

5 The value is the logarithm to the base 2 of the current gPTP capable message interval (see 10.7.2.5).

6 14.8.29 useMgtSettableLogGptpCapableMessageInterval

7 The managed object is a Boolean that determines the source of the gPTP capable message interval. If the
 8 value is TRUE, the value of currentLogGptpCapableMessageInterval is set equal to the value of
 9 mgtSettableLogGptpCapableMessageInterval (see 14.8.30). If the value of the managed object is FALSE,
 10 the value of currentLogGptpCapableMessageInterval is determined by the
 11 GptpCapableMessageIntervalSetting state machine (see 10.4.3). The default value of
 12 useMgtSettableLogGptpCapableMessageInterval is FALSE.

13 14.8.30 mgtSettableLogGptpCapableMessageInterval

14 The value is the logarithm to base 2 of the gPtpCapableMessageInterval
 15 if useMgtSettableLogGptpCapableMessageInterval is TRUE. The value is not used if
 16 useMgtSettableLogGptpCapableMessageInterval is FALSE.

17 14.8.31 initialComputeNeighborRateRatio

18 If useMgtSettableComputeNeighborRateRatio is FALSE, then for full-duplex IEEE 802.3 media and for
 19 CSN media that use the peer-to-peer delay mechanism to measure path delay (see 16.4.3.2), the value is the
 20 initial value of computeNeighborRateRatio (see 10.2.5.10).

21 For all other media, the value is TRUE.

22 14.8.32 currentComputeNeighborRateRatio

23 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 24 path delay (see 16.4.3.2), the value is the current value of computeNeighborRateRatio.

25 For all other media, the value is TRUE.

26 14.8.33 useMgtSettableComputeNeighborRateRatio

27 The managed object is a Boolean that determines the source of the value of computeNeighborRateRatio. If
 28 the value is TRUE, the value of computeNeighborRateRatio is set equal to the value of
 29 mgtSettableComputeNeighborRateRatio (see 14.16.17). If the value of the managed object is FALSE, the
 30 value of currentComputeNeighborRateRatio is determined by the LinkDelayIntervalSetting state machine
 31 (see 11.2.21). The default value of useMgtSettableComputeNeighborRateRatio is FALSE.

32 14.8.34 mgtSettableComputeNeighborRateRatio

33 computeNeighborRateRatio is configured to this value if useMgtSettableComputeNeighborRateRatio is
 34 TRUE. The value is not used if useMgtSettableComputeNeighborRateRatio is FALSE.

1 14.8.35 initialComputeMeanLinkDelay

2 If useMgtSettableComputeMeanLinkDelay is FALSE, then for full-duplex IEEE 802.3 media and for CSN
 3 media that use the peer-to-peer delay mechanism to measure path delay (see 16.4.3.2), the value is the initial
 4 value of computeMeanLinkDelay (see 10.2.5.10).

5 For all other media, the value is TRUE.

6 14.8.36 currentComputeMeanLinkDelay

7 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 8 path delay (see 16.4.3.2), the value is the current value of computeMeanLinkDelay.

9 For all other media, the value is TRUE.

10 14.8.37 useMgtSettableComputeMeanLinkDelay

11 The managed object is a Boolean that determines the source of the value of computeMeanLinkDelay. If the
 12 value is TRUE, the value of computeMeanLinkDelay is set equal to the value of
 13 mgtSettableComputeMeanLinkDelay (see 14.8.38). If the value of the managed object is FALSE, the value
 14 of currentComputeMeanLinkDelay is determined by the LinkDelayIntervalSetting state machine
 15 (see 11.2.21). The default value of useMgtSettableComputeMeanLinkDelay is FALSE.

16 14.8.38 mgtSettableComputeMeanLinkDelay

17 computeMeanLinkDelay is configured to this value if useMgtSettableComputeMeanLinkDelay is TRUE.
 18 The value is not used if useMgtSettableComputeMeanLinkDelay is FALSE.

19 14.8.39 allowedLostResponses

20 The value is equal to the value of the per-PTP Port global variable allowedLostResponses (see 11.5.3 and
 21 11.2.13.4). It is the number of Pdelay_Req messages without valid responses above which a PTP Port is
 22 considered to be not exchanging peer delay messages with its neighbor. Setting the managed object
 23 allowedLostResponses causes the per PTP Port global variable allowedLostResponses to have the same
 24 value.

25 14.8.40 allowedFaults

26 The value is equal to the value of the per-PTP Port global variable allowedFaults (see 11.5.4 and 11.2.13.5).
 27 It is the number of faults (see 11.5.4) above which asCapable is set to FALSE, i.e., a PTP Port is considered
 28 not capable of interoperating with its neighbor via the IEEE 802.1AS protocol (see 10.2.5.1). Setting the
 29 managed object allowedLostResponses causes the per PTP Port global variable allowedFaults to have the
 30 same value.

31 14.8.41 gPtpCapableReceiptTimeout

32 The value is the number of transmission intervals that a PTP Port waits without receiving the gPTP capable
 33 TLV before assuming that the neighbor PTP Port is no longer invoking gPTP (see 10.7.3.3).

34 14.8.42 versionNumber

35 This value is set to versionPTP as specified in 10.6.2.2.4.

1 **14.8.43 nup**

2 For an OLT port of an IEEE 802.3 EPON link, the value is the effective index of refraction for the EPON
 3 upstream wavelength light of the optical path (see 13.1.4 and 13.8.1.2.2). The default value is 1.46770 for
 4 1 Gb/s upstream links and 1.46773 for 10 Gb/s upstream links.

5 For all other PTP Ports, the value is 0.

6 **14.8.44 ndown**

7 For an OLT port of an IEEE 802.3 EPON link, the value is the effective index of refraction for the EPON
 8 downstream wavelength light of the optical path (see 13.1.4 and 13.8.1.2.1). The default value is 1.46805 for
 9 1 Gb/s downstream links and 1.46851 for 10 Gb/s downstream links.

10 For all other PTP Ports, the value is 0.

11 **14.8.45 oneStepTxOper**

12 The value is equal to the value of the per-PTP Port global variable oneStepTxOper (see 11.2.13.11). Its value
 13 is TRUE if the PTP Port is sending one-step Sync messages and FALSE if the PTP Port is sending two-step
 14 Sync and Follow-Up messages.

15 **14.8.46 oneStepReceive**

16 The value is equal to the value of the per-PTP Port global variable oneStepReceive (see 11.2.13.9). Its value
 17 is TRUE if the PTP Port is capable of receiving and processing one-step Sync messages.

18 **14.8.47 oneStepTransmit**

19 The value is equal to the value of the per-PTP Port global variable oneStepTransmit (see 11.2.13.10). Its value
 20 is TRUE if the PTP Port is capable of transmitting one-step Sync messages.

21 **14.8.48 initialOneStepTxOper**

22 If useMgtSettableOneStepTxOper is FALSE, the value is used to initialize currentOneStepTxOper when the
 23 PTP Port is initialized. If useMgtSettableOneStepTxOper is TRUE, the value of initialOneStepTxOper is not
 24 used. The default value of initialOneStepTxOper shall be FALSE.

25 **14.8.49 currentOneStepTxOper**

26 The value is TRUE if it is desired, either via management or via a received Signaling message, that the PTP
 27 Port transmit one-step Sync messages. The value is FALSE if it is not desired, either via management or via
 28 a received Signaling message, that the PTP Port transmit one-step Sync messages.

29 NOTE—The PTP Port will send one-step Sync messages only if currentOneStepTxOper and oneStepTransmit
 30 (see 14.8.47) are both TRUE (see 11.2.16 and Figure 11-8).

31 **14.8.50 useMgtSettableOneStepTxOper**

32 The managed object is a Boolean that determines the source of currentOneStepTxOper. If the value is
 33 TRUE, the value of currentOneStepTxOper is set equal to the value of mgtSettableOneStepTxOper
 34 (see 14.8.51). If the value is FALSE, the value of currentOneStepTxOper is determined by the
 35 OneStepTxOperSetting state machine (see 11.2.16 and Figure 11-8). The default value of
 36 useMgtSettableOneStepTxOper is TRUE.

1 14.8.51 mgtSettableOneStepTxOper

2 If useMgtSettableOneStepTxOper is TRUE, currentOneStepTxOper is set equal to the value of
 3 mgtSettableOneStepTxOper. The value of mgtSettableOneStepTxOper is not used if
 4 useMgtSettableOneStepTxOper is FALSE. The default value of mgtSettableOneStepTxOper is FALSE for
 5 domains other than domain 0.

6 14.8.52 syncLocked

7 The value is equal to the value of the per-PTP Port global variable syncLocked (see 10.2.5.15). Its value is
 8 TRUE if the PTP Port will transmit a Sync as soon as possible after the timeReceiver PTP Port receives a
 9 Sync.

10 14.8.53 pdelayTruncatedTimestampsArray

11 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 12 path delay (see 16.4.3.2), the values of the four elements of this array are as described in Table 14-9. For all
 13 other media, the values are zero. Array elements 0, 1, 2, and 3 correspond to the timestamps t1, t2, t3, and t4,
 14 respectively, in Figure 11-1 and are expressed in units of 2^{-16} ns (i.e., the value of each array element is
 15 equal to the remainder obtained upon dividing the respective timestamp, expressed in units of 2^{-16} ns,
 16 by 2^{48}). At any given time, the timestamp values stored in the array are for the same, and most recently
 17 completed, peer delay message exchange.

Table 14-9—Description of pdelayTruncatedTimestampsArray

Array element	Timestamp description	Corresponding timestamp of Figure 11-1	Units
0	pdelayReqEventEgressTimestamp for Pdelay_Req message, of most recently completed peer delay message exchange, transmitted by this PTP Instance (NOTE 1)	t1	2^{-16} ns
1	pdelayReqEventIngressTimestamp for Pdelay_Req message received at peer delay responder to which this Link Port sends Pdelay_Req, of most recently completed peer delay message exchange; it is equal to the sum of the following: a) The ns field of the requestReceiptTimestamp (see Table 11-13), multiplied by 2^{16} b) The correctionField (see Table 11-6) (NOTE 2)	t2	2^{-16} ns
2	pdelayRespEventEgressTimestamp for Pdelay_Resp message, of most recently completed peer delay message exchange, transmitted by peer delay responder to which this Link Port sends Pdelay_Req; it is equal to the sum of the following: a) The ns field of the responseOriginTimestamp (see Table 11-14), multiplied by 2^{16} b) The correctionField (see Table 11-6) (NOTE 3)	t3	2^{-16} ns

Table 14-9—Description of pdelayTruncatedTimestampsArray (continued)

Array element	Timestamp description	Corresponding timestamp of Figure 11-1	Units
3	pdelayRespEventIngressTimestamp for Pdelay_Resp message, of most recently completed peer delay message exchange, received by this PTP Instance <small>(NOTE 4)</small>	t4	2^{-16} ns

NOTE 1—This quantity is not simply the nanoseconds plus fractional nanoseconds portion of the pdelayReqEventEgressTimestamp. Rather, it is equal to
 $[pdelayReqEventEgressTimestamp.seconds \times (10^9)(2^{16}) +$
 $pdelayReqEventEgressTimestamp.fractionalNanoseconds] \bmod 2^{48}$, where
 pdelayReqEventEgressTimestamp is expressed as an ExtendedTimestamp (see 6.4.3.5). Its units are 2^{-16} ns.

NOTE 2—This quantity is not simply the nanoseconds plus fractional nanoseconds portion of the pdelayReqEventIngressTimestamp. Rather, it is equal to
 $[pdelayReqEventIngressTimestamp.seconds \times (10^9)(2^{16}) +$
 $pdelayReqEventIngressTimestamp.fractionalNanoseconds] \bmod 2^{48}$, where
 pdelayReqEventIngressTimestamp is expressed as an ExtendedTimestamp (see 6.4.3.5). Its units are 2^{-16} ns.

NOTE 3—This quantity is not simply the nanoseconds plus fractional nanoseconds portion of the pdelayRespEventEgressTimestamp. Rather, it is equal to
 $[pdelayRespEventEgressTimestamp.seconds \times (10^9)(2^{16}) +$
 $pdelayRespEventEgressTimestamp.fractionalNanoseconds] \bmod 2^{48}$, where
 pdelayRespEventEgressTimestamp is expressed as an ExtendedTimestamp (see 6.4.3.5). Its units are 2^{-16} ns.

NOTE 4—This quantity is not simply the nanoseconds plus fractional nanoseconds portion of the pdelayRespEventIngressTimestamp. Rather, it is equal to
 $[pdelayRespEventIngressTimestamp.seconds \times (10^9)(2^{16}) +$
 $pdelayRespEventIngressTimestamp.fractionalNanoseconds] \bmod 2^{48}$, where
 pdelayRespEventIngressTimestamp is expressed as an ExtendedTimestamp (see 6.4.3.5). Its units are 2^{-16} ns.

1 14.8.54 minorVersionNumber

2 This value is set to minorVersionPTP as specified in 10.6.2.2.3.

3 14.8.54a gptpCapableStateMachinesEnabled

4 A Boolean that is used to enable or disable the GptpCapableTransmit, GptpCapableReceive, and
 5 GptpCapableIntervalSetting state machines. If the value is TRUE, the GptpCapableTransmit and
 6 GptpCapableReceive state machines are enabled, and the GptpCapableIntervalSetting state machine is
 7 enabled if it is implemented. If the value is FALSE, the GptpCapableTransmit and GptpCapableReceive
 8 state machines are disabled, and the GptpCapableIntervalSetting state machine is disabled if it is
 9 implemented. The default value is TRUE.

10 14.8.54b nrrPdelay

11 The value of the global variable nrrPdelay (see 11.2.13.13).

12 The value is an estimate of the ratio of the frequency of the LocalClock entity of the time-aware system at
 13 the other end of the link attached to this Link Port, to the frequency of the LocalClock entity of this time-
 14 aware system (see 10.2.5.7). nrrPdelay is expressed as the fractional frequency offset stored in the global
 15 variable nrrPdelay (see 11.2.13.13) multiplied by 2^{41} , i.e., the quantity $(nrrPdelay - 1.0)(2^{41})$. The default
 16 value of nrrPdelay is 1.0.

1 14.8.54c nrSync

2 The value of the global variable nrSync (see 11.2.13.14).

3 The value is an estimate of the ratio of the frequency of the LocalClock entity of the time-aware system at
4 the other end of the link attached to this Link Port, to the frequency of the LocalClock entity of this time-
5 aware system (see 10.2.5.7). nrSync is expressed as the fractional frequency offset stored in the global
6 variable nrSync (see 11.2.13.14) multiplied by 2^{41} , i.e., the quantity $(\text{nrSync} - 1.0)(2^{41})$.

7 14.8.54d nrCompMethod

8 The value of the global variable nrCompMethod (see 11.2.13.15).

9 14.8.54e asCapableAcrossDomains

10 The value is equal to the value of the Boolean asCapableAcrossDomains (see 11.2.2 and 11.2.13.12).

11 14.8.55 portDS table

12 There is one portDS table per PTP Port, per PTP Instance of a time-aware system. Each portDS table
13 contains a set of parameters for each PTP Port that supports the time-synchronization capability, as detailed
14 in Table 14-10. Each table can be created or removed dynamically in implementations that support dynamic
15 configuration of PTP Ports and components.

Table 14-10—portDS table

Name	Data type	Operations supported ^a	References
portIdentity	PortIdentity (see 6.4.3.7)	R	14.8.2
portState	Enumeration8	R	14.8.3, Table 14-7
ptpPortEnabled	Boolean	RW	14.8.4
delayMechanism	Enumeration8	RW	14.8.5
isMeasuringDelay	Boolean	R	14.8.6
asCapable	Boolean	R	14.8.7
meanLinkDelay	TimeInterval	R	14.8.8
meanLinkDelayThresh	TimeInterval	RW	14.8.9
delayAsymmetry	TimeInterval	RW	14.8.10
neighborRateRatio	Integer32	R	14.8.11
initialLogAnnounceInterval	Integer8	RW	14.8.12
currentLogAnnounceInterval	Integer8	R	14.8.13
useMgtSettableLogAnnounceInterval	Boolean	RW	14.8.14
mgtSettableLogAnnounceInterval	Integer8	RW	14.8.15
announceReceiptTimeout	UInteger8	RW	14.8.16
initialLogSyncInterval	Integer8	RW	14.8.17

Table 14-10—portDS table (continued)

Name	Data type	Operations supported ^a	References
currentLogSyncInterval	Integer8	R	14.8.18
useMgtSettableLogSyncInterval	Boolean	RW	14.8.19
mgtSettableLogSyncInterval	Integer8	RW	14.8.20
syncReceiptTimeout	UInteger8	RW	14.8.21
syncReceiptTimeoutTimeInterval	UScaledNs	R	14.8.22
initialLogPdelayReqInterval	Integer8	RW	14.8.23
currentLogPdelayReqInterval	Integer8	R	14.8.24
useMgtSettableLogPdelayReqInterval	Boolean	RW	14.8.25
mgtSettableLogPdelayReqInterval	Integer8	RW	14.8.26
initialLogGtpCapableMessageInterval	Integer8	RW	14.8.27
currentLogGtpCapableMessageInterval	Integer8	R	14.8.28
useMgtSettableLogGtpCapableMessageInterval	Boolean	RW	14.8.29
mgtSettableLogGtpCapableMessageInterval	Integer8	RW	14.8.30
initialComputeNeighborRateRatio	Boolean	RW	14.8.31
currentComputeNeighborRateRatio	Boolean	R	14.8.32
useMgtSettableComputeNeighborRateRatio	Boolean	RW	14.8.33
mgtSettableComputeNeighborRateRatio	Boolean	RW	14.8.34
initialComputeMeanLinkDelay	Boolean	RW	14.8.35
currentComputeMeanLinkDelay	Boolean	R	14.8.36
useMgtSettableComputeMeanLinkDelay	Boolean	RW	14.8.37
mgtSettableComputeMeanLinkDelay	Boolean	RW	14.8.38
allowedLostResponses	UInteger8	RW	14.8.39
allowedFaults	UInteger8	RW	14.8.40
gPtpCapableReceiptTimeout	UInteger8	RW	14.8.41
versionNumber	UInteger4	R	14.8.42
nup (NOTE)	Float64	RW	14.8.43
ndown (NOTE)	Float64	RW	14.8.44
oneStepTxOper	Boolean	R	14.8.45
oneStepReceive	Boolean	R	14.8.46
oneStepTransmit	Boolean	R	14.8.47
initialOneStepTxOper	Boolean	RW	14.8.48

Table 14-10—portDS table (continued)

Name	Data type	Operations supported ^a	References
currentOneStepTxOper	Boolean	R	14.8.49
useMgtSettableOneStepTxOper	Boolean	RW	14.8.50
mgtSettableOneStepTxOper	Boolean	RW	14.8.51
syncLocked	Boolean	R	14.8.52
pdelayTruncatedTimestampsArray	UIInteger48[4]	R	14.8.53
minorVersionNumber	UIInteger4	R	14.8.54
gptpCapableStateMachinesEnabled	Boolean	RW	14.8.54a
nrrPdelay	Integer32	R	14.8.54b
nrrSync	Integer32	R	14.8.54c
nrrCompMethod	Enumeration2	RW	14.8.54d
asCapableAcrossDomains	Boolean	R	14.8.54e
NOTE—The values of nup and ndown in Table 14-10 depend on the particular PHY used.			

^a R = Read only access; RW = Read/write access.

1 14.9 Description Port Parameter Data Set (descriptionPortDS)

2 14.9.1 General

3 The descriptionPortDS contains the profileIdentifier for this PTP profile, as specified in F.2.

4 14.9.2 profileIdentifier

5 The value is the profileIdentifier for this PTP profile (see F.2).

6 14.9.3 descriptionPortDS table

7 There is one descriptionPortDS table per PTP Port of a PTP Instance, as detailed in Table 14-11.

Table 14-11—descriptionPortDS table

Name	Data type	Operations supported ^a	References
profileIdentifier	Octet6	R	14.9.2

^a R= Read only access.

1 14.10 Port Parameter Statistics Data Set (portStatisticsDS)

2 14.10.1 General

3 For the single PTP Port of a PTP End Instance and for each PTP Port of a PTP Relay Instance, the
 4 portStatisticsDS provides counters associated with PTP Port capabilities at a given PTP Instance. The
 5 number of such statistics sets is the same as the value of defaultDS.numberPorts.

6 14.10.2 rxSyncCount

7 This counter increments every time synchronization information is received, denoted by one of the following
 8 events:

- 9 — A transition to TRUE from FALSE of the rcvdSync variable of the MDSyncReceiveSM state
 10 machine (see 11.2.14.1.2 and Figure 11-6) when in the DISCARD, WAITING_FOR_SYNC, or
 11 WAITING_FOLLOW_UP states; or
- 12 — rcvdIndication transitions to TRUE (see Figure 12-7).

13 14.10.3 rxOneStepSyncCount

14 This counter increments every time a one-step Sync message is received, denoted by a transition to TRUE
 15 from FALSE of the rcvdSync variable of the MDSyncReceiveSM state machine (see 11.2.14.1.2 and
 16 Figure 11-6) with the variable rcvdSyncPtr->twoStepFlag FALSE when in the DISCARD or
 17 WAITING_FOR_SYNC states.

18 14.10.4 rxFollowUpCount

19 This counter increments every time a Follow_Up message is received, denoted by a transition to TRUE from
 20 FALSE of the rcvdFollowUp variable of the MDSyncReceiveSM state machine (see 11.2.14.1.3 and
 21 Figure 11-6) when in the WAITING_FOLLOW_UP state.

22 14.10.5 rxPdelayRequestCount

23 This counter increments every time a Pdelay_Req message is received, denoted by a transition to TRUE
 24 from FALSE of the rcvdPdelayReq variable of the MDPdelayResp state machine (see 11.2.20.2.1 and
 25 Figure 11-10) when in the WAITING_FOR_PDELAY_REQ or INITIAL_WAITING_FOR_PDELAY_REQ
 26 states.

27 14.10.6 rxPdelayResponseCount

28 This counter increments every time a Pdelay_Resp message is received, denoted by a transition to TRUE
 29 from FALSE of the rcvdPdelayResp variable of the MDPdelayReq state machine (see 11.2.19.2.2 and
 30 Figure 11-9) when in the WAITING_FOR_PDELAY_RESP state.

31 14.10.7 rxPdelayResponseFollowUpCount

32 This counter increments every time a Pdelay_Resp_Follow_Up message is received, denoted by a transition
 33 to TRUE from FALSE of the rcvdPdelayRespFollowUp variable of the MDPdelayReq state machine
 34 (see 11.2.19.2.4 and Figure 11-9) when in the WAITING_FOR_PDELAY_RESP_FOLLOW_UP state.

1 14.10.8 rxAnnounceCount

2 This counter increments every time an Announce message is received, denoted by a transition to TRUE from
 3 FALSE of the rcvdAnnounce variable of the PortAnnounceReceive state machine (see 10.3.11 and
 4 Figure 10-13) when in the DISCARD or RECEIVE states.

5 14.10.9 rxPtpPacketDiscardCount

6 This counter increments every time a PTP message of the respective PTP Instance is discarded, caused by
 7 the occurrence of any of the following conditions:

- 8 a) A received Announce message is not qualified, denoted by the function qualifyAnnounce (see
 9 10.3.11.2.1 and 13.1.3.4) of the PortAnnounceReceive state machine (see 10.3.11 and Figure 10-13)
 10 returning FALSE;
- 11 b) A Follow_Up message corresponding to a received Sync message is not received, denoted by a
 12 transition of the condition (currentTime >= followUpReceiptTimeoutTime) to TRUE from FALSE
 13 when in the WAITING_FOR_FOLLOW_UP state of the MDSyncReceiveSM state machine (see
 14 11.2.14 and Figure 11-6);
- 15 c) A Pdelay_Resp message corresponding to a transmitted Pdelay_Req message is not received,
 16 denoted by a transition from the WAITING_FOR_PDELAY_RESP state to the RESET state of the
 17 MDPdelayReq state machine (see 11.2.19 and Figure 11-9);
- 18 d) A Pdelay_Resp_Follow_Up message corresponding to a transmitted Pdelay_Req message is not
 19 received, denoted by a transition from the WAITING_FOR_PDELAY_RESP_FOLLOW_UP state
 20 to the RESET state of the MDPdelayReq state machine (see 11.2.19 and Figure 11-9).

21 14.10.10 syncReceiptTimeoutCount

22 This counter increments every time sync receipt timeout occurs, denoted by entering the AGED state of the
 23 PortAnnounceInformation state machine (see 10.3.12 and Figure 10-14) with the condition (currentTime >=
 24 syncReceiptTimeoutTime && gmPresent) evaluating to TRUE.

25 14.10.11 announceReceiptTimeoutCount

26 This counter increments every time announce receipt timeout occurs, denoted by entering the AGED state of
 27 the PortAnnounceInformation state machine from the CURRENT state of the PortAnnounceInformation
 28 state machine (see 10.3.12 and Figure 10-14) with the condition (currentTime >=
 29 announceReceiptTimeoutTime) evaluating to TRUE.

30 14.10.12 pdelayAllowedLostResponsesExceededCount

31 This counter increments every time the value of the variable lostResponses (see 11.2.19.2.9) exceeds the
 32 value of the variable allowedLostResponses (see 11.2.13.4), in the RESET state of the MDPdelayReq state
 33 machine (see 11.2.19 and Figure 11-9).

34 14.10.13 txSyncCount

35 This counter increments every time synchronization information is transmitted, denoted by one of the
 36 following events:

- 37 — A transition to TRUE from FALSE of the rcvdMDSyncMDSS variable of the MDSyncSendSM
 38 state machine (see 11.2.15.1.1 and Figure 11-7) when in the INITIALIZING, SEND_FOLLOW_UP,
 39 or SET_CORRECTION_FIELD states; or

- 1 — The INITIATE_REQUEST_WAIT_CONFIRM_OR_SAVE_INFO state is entered in Figure 12-5
- 2 and TM is being used [i.e., $(\text{tmFtmSupport} == 0x01) \parallel (\text{tmFtmSupport} \& 0x01 == 0x01) \&\&$
- 3 $\text{ftmReqGrantedTimeTransmitter})$ in timeTransmitter state machine A of Figure 12-5]; or
- 4 — The INITIATE_REQUEST_WAIT_CONFIRM state is entered in Figure 12-6 (in this case FTM is
- 5 being used).

6 **14.10.14 txOneStepSyncCount**

7 This counter increments every time a one-step Sync message is transmitted, denoted by a transition to TRUE
 8 from FALSE of the rcvdMDSyncMDSS variable of the MDSyncSendSM state machine (see 11.2.15.1.1 and
 9 Figure 11-7) with the variable oneStepTxOper TRUE when in the INITIALIZING,
 10 SEND_SYNC_ONE_STEP, or SET_CORRECTION_FIELD states.

11 **14.10.15 txFollowUpCount**

12 This counter increments every time a Follow_Up message is transmitted, denoted by a transition to TRUE
 13 from FALSE of the rcvdMDTimestampReceive variable of the MDSyncSendSM state machine (see
 14 11.2.15.1.3 and Figure 11-7) when in the SEND_SYNC state.

15 **14.10.16 txPdelayRequestCount**

16 This counter increments every time a Pdelay_Req message is transmitted, denoted by entering the
 17 INITIAL_SEND_PDELAY_REQ or SEND_PDELAY_REQ states of the MDPdelayReq state machine (see
 18 11.2.19 and Figure 11-9).

19 **14.10.17 txPdelayResponseCount**

20 This counter increments every time a Pdelay_Resp message is transmitted, denoted by the following events:

- 21 — A transition to TRUE from FALSE of the rcvdPdelayReq variable of the MDPdelayResp state
 22 machine (see 11.2.20.2.1 and Figure 11-10) when in the WAITING_FOR_PDELAY_REQ or
 23 INITIAL_WAITING_FOR_PDELAY_REQ state and
- 24 — The resulting entry to the SENT_PDELAY_RESP_WAITING_FOR_TIMESTAMP state.

25 **14.10.18 txPdelayResponseFollowUpCount**

26 This counter increments every time a Pdelay_Resp_Follow_Up message is transmitted, denoted by the
 27 following events:

- 28 — A transition to TRUE from FALSE of the rcvdMDTimestampReceive variable of the
 29 MDPdelayResp state machine (see 11.2.20.2.2 and Figure 11-10) when in the
 30 SENT_PDELAY_RESP_WAITING_FOR_TIMESTAMP states and
- 31 — The resulting entry to the WAITING_FOR_PDELAY_REQ state.

32 **14.10.19 txAnnounceCount**

33 This counter increments every time an Announce message is transmitted, denoted by entering the
 34 TRANSMIT_ANNOUNCE state of the PortAnnounceReceive state machine (see 10.3.16 and Figure 10-18).

35 **14.10.19a rxSyncCountTimeReceiverP**

36 This counter increments whenever time synchronization information is received on a PTP Port when its port
 37 state is TimeReceiverPort. The receipt of time synchronization information is denoted by one of the following
 38 events:

- 1 — A transition to TRUE from FALSE of the rcvdSync variable of the MDSyncReceiveSM state machine (see 11.2.14.1.2 and Figure 11-6) when in the DISCARD, WAITING_FOR_SYNC, or WAITING_FOLLOW_UP states; or
- 2 — rcvdIndication transitions to TRUE (see Figure 12-7).

5 This counter is initialized to zero, and resets to zero when the port state is not TimeReceiverPort.

6 14.10.20 portStatisticsDS table

7 There is one portStatisticsDS table per PTP Port, per PTP Instance of a time-aware system. Each portStatisticsDS table contains a set of counters for each PTP Port that supports the time-synchronization capability, as detailed in Table 14-12. Each table can be created or removed dynamically in implementations that support dynamic configuration of PTP Ports and components.

Table 14-12—portStatisticsDS table

Name	Data type	Operations supported ^a	References
rxSyncCount	UInteger32	R	14.10.2
rxOneStepSyncCount	UInteger32	R	14.10.3
rxFollowUpCount	UInteger32	R	14.10.4
rxPdelayRequestCount	UInteger32	R	14.10.5
rxPdelayResponseCount	UInteger32	R	14.10.6
rxPdelayResponseFollowUpCount	UInteger32	R	14.10.7
rxAnnounceCount	UInteger32	R	14.10.8
rxPtpPacketDiscardCount	UInteger32	R	14.10.9
syncReceiptTimeoutCount	UInteger32	R	14.10.10
announceReceiptTimeoutCount	UInteger32	R	14.10.11
pdelayAllowedLostResponsesExceededCount	UInteger32	R	14.10.12
txSyncCount	UInteger32	R	14.10.13
txOneStepSyncCount	UInteger32	R	14.10.14
txFollowUpCount	UInteger32	R	14.10.15
txPdelayRequestCount	UInteger32	R	14.10.16
txPdelayResponseCount	UInteger32	R	14.10.17
txPdelayResponseFollowUpCount	UInteger32	R	14.10.18
txAnnounceCount	UInteger32	R	14.10.19
rxSyncCountTimeReceiverP	UInteger32	R	14.10.19a

^aR= Read only access.

1 **14.11 Acceptable TimeTransmitter Port Parameter Data Set**
2 **(acceptableTimeTransmitterPortDS)**

3 **14.11.1 General**

4 The acceptableTimeTransmitterPortDS represents the capability to enable/disable the acceptable
5 timeTransmitter table feature on a PTP Port. For the single PTP Port of a PTP End Instance and for each PTP
6 Port of a PTP Relay Instance, this data set contains the single member
7 acceptableTimeTransmitterTableEnabled, which is used to enable/disable the Acceptable TimeTransmitter
8 Table Feature. The number of such data sets is the same as the value of defaultDS.numberPorts.

9 **14.11.2 acceptableTimeTransmitterTableEnabled**

10 The value is equal to the value of the Boolean acceptableTimeTransmitterTableEnabled (see 13.1.3.2 and
11 13.1.3.5).

12 **14.11.3 acceptableTimeTransmitterPortDS table**

13 There is one acceptableTimeTransmitterPortDS table per PTP Port, per PTP Instance of a time-aware system
14 as detailed in Table 14-13.

Table 14-13—acceptableTimeTransmitterPortDS table

Name	Data type	Operations supported ^a	References
acceptableTimeTransmitterTableEnabled	Boolean	RW	14.11.2

^a R = Read only access; RW = Read/write access.

15 **14.12 External Port Configuration Port Parameter Data Set**
16 **(externalPortConfigurationPortDS)**

17 **14.12.1 General**

18 The externalPortConfigurationPortDS is used with the external port configuration option to indicate the
19 desired state for the PTP Port. This data set contains the single member desiredState, which indicates
20 the desired state for the PTP Port. The number of such data sets is the same as the value of
21 defaultDS.numberPorts.

22 **14.12.2 desiredState**

23 When the value of defaultDS.externalPortConfigurationEnabled is TRUE, the value of
24 externalPortConfigurationPortDS.desiredState is the desired state of the PTP Port. This member sets the
25 value of the variable portStateInd (see 10.3.15.1.5). When a new value is written to the member by
26 management, the variable rcvdPortStateInd (see 10.3.15.1.4) is set to TRUE.

1 14.12.3 externalPortConfigurationPortDS table

2 There is one externalPortConfigurationPortDS table per gPTPport, per PTP Instance of a time-aware system
 3 as detailed in Table 14-14.

Table 14-14—externalPortConfigurationPortDS table

Name	Data type	Operations supported ^a	References
desiredState	Enumeration8 (see 6.4.2)	RW	14.12.2, Table 14-7

^a R = Read only access; RW = Read/write access.

4 14.13 Asymmetry Measurement Mode Parameter Data Set 5 (asymmetryMeasurementModeDS)

6 14.13.1 General

7 The asymmetryMeasurementModeDS represents the capability to enable/disable the Asymmetry Compensation Measurement Procedure on a PTP Port (see Annex G). This data set is used instead of the cmldsAsymmetryMeasurementModeDS when only domain 0 is present and CMLDS is not used.

10 For the single PTP Port of a PTP End Instance and for each PTP Port of a PTP Relay Instance, the
 11 asymmetryMeasurementModeDS contains the single member asymmetryMeasurementMode, which is used
 12 to enable/disable the Asymmetry Compensation Measurement Procedure. The number of such data sets is
 13 the same as the value of defaultDS.numberPorts.

14 14.13.2 asymmetryMeasurementMode

15 The value is equal to the value of the Boolean asymmetryMeasurementMode (see G.3). For full-duplex
 16 IEEE 802.3 media, the value is TRUE if an asymmetry measurement is being performed for the link
 17 attached to this PTP Port and FALSE otherwise. For all other media, the value shall be FALSE
 18 (see 10.2.5.2). Setting the managed object asymmetryMeasurementMode causes the Boolean
 19 asymmetryMeasurementMode to have the same value.

20 NOTE—If an asymmetry measurement is being performed for a link, asymmetryMeasurementMode must be TRUE for
 21 the PTP Ports at each end of the link.

22 14.13.3 asymmetryMeasurementModeDS table

23 There is one asymmetryMeasurementModeDS table for the single PTP Instance whose domainNumber is 0,
 24 per PTP Port, as detailed in Table 14-15. This data set is used only when there is a single gPTP domain and
 25 CMLDS is not used.

Table 14-15—asymmetryMeasurementModeDS table

Name	Data type	Operations supported ^a	References
asymmetryMeasurementMode	Boolean	RW	14.13.2

^a R = Read only access; RW = Read/write access.

1 14.14 Common Services Port Parameter Data Set (commonServicesPortDS)

2 14.14.1 General

3 The commonServicesPortDS enables a PTP Port of a PTP Instance to determine which port of the respective
4 common service corresponds to that PTP Port.

5 At present, the only common service specified is the CMLDS, and the only member of the
6 commonServicesPortDS is the cmldsLinkPortPortNumber. This member contains the port number of the
7 CMLDS Link Port that corresponds to this PTP Port.

8 14.14.2 cmldsLinkPortPortNumber

9 The value is the portNumber attribute of the cmldsLinkPortDS.portIdentity (see 14.16.2) of the Link Port
10 that corresponds to this PTP Port.

11 14.14.3 commonServicesPortDS table

12 There is one commonServicesPortDS table per PTP Port, per PTP Instance of a time-aware system as
13 detailed in Table 14-16.

Table 14-16—commonServicesPortDS table

Name	Data type	Operations supported ^a	References
cmldsLinkPortPortNumber	UIInteger16	R	14.14.2

^a R = Read only access.

14 14.15 Common Mean Link Delay Service Default Parameter Data Set (cmldsDefaultDS)

15 14.15.1 General

16 The cmldsDefaultDS describes the per-time-aware-system attributes of the Common Mean Link Delay
17 Service.

18 NOTE—The value of sdId for the Common Mean Link Delay Service is fixed as 0x200 (see 11.2.17) and cannot be
19 changed. Therefore, a corresponding data set member for sdId is not needed.

20 14.15.2 clockIdentity

21 The value is the clockIdentity (see 8.5.2.2) that will be used to identify the Common Mean Link Delay
22 Service.

23 14.15.3 numberLinkPorts

24 The value is the number of Link Ports of the time-aware system on which the Common Mean Link Delay
25 Service is implemented. For an end station the value is 1.

26 14.15.4 cmldsDefaultDS table

27 There is one cmldsDefaultDS table per time-aware system, as detailed in Table 14-17.

Table 14-17—cmldsDefaultDS table

Name	Data type	Operations supported ^a	References
clockIdentity	ClockIdentity	R	14.15.2
numberLinkPorts	UIInteger16	R	14.15.3

^a R = Read only access; RW = Read/write access.

1 14.16 Common Mean Link Delay Service Link Port Parameter Data Set

2 (cmldsLinkPortDS)

3 14.16.1 General

4 The cmldsLinkPortDS represents time-aware Link Port capabilities for the Common Mean Link Delay Service of a Link Port of a time-aware system.

6 For every Link Port of the Common Mean Link Delay Service of a time-aware system, the cmldsLinkPortDS is maintained as the basis for making protocol decisions and providing values for message fields. The number of such data sets is the same as the value of cmldsDefaultDS.numberLinkPorts.

9 14.16.2 portIdentity

10 The value is the portIdentity attribute of the local port (see 8.5.2).

11 14.16.3 cmldsLinkPortEnabled

12 The value is equal to the value of the Boolean cmldsLinkPortEnabled (see 11.2.18.1).

13 14.16.4 isMeasuringDelay

14 The value is equal to the value of the **instance of the** Boolean isMeasuringDelay (see 11.2.13.6 and 16.4.3.3) **that is per Link Port and across all domains.**

16 14.16.5 asCapableAcrossDomains

17 The value is equal to the value of the Boolean asCapableAcrossDomains (see 11.2.2 and 11.2.13.12).

18 14.16.6 meanLinkDelay

19 The value is equal to the value of the per-port global variable meanLinkDelay (see 10.2.5.8). It is an estimate of the current one-way propagation time on the link attached to this Link Port, measured as specified for the respective medium (see 11.2.17, 12.5.2, and 16.4). The value is zero for Link Ports attached to IEEE 802.3 EPON links and for the timeTransmitter port of an IEEE 802.11 link because one-way propagation delay is not measured on the latter and not directly measured on the former. The data type shall be TimeInterval. The default value is zero.

25 NOTE—The underlying per-port global variable meanLinkDelay is of type UScaledNS, which is a 96-bit value (see 6.4.3.2). meanLinkDelay values that are larger than the maximum value that can be represented by the TimeInterval data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this managed object are set to this largest value.

1 14.16.7 meanLinkDelayThresh

2 The value is equal to the value of the per-Link-Port global variable meanLinkDelayThresh (see 11.2.13.7). It
 3 is the propagation time threshold above which a Link Port (and therefore any PTP Ports that use the CMLDS
 4 on this Link Port) is considered not capable of participating in the IEEE 802.1AS protocol. Setting the
 5 managed object meanLinkDelayThresh causes the per-Link-Port global variable meanLinkDelayThresh to
 6 have the same value.

7 NOTE—The underlying per-port global variable meanLinkDelayThresh is of type UScaledNS, which is a 96-bit value
 8 (see 6.4.3.2). meanLinkDelayThresh values that are larger than the maximum value that can be represented by the
 9 TimeInterval data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this managed
 10 object are set to this largest value.

11 14.16.8 delayAsymmetry

12 The value is the asymmetry in the propagation delay on the link attached to this Link Port relative to the
 13 local clock, as defined in 10.2.5.9 and 8.3. If propagation delay asymmetry is not modeled, then
 14 delayAsymmetry is 0.

15 NOTE—The underlying per-port global variable delayAsymmetry is of type ScaledNS, which is a 96-bit value
 16 (see 6.4.3.1). delayAsymmetry values that are larger than the maximum value that can be represented by the
 17 TimeInterval data type, i.e., 0x7FFF FFFF FFFF FFFF (where the units are 2^{-16} ns; see 6.4.3.3), used for this
 18 managed object are set to this largest value. delayAsymmetry values that are less than the minimum value that can be
 19 represented by the TimeInterval data type, i.e., 0x8000 0000 0000 0001 written in two's complement form (where the
 20 units are 2^{-16} ns; see 6.4.3.3), used for this managed object are set to this smallest value.

21 14.16.9 nrrPdelay

22 The value is an estimate of the ratio of the frequency of the LocalClock entity of the time-aware system at
 23 the other end of the link attached to this Link Port, to the frequency of the LocalClock entity of this time-
 24 aware system (see 10.2.5.7). nrrPdelay is expressed as the fractional frequency offset stored in the global
 25 variable nrrPdelay (see 11.2.13.13) multiplied by 2^{41} , i.e., the quantity $(\text{nrrPdelay} - 1.0)(2^{41})$. The default
 26 value of nrrPdelay is 1.0.

27 NOTE—This data set member corresponds to the scaledNeighborRateRatio member of the
 28 CommonMeanLinkDelayInformation Structure in 16.6.3.2 of IEEE Std 1588-2019.

29 14.16.10 initialLogPdelayReqInterval

30 If useMgtSettableLogPdelayReqInterval is FALSE, then for full-duplex IEEE 802.3 media and for CSN
 31 media that use the peer-to-peer delay mechanism to measure path delay (see 16.4.3.2), the value is the
 32 logarithm to base 2 of the Pdelay_Req message transmission interval used when:

- 33 a) The Link Port is initialized, or
- 34 b) A message interval request TLV is received with the logLinkDelayInterval field set to 126 (see
 35 11.5.2.2 and the LinkDelayIntervalSetting state machine in 11.2.21).

36 For all other media, the value is 127.

37 14.16.11 currentLogPdelayReqInterval

38 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 39 path delay (see 16.4.3.2), the value is the logarithm to the base 2 of the current Pdelay_Req message
 40 transmission interval (see 11.5.2.2).

1 For all other media, the value is 127.

2 **14.16.12 useMgtSettableLogPdelayReqInterval**

3 The managed object is a Boolean that determines the source of the sync interval and mean time interval
4 between successive Pdelay_Req messages. If the value is TRUE, the value of currentLogPdelayReqInterval
5 is set equal to the value of mgtSettableLogPdelayReqInterval (see 14.16.13). If the value of the managed
6 object is FALSE, the value of currentLogPdelayReqInterval is determined by the LinkDelayIntervalSetting
7 state machine (see 11.2.21). The default value of useMgtSettableLogPdelayReqInterval is FALSE.

8 **14.16.13 mgtSettableLogPdelayReqInterval**

9 The value is the logarithm to base 2 of the mean time interval between successive Pdelay_Req messages
10 if useMgtSettableLogPdelayReqInterval is TRUE. The value is not used if
11 useMgtSettableLogPdelayReqInterval is FALSE.

12 **14.16.14 initialComputeNeighborRateRatio**

13 If useMgtSettableComputeNeighborRateRatio is FALSE, then for full-duplex IEEE 802.3 media and for
14 CSN media that use the peer-to-peer delay mechanism to measure path delay (see 16.4.3.2), the value is the
15 initial value of computeNeighborRateRatio (see 10.2.5.10).

16 For all other media, the value is TRUE.

17 **14.16.15 currentComputeNeighborRateRatio**

18 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
19 path delay (see 16.4.3.2), the value is the current value of computeNeighborRateRatio.

20 For all other media, the value is TRUE.

21 **14.16.16 useMgtSettableComputeNeighborRateRatio**

22 The managed object is a Boolean that determines the source of the value of computeNeighborRateRatio. If
23 the value is TRUE, the value of computeNeighborRateRatio is set equal to the value of
24 mgtSettablecomputeNeighborRateRatio (see 14.16.17). If the value of the managed object is FALSE, the
25 value of currentComputeNeighborRateRatio is determined by the LinkDelayIntervalSetting state machine
26 (see 11.2.21). The default value of useMgtSettableComputeNeighborRateRatio is FALSE.

27 **14.16.17 mgtSettableComputeNeighborRateRatio**

28 computeNeighborRateRatio is configured to this value if useMgtSettableComputeNeighborRateRatio is
29 TRUE. The value is not used if useMgtSettableComputeNeighborRateRatio is FALSE.

30 **14.16.18 initialComputeMeanLinkDelay**

31 If useMgtSettableComputeMeanLinkDelay is FALSE, then for full-duplex IEEE 802.3 media and for CSN
32 media that use the peer-to-peer delay mechanism to measure path delay (see 16.4.3.2), the value is the initial
33 value of computeMeanLinkDelay (see 10.2.5.10).

34 For all other media, the value is TRUE.

1 14.16.19 currentComputeMeanLinkDelay

2 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 3 path delay (see 16.4.3.2), the value is the current value of computeMeanLinkDelay.

4 For all other media, the value is TRUE.

5 14.16.20 useMgtSettableComputeMeanLinkDelay

6 The managed object is a Boolean that determines the source of the value of computeMeanLinkDelay. If the
 7 value is TRUE, the value of computeMeanLinkDelay is set equal to the value of
 8 mgtSettableComputeMeanLinkDelay (see 14.16.17). If the value of the managed object is FALSE, the value
 9 of currentComputeMeanLinkDelay is determined by the LinkDelayIntervalSetting state machine
 10 (see 11.2.21). The default value of useMgtSettableComputeMeanLinkDelay is FALSE.

11 14.16.21 mgtSettableComputeMeanLinkDelay

12 computeMeanLinkDelay is configured to this value if useMgtSettableComputeMeanLinkDelay is TRUE.
 13 The value is not used if useMgtSettableComputeMeanLinkDelay is FALSE.

14 14.16.22 allowedLostResponses

15 The value is equal to the value of the per-Link-Port global variable allowedLostResponses (see 11.5.3 and
 16 11.2.13.4). It is the number of Pdelay_Req messages without valid responses above which a Link Port is
 17 considered to be not exchanging peer delay messages with its neighbor. Setting the managed object
 18 allowedLostResponses causes the per-Link-Port global variable allowedLostResponses to have the same value.

19 14.16.23 allowedFaults

20 The value is equal to the value of the per-Link-Port global variable allowedFaults (see 11.5.4 and 11.2.13.5).
 21 It is the number of faults (see 11.5.4) above which asCapableAcrossDomains is set to FALSE, i.e., a Link
 22 Port is considered not capable of interoperating with its neighbor via the IEEE 802.1AS protocol
 23 (see 10.2.5.1). Setting the managed object allowedFaults causes the per-Link-Port global variable
 24 allowedFaults to have the same value.

25 14.16.24 versionNumber

26 This value is set to versionPTP as specified in 10.6.2.2.4.

27 14.16.25 pdelayTruncatedTimestampsArray

28 For full-duplex IEEE 802.3 media and for CSN media that use the peer-to-peer delay mechanism to measure
 29 path delay (see 16.4.3.2), the values of the four elements of this array are as described in Table 14-9. For all
 30 other media, the values are zero. Array elements 0, 1, 2, and 3 correspond to the timestamps t1, t2, t3, and t4,
 31 modulo 2^{32} , respectively, in Figure 11-1 and are expressed in units of 2^{-16} ns (i.e., the value of each
 32 array element is equal to the remainder obtained upon dividing the respective timestamp, expressed in units
 33 of 2^{-16} ns, by 2^{48}). At any given time, the timestamp values stored in the array are for the same, and most
 34 recently completed, peer delay message exchange.

35 NOTE—This managed object is used with the asymmetry measurement compensation procedure, which is based on
 36 line-swapping.

37 14.16.26 minorVersionNumber

38 This value is set to minorVersionPTP as specified in 10.6.2.2.3.

14.16.27 cmldsLinkPortDS table

2 There is one cmldsLinkPortDS table per Link Port, for the PTP Instance of a time-aware system. Each
 3 cmldsLinkPortDS table contains a set of parameters for each Link Port that supports the time-
 4 synchronization capability, as detailed in Table 14-18. Each table can be created or removed dynamically in
 5 implementations that support dynamic configuration of Link Ports and components.

Table 14-18—cmldsLinkPortDS table

Name	Data type	Operations supported ^a	References
portIdentity	PortIdentity (see 6.4.3.7)	R	14.16.2
cmldsLinkPortEnabled	Boolean	R	14.16.3
isMeasuringDelay	Boolean	R	14.16.4
asCapableAcrossDomains	Boolean	R	14.16.5
meanLinkDelay	TimeInterval	R	14.16.6
meanLinkDelayThresh	TimeInterval	RW	14.16.7
delayAsymmetry	TimeInterval	RW	14.16.8
nrrPdelay	Integer32	R	14.16.9
initialLogPdelayReqInterval	Integer8	RW	14.16.10
currentLogPdelayReqInterval	Integer8	R	14.16.11
useMgtSettableLogPdelayReqInterval	Boolean	RW	14.16.12
mgtSettableLogPdelayReqInterval	Integer8	RW	14.16.13
initialComputeNeighborRateRatio	Boolean	RW	14.16.14
currentComputeNeighborRateRatio	Boolean	R	14.16.15
useMgtSettableComputeNeighborRateRatio	Boolean	RW	14.16.16
mgtSettableComputeNeighborRateRatio	Boolean	RW	14.16.17
initialComputeMeanLinkDelay	Boolean	RW	14.16.18
currentComputeMeanLinkDelay	Boolean	R	14.16.19
useMgtSettableComputeMeanLinkDelay	Boolean	RW	14.16.20
mgtSettableComputeMeanLinkDelay	Boolean	RW	14.16.21
allowedLostResponses	UInteger8	RW	14.16.22
allowedFaults	UInteger8	RW	14.16.23
versionNumber	UInteger4	R	14.16.24
pdelayTruncatedTimestampsArray	UInteger48[4]	R	14.16.25
minorVersionNumber	UInteger4	R	14.16.26

^a R = Read only access; RW = Read/write access.

**1 14.17 Common Mean Link Delay Service Link Port Parameter Statistics Data Set
2 (cmldsLinkPortStatisticsDS)**

3 14.17.1 General

4 For every Link Port of the Common Mean Link Delay Service of a time-aware system, the
5 cmldsLinkPortStatisticsDS provides counters associated with Link Port capabilities at a given time-aware
6 system. The number of such statistics sets is the same as the value of cmldsDefaultDS.numberLinkPorts.

7 14.17.2 rxPdelayRequestCount

8 This counter increments every time a Pdelay_Req message is received, denoted by a transition to TRUE
9 from FALSE of the rcvdPdelayReq variable of the MDPdelayResp state machine (see 11.2.20.2.1 and
10 Figure 11-10) when in the WAITING_FOR_PDELAY_REQ or INITIAL_WAITING_FOR_PDELAY_REQ
11 states.

12 14.17.3 rxPdelayResponseCount

13 This counter increments every time a Pdelay_Resp message is received, denoted by a transition to TRUE
14 from FALSE of the rcvdPdelayResp variable of the MDPdelayReq state machine (see 11.2.19.2.2 and
15 Figure 11-9) when in the WAITING_FOR_PDELAY_RESP state.

16 14.17.4 rxPdelayResponseFollowUpCount

17 This counter increments every time a Pdelay_Resp_Follow_Up message is received, denoted by a transition
18 to TRUE from FALSE of the rcvdPdelayRespFollowUp variable of the MDPdelayReq state machine (see
19 11.2.19.2.4 and Figure 11-9) when in the WAITING_FOR_PDELAY_RESP_FOLLOW_UP state.

20 14.17.5 rxPtpPacketDiscardCount

21 This counter increments every time a PTP message of the Common Mean Link Delay Service is discarded,
22 caused by the occurrence of any of the following conditions:

- 23 — A Pdelay_Resp message corresponding to a transmitted Pdelay_Req message is not received,
24 denoted by a transition from the WAITING_FOR_PDELAY_RESP state to the RESET state of the
25 MDPdelayReq state machine (see 11.2.19 and Figure 11-9).
- 26 — A Pdelay_Resp_Follow_Up message corresponding to a transmitted Pdelay_Req message is not
27 received, denoted by a transition from the WAITING_FOR_PDELAY_RESP_FOLLOW_UP state
28 to the RESET state of the MDPdelayReq state machine (see 11.2.19 and Figure 11-9).

29 14.17.6 pdelayAllowedLostResponsesExceededCount

30 This counter increments every time the value of the variable lostResponses (see 11.2.19.2.9) exceeds the
31 value of the variable allowedLostResponses (see 11.2.13.4) in the RESET state of the MDPdelayReq state
32 machine (see 11.2.19 and Figure 11-9).

33 14.17.7 txPdelayRequestCount

34 This counter increments every time a Pdelay_Req message is transmitted, denoted by entering the
35 INITIAL_SEND_PDELAY_REQ or SEND_PDELAY_REQ states of the MDPdelayReq state machine
36 (see 11.2.19 and Figure 11-9).

1 14.17.8 txPdelayResponseCount

2 This counter increments every time a Pdelay_Resp message is transmitted, denoted by the following events:

- 3 — A transition to TRUE from FALSE of the rcvdPdelayReq variable of the MDPdelayResp state machine (see 11.2.20.2.1 and Figure 11-10) when in the WAITING_FOR_PDELAY_REQ or INITIAL_WAITING_FOR_PDELAY_REQ states and
- 6 — The resulting entry to the SENT_PDELAY_RESP_WAITING_FOR_TIMESTAMP state.

7 14.17.9 txPdelayResponseFollowUpCount

8 This counter increments every time a Pdelay_Resp_Follow_Up message is transmitted, denoted by the following events:

- 10 — A transition to TRUE from FALSE of the rcvdMDTimestampReceiveMDPResp variable of the MDPdelayResp state machine (see 11.2.20.2.2 and Figure 11-10) when in the SENT_PDELAY_RESP_WAITING_FOR_TIMESTAMP state and
- 13 — The resulting entry to the WAITING_FOR_PDELAY_REQ state.

14 14.17.10 cmldsLinkPortStatisticsDS table

15 There is one cmldsLinkPortStatisticsDS table per Link Port of a time-aware system. The cmldsLinkPortStatisticsDS table contains a set of counters for each Link Port that supports the time-synchronization capability, as detailed in Table 14-19. Each table can be created or removed dynamically in implementations that support dynamic configuration of Link Ports and components.

Table 14-19—cmldsLinkPortStatisticsDS table

Name	Data type	Operations supported ^a	References
rxPdelayRequestCount	UIInteger32	R	14.17.2
rxPdelayResponseCount	UIInteger32	R	14.17.3
rxPdelayResponseFollowUpCount	UIInteger32	R	14.17.4
rxPtpPacketDiscardCount	UIInteger32	R	14.17.5
pdelayAllowedLostResponsesExceededCount	UIInteger32	R	14.17.6
txPdelayRequestCount	UIInteger32	R	14.17.7
txPdelayResponseCount	UIInteger32	R	14.17.8
txPdelayResponseFollowUpCount	UIInteger32	R	14.17.9

^a R = Read only access.

1 14.18 Common Mean Link Delay Service Asymmetry Measurement Mode Parameter Data

2 Set (cmldsAsymmetryMeasurementModeDS)

3 14.18.1 General

4 The cmldsAsymmetryMeasurementModeDS represents the capability to enable/disable the Asymmetry
5 Compensation Measurement Procedure on a Link Port (see Annex G).

6 For every Link Port of the Common Mean Link Delay Service of a time-aware system, the
7 cmldsAsymmetryMeasurementModeDS contains the single member asymmetryMeasurementMode, which
8 is used to enable/disable the Asymmetry Compensation Measurement Procedure. The number of such data
9 sets is the same as the numberLinkPorts value of the cmldsDefaultDS.

10 14.18.2 asymmetryMeasurementMode

11 The value is equal to the value of the Boolean asymmetryMeasurementMode (see G.3). For full-duplex
12 IEEE 802.3 media, the value is TRUE if an asymmetry measurement is being performed for the link
13 attached to this Link Port and FALSE otherwise. For all other media, the value shall be FALSE
14 (see 10.2.5.2). Setting the managed object asymmetryMeasurementMode causes the Boolean
15 asymmetryMeasurementMode to have the same value.

16 NOTE—If an asymmetry measurement is being performed for a link, asymmetryMeasurementMode must be TRUE for
17 the Link Ports at each end of the link.

18 14.18.3 cmldsAsymmetryMeasurementModeDS table

19 There is one cmldsAsymmetryMeasurementModeDS table for all PTP Instances, per Link Port, as detailed
20 in Table 14-20.

Table 14-20—cmldsAsymmetryMeasurementModeDS table

Name	Data type	Operations supported ^a	References
asymmetryMeasurementMode	Boolean	RW	14.18.2

^a R = Read only access; RW = Read/write access.

21 14.19 Hot Standby System Parameter Data Set (hotStandbySystemDS)

22 14.19.1 General

23 The hotStandbySystemDS describes the attributes of the respective instance of the Hot Standby Service.

24 14.19.2 primaryPtInstIndex

25 The value of primaryPtInstIndex is the index (see 14.1.1) of the primary PTP Instance associated with
26 this hotStandbySystem instance.

27 14.19.3 secondaryPtInstIndex

28 The value of secondaryPtInstIndex is the index (see 14.1.1) of the secondaryPTP Instance associated
29 with this hotStandbySystem instance.

1 **14.19.4 hotStandbySystemEnable**

2 The value is the hotStandbySystemEnable attribute of the HotStandbySystem entity (see 18.5.1.2).

3 **14.19.5 hotStandbySystemState**

4 The value of hotStandbySystemState is the state of the hotStandbySystem, i.e., the value of the global
5 variable hotStandbySystemState (see 18.5.1.1).

6 **14.19.6 hotStandbySystemSplitFunctionality**

7 If the value is TRUE, the optional split functionality (see 18.5.3.4) is used. If the value is FALSE, the
8 optional split functionality is not used.

9 **14.19.7 primarySecondaryOffset**

10 The absolute value of the difference between the clockTimeReceiverTimes (see 10.2.4.3) of the primary and
11 secondary PTP Instances.

12 **14.19.8 primarySecondaryOffsetThresh**

13 The threshold for hotStandbySystemDS.primarySecondaryOffset (see 14.19.7), above which the
14 hotStandbySystemState transitions from REDUNDANT to NOT_REDUNDANT, or does not transition from
15 NOT_REDUNDANT or OUT_OF_SYNC to REDUNDANT even if other conditions for these transitions
16 are satisfied.

17 **14.19.9 hotStandbySystemLogSyncTimeThresh**

18 The value of hotStandbySystemLogSyncTimeThresh is the logarithm to base 2 of the time interval, in
19 seconds, after which the hotStandbySystem transitions from the OUT_OF_SYNC state to either the NOT-
20 REDUNDANT or REDUNDANT state, or from the NOT_REDUNDANT to the REDUNDANT state, if all
21 other conditions for the respective transition are met. The value -128 means that the transition time is zero,
22 i.e., the transition occurs immediately.

23 **14.19.10 hotStandbySystemDS table**

24 There is one hotStandbyDS table per time-aware system, as detailed in Table 14-17.

Table 14-21—hotStandbySystemDS table

Name	Data type	Operations supported ^a	References
primaryPtpInstanceIndex	UInteger32	RW	14.19.2
secondaryPtpInstanceIndex	UInteger32	RW	14.19.3
hotStandbySystemEnable	Boolean	RW	14.19.4
hotStandbySystemState	Enumeration8	R	14.19.5
hotStandbySystemSplitFunctionality	Boolean	RW	14.19.6
primarySecondaryOffset	ScaledNs	R	14.19.7
primarySecondaryOffsetThresh	ScaledNs	RW	14.19.8
hotStandbySystemLogSyncTimeThresh	Integer8	RW	14.19.9

^a R = Read only access; RW = Read/write access.

1 **14.20 Hot Standby System Description Parameter Data Set**

2 **(hotStandbySystemDescriptionDS)**

3 **14.20.1 General**

4 The hotStandbySystemDescriptionDS contains descriptive information for the respective instance of the Hot Standby Service.

6 **14.20.2 userDescription**

7 The user description is a character string whose maximum length is 128.

8 **14.20.3 hotStandbySystemDescriptionDS table**

9 There is one hotStandbySystemDescriptionDS table per hotStandbySystem instance, as detailed in Table 14-22.

Table 14-22—hotStandbySystemDescriptionDS table

Name	Data type	Operations supported ^a	References
userDescription	Octet128	RW	14.20.2

^a RW= Read/write access.

1 15. Management Information Base (MIB)

2 <<Editor's note: There is no MIB for ASdm, so we need to be clear that this clause only addresses the
 3 managed objects of IEEE Std 802.1AS-2020. Should this clause be removed? Comments are welcome
 4 to address this question.>>

5 15.1 Internet Standard Management Framework

6 For a detailed overview of the documents that describe the current Internet Standard Management
 7 Framework, refer to section 7 of IETF RFC 3410 (Dec. 2002).

8 Managed objects are accessed via a virtual information store, termed the *Management Information Base*
 9 (MIB). MIB objects are generally accessed through the Simple Network Management Protocol (SNMP).
 10 Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information
 11 (SMI). This clause specifies a MIB module that is compliant to the SMIV2, which is described in
 12 IETF STD 58, comprising IETF RFC 2578 [B11], IETF RFC 2579 [B12], and IETF RFC 2580 .

13 This clause contains a complete SMIV2 MIB set for all features of this standard.

14 15.2 Structure of the MIB

15 The IEEE 802.1AS MIB provides objects to configure and manage the IEEE 802.1AS timing and
 16 synchronization for time-sensitive applications.

17 The MIB contains a set of textual conventions and is additionally subdivided into the following subtrees,
 18 each of which is organized as a set of related objects:

- 19 a) The Default Parameter Data Set (defaultDS) represents the native capabilities of a PTP Instance.
- 20 b) The Current Parameter Data Set (currentDS) represents topological position of a local PTP Instance
 relative to the Grandmaster PTP Instance.
- 22 c) The Parent Parameter Data Set (parentDS) represents capabilities of the upstream PTP Instance
 toward the Grandmaster PTP Instance, as measured at a local PTP Instance.
- 24 d) The Time Properties Parameter Data Set (timePropertiesDS) represents capabilities of the
 Grandmaster PTP Instance, as measured at a local PTP Instance.
- 26 e) The Path Trace Parameter Data Set (pathTraceDS) represents the current path trace information
 (see 10.3.9.23) available at the PTP Instance.
- 28 f) The Acceptable TimeTransmitter Table Parameter Data Set (acceptableTimeTransmitterTableDS)
 represents the acceptable timeTransmitter table used when the media-dependent PTP Port type of
 EPON is present in a PTP Instance.
- 31 g) The Port Parameter Data Set (portDS) represents time-aware capabilities at a given PTP Port, as a
 set of augmentation to the interface table entry (ifEntry).
- 33 h) The Description Port Parameter Data Set (descriptionPortDS) contains the profileIdentifier for this
 PTP profile as specified in F.2.
- 35 i) The Port Parameter Statistics Data Set (portStatisticsDS) represents statistics and counters
 associated with time-aware capabilities at a given PTP Relay Instance or PTP End Instance port.
- 37 j) The Acceptable TimeTransmitter Port Parameter Data Set (acceptableTimeTransmitterPortDS)
 represents the capability to enable/disable the acceptable timeTransmitter table feature on a PTP
 Port.
- 40 k) The External Port Configuration Port Parameter Data Set (externalPortConfigurationPortDS) is used
 with the external port configuration option to indicate the desired state of a PTP Port.

- 1 l) The Asymmetry Measurement Mode Parameter Data Set (*asymmetryMeasurementModeDS*)
 2 represents the capability to enable/disable the Asymmetry Compensation Measurement Procedure
 3 on a port (see Annex G) and is used instead of the *cmldsAsymmetryMeasurementModeDS* when
 4 CMLDS is not used and there is a single gPTP domain.
- 5 m) The Common Services Port Parameter Data Set (*commonServicesPortDS*) enables a PTP Port of a
 6 PTP Instance to determine which port of the respective common service corresponds to that
 7 PTP Port.
- 8 n) The Common Mean Link Delay Service Default Parameter Data Set (*cmldsDefaultDS*) describes
 9 the per-time-aware-system attributes of the Common Mean Link Delay Service.
- 10 o) The Common Mean Link Delay Service Link Port Parameter Data Set (*cmldsLinkPortDS*)
 11 represents time-aware Link Port capabilities for the Common Mean Link Delay Service of a time-
 12 aware system.
- 13 p) The Common Mean Link Delay Service Link Port Parameter Statistics Data Set
 14 (*cmldsLinkPortStatisticsDS*) represents statistics and counters associated with Link Port capabilities
 15 at a given time-aware system.
- 16 q) The Common Mean Link Delay Service Asymmetry Measurement Mode Parameter Data Set
 17 (*cmldsAsymmetryMeasurementModeDS*) represents the capability to enable/disable the
 18 Asymmetry Compensation Measurement Procedure on a Link Port (see Annex G).

19 Table 15-1 shows the structure of the MIB and the relationship of the MIB objects to the above data sets.

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference

MIB table	MIB object	Reference
ieee8021AsV3DefaultDS		defaultDS table (Table 14-1)
	ieee8021AsV3DefaultDSClockIdentity	14.2.2
	ieee8021AsV3DefaultDSNumberPorts	14.2.3
	ieee8021AsV3DefaultDSClockQualityClockClass	14.2.4.2
	ieee8021AsV3DefaultDSClockQualityClockAccuracy	14.2.4.3
	ieee8021AsV3DefaultDSClockQualityOffsetScaledLogVariance	14.2.4.4
	ieee8021AsV3DefaultDSPriority1	14.2.5
	ieee8021AsV3DefaultDSPriority2	14.2.6
	ieee8021AsV3DefaultDSGmCapable	14.2.7
	ieee8021AsV3DefaultDSCurrentUtcOffset	14.2.8
	ieee8021AsV3DefaultDSCurrentUtcOffsetValid	14.2.9
	ieee8021AsV3DefaultDSLeap59	14.2.10
	ieee8021AsV3DefaultDSLeap61	14.2.11
	ieee8021AsV3DefaultDSTimeTraceable	14.2.12
	ieee8021AsV3DefaultDSFrequencyTraceable	14.2.13
	ieee8021AsV3DefaultDSPtpTimescale	14.2.14
	ieee8021AsV3DefaultDSTimeSource	14.2.15
	ieee8021AsV3DefaultDSDomainNumber	14.2.16

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
	ieee8021AsV3DefaultDSSdId	14.2.17
	ieee8021AsV3DefaultDSExternalPortConfigurationEnabled	14.2.18
	ieee8021AsV3DefaultDSInstanceEnable	14.2.19
ieee8021AsV3CurrentDS		currentDS table (Table 14-2)
	ieee8021AsV3CurrentDSStepsRemoved	14.3.2
	ieee8021AsV3CurrentDSOffsetFromTimeTransmitter	14.3.3
	ieee8021AsV3CurrentDSLastGmPhaseChange	14.3.4
	ieee8021AsV3CurrentDSLastGmFreqChange	14.3.5
	ieee8021AsV3CurrentDSGmTimebaseIndicator	14.3.6
	ieee8021AsV3CurrentDSGmChangeCount	14.3.7
	ieee8021AsV3CurrentDSTimeOfLastGmChangeEvent	14.3.8
	ieee8021AsV3CurrentDSTimeOfLastGmPhaseChangeEvent	14.3.9
	ieee8021AsV3CurrentDSTimeOfLastGmFreqChangeEvent	14.3.10
ieee8021AsV3ParentDS		parentDS table (Table 14-3)
	ieee8021AsV3ParentDSParentClockIdentity	14.4.2
	ieee8021AsV3ParentDSParentPortNumber	14.4.2
	ieee8021AsV3ParentDSCumulativeRateRatio	14.4.3
	ieee8021AsV3ParentDSGrandmasterIdentity	14.4.4
	ieee8021AsV3ParentDSGrandmasterClockQualityclockClass	14.4.5.2
	ieee8021AsV3ParentDSGrandmasterClockQualityclockAccuracy	14.4.5.3
	ieee8021AsV3ParentDSGrandmasterClockQualityoffsetScaledLogVar	14.4.5.4
	ieee8021AsV3ParentDSGrandmasterPriority1	14.4.6
	ieee8021AsV3ParentDSGrandmasterPriority2	14.4.7
ieee8021AsV3TimePropertiesDS		timePropertiesDS table (Table 14-4)
	ieee8021AsV3TimePropertiesDSCurrentUtcOffset	14.5.2
	ieee8021AsV3TimePropertiesDSCurrentUtcOffsetValid	14.5.3
	ieee8021AsV3TimePropertiesDSLeap59	14.5.4
	ieee8021AsV3TimePropertiesDSLeap61	14.5.5
	ieee8021AsV3TimePropertiesDSTimeTraceable	14.5.6
	ieee8021AsV3TimePropertiesDSFrequencyTraceable	14.5.7
	ieee8021AsV3TimePropertiesDSPtpTimescale	14.5.8
	ieee8021AsV3TimePropertiesDSTimeSource	14.5.9

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
ieee8021AsV3PathTraceDS		pathTraceDS table (Table 14-5)
	ieee8021AsV3PathTraceDSEnable	14.6.3
ieee8021AsV3PathTraceDSArray		pathTraceDS table (Table 14-5)
	ieee8021AsV3PathTraceDSArrayList	14.6.2
ieee8021AsV3AcceptableTimeTransmitterTableDS		acceptableTimeTransmitterTable DS table (Table 14-6)
	ieee8021AsV3AcceptableTimeTransmitterTableDSMaxTableSize	14.7.2
	ieee8021AsV3AcceptableTimeTransmitterTableDSActualTableSize	14.7.3
ieee8021AsV3AcceptableTimeTransmitterTableDSArray		acceptableTimeTransmitterTable DS table (Table 14-6)
	ieee8021AsV3AcceptableTimeTransmitterTableDSArrayPortIdentit y	14.7.4
	ieee8021AsV3AcceptableTTTableDSArrayAlternatePriority1	14.7.4
ieee8021AsV3PortDS		portDS table (Table 14-10)
	ieee8021AsV3PortDSClockIdentity	14.8.2
	ieee8021AsV3PortDSPortNumber	14.8.2
	ieee8021AsV3PortDSPortState	14.8.3
	ieee8021AsV3PortDSPtpPortEnabled	14.8.4
	ieee8021AsV3PortDSdelayMechanism	14.8.5
	ieee8021AsV3PortDSIsMeasuringDelay	14.8.6
	ieee8021AsV3PortDSAsCapable	14.8.7
	ieee8021AsV3PortDSMeanLinkDelay	14.8.8
	ieee8021AsV3PortDSMeanLinkDelayThresh	14.8.9
	ieee8021AsV3PortDSDelayAsym	14.8.10
	ieee8021AsV3PortDSNbrRateRatio	14.8.11
	ieee8021AsV3PortDSInitialLogAnnounceInterval	14.8.12
	ieee8021AsV3PortDSCurrentLogAnnounceInterval	14.8.13
	ieee8021AsV3PortDSUseMgtSettableLogAnnounceInterval	14.8.14
	ieee8021AsV3PortDSMgtSettableLogAnnounceInterval	14.8.15
	ieee8021AsV3PortDSAnnounceReceiptTimeout	14.8.16
	ieee8021AsV3PortDSInitialLogSyncInterval	14.8.17
	ieee8021AsV3PortDSCurrentLogSyncInterval	14.8.18
	ieee8021AsV3PortDSUseMgtSettableLogSyncInterval	14.8.19
	ieee8021AsV3PortDSMgtSettableLogSyncInterval	14.8.20

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
	ieee8021AsV3PortDSSyncReceiptTimeout	14.8.21
	ieee8021AsV3PortDSSyncReceiptTimeoutTimeInterval	14.8.22
	ieee8021AsV3PortDSInitialLogPdelayReqInterval	14.8.23
	ieee8021AsV3PortDSCurrentLogPdelayReqInterval	14.8.24
	ieee8021AsV3PortDSUseMgtSettableLogPdelayReqInterval	14.8.25
	ieee8021AsV3PortDSMgtSettableLogPdelayReqInterval	14.8.26
	ieee8021AsV3PortDSInitialLogGptpCapableMessageInterval	14.8.27
	ieee8021AsV3PortDSCurrentLogGptpCapableMessageInterval	14.8.28
	ieee8021AsV3PortDSUseMgtSettableLogGptpCapableMessageInterval	14.8.29
	ieee8021AsV3PortDSMgtSettableLogGptpCapableMessageInterval	14.8.30
	ieee8021AsV3PortDSInitialComputeNbrRateRatio	14.8.31
	ieee8021AsV3PortDSCurrentComputeNbrRateRatio	14.8.32
	ieee8021AsV3PortDSUseMgtSettableComputeNbrRateRatio	14.8.33
	ieee8021AsV3PortDSMgtSettableComputeNbrRateRatio	14.8.34
	ieee8021AsV3PortDSInitialComputeMeanLinkDelay	14.8.35
	ieee8021AsV3PortDSCurrentComputeMeanLinkDelay	14.8.36
	ieee8021AsV3PortDSUseMgtSettableComputeMeanLinkDelay	14.8.37
	ieee8021AsV3PortDSMgtSettableComputeMeanLinkDelay	14.8.38
	ieee8021AsV3PortDSAllowedLostRsp	14.8.39
	ieee8021AsV3PortDSAllowedFaults	14.8.40
	ieee8021AsV3PortDSGPtpCapableReceiptTimeout	14.8.41
	ieee8021AsV3PortDSVersionNumber	14.8.42
	ieee8021AsV3PortDSNup	14.8.43
	ieee8021AsV3PortDSNdown	14.8.44
	ieee8021AsV3PortDSOneStepTxOper	14.8.45
	ieee8021AsV3PortDSOneStepReceive	14.8.46
	ieee8021AsV3PortDSOneStepTransmit	14.8.47
	ieee8021AsV3PortDSInitialOneStepTxOper	14.8.48
	ieee8021AsV3PortDSCurrentOneStepTxOper	14.8.49
	ieee8021AsV3PortDSUseMgtSettableOneStepTxOper	14.8.50
	ieee8021AsV3PortDSMgtSettableOneStepTxOper	14.8.51
	ieee8021AsV3PortDSSyncLocked	14.8.52
	ieee8021AsV3PortDSPdelayTruncTST1	14.8.53

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
	ieee8021AsV3PortDSPdelayTruncTST2	14.8.53
	ieee8021AsV3PortDSPdelayTruncTST3	14.8.53
	ieee8021AsV3PortDSPdelayTruncTST4	14.8.53
	ieee8021AsV3PortDSMinorVersionNumber	14.8.54
ieee8021AsV3DescriptionPortDS		descriptionPortDS table (Table 14-11)
	ieee8021AsV3DescriptionPortDSProfileIdentifier	14.9.2
ieee8021AsV3PortStatDS		portStatisticsDS table (Table 14-12)
	ieee8021AsV3PortStatRxSyncCount	14.10.2
	ieee8021AsV3PortStatRxOneStepSyncCount	14.10.3
	ieee8021AsV3PortStatRxFollowUpCount	14.10.4
	ieee8021AsV3PortStatRxPdelayRequestCount	14.10.5
	ieee8021AsV3PortStatRxPdelayRspCount	14.10.6
	ieee8021AsV3PortStatRxPdelayRspFollowUpCount	14.10.7
	ieee8021AsV3PortStatRxAnnounceCount	14.10.8
	ieee8021AsV3PortStatRxPtpPacketDiscardCount	14.10.9
	ieee8021AsV3PortStatSyncReceiptTimeoutCount	14.10.10
	ieee8021AsV3PortStatAnnounceReceiptTimeoutCount	14.10.11
	ieee8021AsV3PortStatPdelayAllowedLostRspExceededCount	14.10.12
	ieee8021AsV3PortStatTxSyncCount	14.10.13
	ieee8021AsV3PortStatTxOneStepSyncCount	14.10.14
	ieee8021AsV3PortStatTxFollowUpCount	14.10.15
	ieee8021AsV3PortStatTxPdelayRequestCount	14.10.16
	ieee8021AsV3PortStatTxPdelayRspCount	14.10.17
	ieee8021AsV3PortStatTxPdelayRspFollowUpCount	14.10.18
	ieee8021AsV3PortStatTxAnnounceCount	14.10.19
ieee8021AsV3AcceptableTimeTransmitterPortDS		acceptableTimeTransmitterTable DS table (Table 14-13)
	ieee8021AsV3AcceptableTTPortDSAcceptableTTTableEnabled	14.11.2
ieee8021AsV3ExternalPortConfigurationPortDS		externalPortConfigurationPortDS table (Table 14-14)
	ieee8021AsV3ExternalPortConfigurationPortDSDesiredState	14.12.2

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
	ieee8021AsV3AsymMeasurementModeDS	asymmetryMeasurementModeDS table (Table 14-15)
	ieee8021AsV3AsymMeasurementModeDSAsymMeasurementMode	14.13.2
	ieee8021AsV3CommonServicesPortDS	commonServicesPortDS table (Table 14-16)
	ieee8021AsV3CommonServicesPortDSCmlsLinkPortPortNumber	14.14.2
	ieee8021AsV3CommonMeanLinkDelayServiceDefaultDS	cmlsDefaultDS table (Table 14-17)
	ieee8021AsV3CmlsDefaultDSClockIdentity	14.15.2
	ieee8021AsV3CmlsDefaultDSNumberLinkPorts	14.15.3
	ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDS	cmlsLinkPortDS table (Table 14-18)
	ieee8021AsV3CmlsLinkPortDSClockIdentity	14.16.2
	ieee8021AsV3CmlsLinkPortDSPortNumber	14.16.2
	ieee8021AsV3CmlsLinkPortDSCmlsLinkPortEnabled	14.16.3
	ieee8021AsV3CmlsLinkPortDSIsMeasuringDelay	14.16.4
	ieee8021AsV3CmlsLinkPortDSAsCapableAcrossDomains	14.16.5
	ieee8021AsV3CmlsLinkPortDSMeanLinkDelay	14.16.6
	ieee8021AsV3CmlsLinkPortDSMeanLinkDelayThresh	14.16.7
	ieee8021AsV3CmlsLinkPortDSDelayAsym	14.16.8
	ieee8021AsV3CmlsLinkPortDSNbrRateRatio	14.16.9
	ieee8021AsV3CmlsLinkPortDSInitialLogPdelayReqInterval	14.16.10
	ieee8021AsV3CmlsLinkPortDSCurrentLogPdelayReqInterval	14.16.11
	ieee8021AsV3CmlsLinkPortDSUseMgtSettableLogPdelayReqInterv	14.16.12
	ieee8021AsV3CmlsLinkPortDSMgtSettableLogPdelayReqInterval	14.16.13
	ieee8021AsV3CmlsLinkPortDSInitialComputeNbrRateRatio	14.16.14
	ieee8021AsV3CmlsLinkPortDSCurrentComputeNbrRateRatio	14.16.15
	ieee8021AsV3CmlsLinkPortDSUseMgtSettableComputeNbrRateRatio	14.16.16
	ieee8021AsV3CmlsLinkPortDSMgtSettableComputeNbrRateRatio	14.16.17
	ieee8021AsV3CmlsLinkPortDSInitialComputeMeanLinkDelay	14.16.18
	ieee8021AsV3CmlsLinkPortDSCurrentComputeMeanLinkDelay	14.16.19
	ieee8021AsV3CmlsLinkPortDSUseMgtSettableComputeMeanLinkDelay	14.16.20
	ieee8021AsV3CmlsLinkPortDSMgtSettableComputeMeanLinkDelay	14.16.21

Table 15-1—IEEE8021-AS-V3 MIB structure and object cross reference (continued)

MIB table	MIB object	Reference
	ieee8021AsV3CmldsLinkPortDSAllowedLostRsp	14.16.22
	ieee8021AsV3CmldsLinkPortDSAllowedFaults	14.16.23
	ieee8021AsV3CmldsLinkPortDSVersionNumber	14.16.24
	ieee8021AsV3CmldsLinkPortDSPdelayTruncTST1	14.16.25
	ieee8021AsV3CmldsLinkPortDSPdelayTruncTST2	14.16.25
	ieee8021AsV3CmldsLinkPortDSPdelayTruncTST3	14.16.25
	ieee8021AsV3CmldsLinkPortDSPdelayTruncTST4	14.16.25
	ieee8021AsV3CmldsLinkPortDSMinorVersionNumber	14.16.26
ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDS	cmldsLinkPortStatisticsDS table (Table 14-19)	
	ieee8021AsV3CmldsLinkPortStatDSRxPdelayRequestCount	14.17.2
	ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspCount	14.17.3
	ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspFollowUpCount	14.17.4
	ieee8021AsV3CmldsLinkPortStatDSRxPtpPacketDiscardCount	14.17.5
	ieee8021AsV3CmldsLinkPortStatDSPdelayAllowedLostRspExceededCount	14.17.6
	ieee8021AsV3CmldsLinkPortStatDSTxPdelayRequestCount	14.17.7
	ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspCount	14.17.8
	ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspFollowUpCount	14.17.9
ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDS	cmldsAsymmetryMeasurementModeDS table (Table 14-20)	
	ieee8021AsV3CmldsAsymMeasurementModeDSAAsymMeasurementMode	14.18.2

¹ 15.3 Relationship to MIB in IEEE Std 802.1AS-2011

² The version 1 MIB module (IEEE8021-AS MIB) that was published in IEEE Std 802.1AS-2011 has been
³ superseded by the version 2 MIB module (IEEE8021-AS-V2 MIB) specified in 15.6 of IEEE Std
⁴ 802.1AS-2020. The version 3 MIB module (IEEE8021-AS-V3 MIB) specified in the current standard
⁵ remains unchanged from the IEEE8021-AS-V2 MIB except updating terminology. Support of the version 3
⁶ module is a requirement for conformance to the required or optional capabilities (Clause 5) in the current
⁷ standard.

⁸ For an implementation that supports a single PTP Instance, version 1, version 2, and version 3
⁹ implementations can successfully co-exist and interoperate.

1 15.4 Security considerations

2 SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure
3 (for example by using IPsec), there is no control as to who on the secure network is allowed to access and
4 GET/SET (read/change/create/delete) the objects in these MIB module.

5 It is recommended that implementers consider the security features as provided by the SNMPv3 framework
6 [see section 8 in IETF RFC 3410 (Dec. 2002)], including full support for the SNMPv3 cryptographic
7 mechanisms (for authentication and privacy).

8 Further, deployment of SNMP versions prior to SNMPv3 is not recommended. Instead, it is recommended
9 to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to
10 ensure that the SNMP entity giving access to an instance of these MIB modules is properly configured to
11 give access to the objects only to the principals (users) that have legitimate rights to indeed GET or SET
12 (change/create/delete) them.

13 A number of management objects defined in the IEEE8021-AS-V3 MIB module have a MAX-ACCESS
14 clause of read-write and/or read-create. Such objects might be considered sensitive or vulnerable in some
15 network environments. The support for SET operations in a non-secure environment without proper
16 protection can have a negative effect on network operations.

17 Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than
18 “not-accessible”) might be considered sensitive or vulnerable in some network environments. It is thus
19 important to control all types of access (including GET and/or NOTIFY) to these objects and possibly to
20 even encrypt the values of these objects when sending them over the network via SNMP.

21 The following objects in the IEEE8021-AS-V2 MIB can be manipulated to interfere with the operation of
22 timing synchronization. This could, for example, be used to force a reinitialization of state machines to
23 cause timing synchronization and network instability. Another possibility would be for an attacker to
24 override Grandmaster PTP Instance status to give a user (or an attacker) unauthorized control over the
25 network time.

26 Improper manipulation of the following writable objects could result in an unintended Grandmaster PTP
27 Instance to be elected when a system is grandmaster-capable in a gPTP domain. It could also be used
28 maliciously to cause frequent Grandmaster PTP Instance changes that could affect network stability.

29 ieee8021AsV3DefaultDSPriority1
30 ieee8021AsV3DefaultDSPriority2

31 Improper manipulation of the following writable objects could result in a segmented time-aware network,
32 could compromise the expected accuracy, and could interrupt paths of the gPTP domain.

33 ieee8021AsV3PortDSPtpPortEnabled
34 ieee8021AsV3PortDSDelayAsymmetry

35 Unintended access to any of the readable tables or variables in the IEEE8021-AS-V3 MIB alerts the reader
36 that timing synchronization in gPTP domain is configured, and on which values timing parameters are
37 configured, and which system is current Grandmaster PTP Instance. This information can suggest to an
38 attacker what applications are being run, and thus suggest application-specific attacks, or can enable the
39 attacker to detect whether their attacks are being successful. It is thus important to control even GET access
40 to these objects and possibly to even encrypt the values of these objects when sending them over the network
41 via SNMP.

1 15.5 Textual conventions defined in this MIB

2 The following textual conventions are defined in this MIB:

- 3 a) Ieee8021AsV3ClockIdentity. IEEE 802 MAC address represented in “canonical” order defined by
4 IEEE Std 802-2014, 64-bit Network Unique Identifier (NUI-64) as described in IEEE Std
5 802c-2017.
- 6 b) Ieee8021AsV3GPtpProfileIdentifier. Profile identifier (see 14.9.2).
- 7 c) Ieee8021AsV3ClockClassValue. Clock class value (see 8.6.2.2).
- 8 d) Ieee8021AsV3ClockAccuracyValue. Clock accuracy value (see 8.6.2.3).
- 9 e) Ieee8021AsV3TimeSourceValue. Source of time used by Grandmaster PTP Instance (see 8.6.2.7).
- 10 f) Ieee8021ASV3PtpTimeInterval. Time intervals in units of 2^{-16} ns (see 6.4.3.3).
- 11 g) Ieee8021ASV3PtpPortIdentity. Identifies a port of a PTP Instance (see 6.4.3.7).
- 12 h) Ieee8021ASV3ScaledNs. Represents signed values of time and time interval in units of 2^{-16} ns
13 (see 6.4.3.1).
- 14 i) Ieee8021ASV3UScaledNs. Represents unsigned values of time and time interval in units of 2^{-16} ns
15 (see 6.4.3.2).
- 16 j) Ieee8021ASV3PTPInstanceIdentifier. Entity of a single time-aware system that executes gPTP in
17 one gPTP domain (see 7.2.1 and 8.1).
- 18 k) Ieee8021ASV3Timestamp. Value of Ieee8021ASV2Timestamp is equal to the remainder obtained
19 upon dividing the respective timestamp, expressed in units of 2^{-16} ns, by 2^{48} (see 14.8.53).

20 15.6 IEEE 802.1AS MIB module^{19,20}

21 In the following MIB modules definitions, if any discrepancy between the DESCRIPTION text and the
22 corresponding definition in any other part of this standard occurs, the definitions outside this subclause take
23 precedence.

¹⁹ Copyright release for MIBs: Users of this standard may freely reproduce the MIBs contained in this subclause so that they can be used for their intended purpose.

²⁰ An ASCII version of this MIB module can be obtained from the IEEE 802.1 website at <https://www.ieee802.org/1/pages/MIBS.html>.

```
1 IEEE8021-AS-V3-MIB DEFINITIONS ::= BEGIN
2 -- =====
3 -- MIB for support of 802.1AS Timing and Synchronization in
4 -- IEEE 802.1Q Bridged Local Area Networks
5 -- =====
6
7 IMPORTS
8   MODULE-IDENTITY, OBJECT-TYPE, Unsigned32, Integer32, Counter32
9     FROM SNMPv2-SMI          -- [RFC2578]
10   TEXTUAL-CONVENTION, TruthValue, RowStatus, TimeStamp
11   FROM SNMPv2-TC           -- [RFC2579]
12   MODULE-COMPLIANCE, OBJECT-GROUP -- [RFC2580]
13   FROM SNMPv2-CONF
14     SnmpAdminString
15   FROM SNMP-FRAMEWORK-MIB    -- [RFC3411]
16   InterfaceIndexOrZero
17   FROM IF-MIB                -- [RFC2863]
18   Float64TC
19   FROM FLOAT-TC-MIB          -- [RFC6340]
20   IEEE8021BridgePortNumber
21   FROM IEEE8021-TC-MIB
22   ;
23
24 ieee8021AsV3TimeSyncMib MODULE-IDENTITY
25   LAST-UPDATED "202402140000Z" -- February 14, 2024
26   ORGANIZATION "IEEE 802.1 Working Group"
27   CONTACT-INFO
```

1 "WG-URL: <http://ieee802.org/1/>

2 WG-EMail: stds-802-1-l@ieee.org

3

4 Contact: IEEE 802.1 Working Group Chair

5 Postal: C/O IEEE 802.1 Working Group

6 IEEE Standards Association

7 445 Hoes Lane

8 Piscataway, NJ 08854

9 USA

10

11 E-mail: stds-802-1-chairs@ieee.org"

12

13 DESCRIPTION

14 "The Management Information Base module for

15 IEEE 802.1AS time-synchronization protocol."

16

17 REVISION "202402140000Z" -- February 14, 2024

18 DESCRIPTION

19 "This MIB module remains unchanged from

20 IEEE8021-AS-V2-MIB except updating the terminology.

21

22 Published as part of IEEE Std 802.1ASdr-2024

23 Copyright (C) IEEE (2024).

24 This version of this MIB module is part of IEEE Std

25 802.1ASdr-2024; see the standard itself for full legal

26 notices."

27

```
1 ::= { iso(1) org(3) ieee(111)
2     standards-association-numbers-series-standards (2)
3     lan-man-stds (802) ieee802dot1 (1) ieee802dot1mibs (1) 40 }
4
5 ieee8021AsV3MIBObjects OBJECT IDENTIFIER ::= {ieee8021AsV3TimeSyncMib 1}
6 ieee8021AsV3Conformance OBJECT IDENTIFIER ::= {ieee8021AsV3TimeSyncMib 2}
7
8 -- =====
9 -- Textual Conventions
10 -- =====
11
12 Ieee8021AsV3ClockIdentity ::= TEXTUAL-CONVENTION
13 DISPLAY-HINT
14 "1x:"
15 STATUS current
16 DESCRIPTION
17 "The Ieee8021AsV3ClockIdentity type identifies a PTP Instance.
18 The clockIdentity attribute shall be as specified in
19 IEEE Std 1588-2019."
20 REFERENCE "6.4.3.6, 8.5.2.2 and IEEE Std 1588-2019 7.5.2.2"
21 SYNTAX OCTET STRING (SIZE (8))
22
23 Ieee8021AsV3GPtpProfileIdentifier ::= TEXTUAL-CONVENTION
24 DISPLAY-HINT
25 "1x:"
26 STATUS current
27 DESCRIPTION
```

1 "The Ieee8021AsV3GPtpprofileIdentifier attribute is the
2 profileIdentifier for this PTP profile."

3 REFERENCE "14.9.2, F.2 "

4 SYNTAX OCTET STRING (SIZE (6))

5

6 Ieee8021AsV3ClockClassValue ::= TEXTUAL-CONVENTION

7 STATUS current

8 DESCRIPTION

9 "The Ieee8021AsV3ClockClassValue attribute denotes the traceability
10 of the synchronized time distributed by a ClockTimeTransmitter when it is
11 the Grandmaster PTP Instance.

12 A more detailed description of clockClass can be found in
13 IEEE Std 1588-2019."

14 REFERENCE "8.6.2.2 and IEEE Std 1588-2019 7.6.2.5"

15 SYNTAX INTEGER {

16 primarySync(6),

17 primarySyncLost(7),

18 applicationSpecificSync(13),

19 applicationSpecificSyncLost(14),

20 primarySyncAlternativeA(52),

21 applicationSpecificAlternativeA(58),

22 primarySyncAlternativeB(187),

23 applicationSpecificAlternativeB(193),

24 defaultClock(248),

25 timeReceiverOnlyClock(255)

26 }

27

1 Ieee8021AsV3ClockAccuracyValue ::= TEXTUAL-CONVENTION

2 STATUS current

3 DESCRIPTION

4 "The Ieee8021AsV3ClockAccuracyValue attribute indicates the
5 expected time accuracy of a ClockTimeTransmitter.

6 A more detailed description of clockAccuracy can be found in
7 IEEE Std 1588-2019."

8 REFERENCE "8.6.2.3 and IEEE Std 1588-2019 7.6.2.6"

9 SYNTAX INTEGER {

10 timeAccurateTo25ns(32),
11 timeAccurateTo100ns(33),
12 timeAccurateTo250ns(34),
13 timeAccurateTo1us(35),
14 timeAccurateTo2dot5us(36),
15 timeAccurateTo10us(37),
16 timeAccurateTo25us(38),
17 timeAccurateTo100us(39),
18 timeAccurateTo250us(40),
19 timeAccurateTo1ms(41),
20 timeAccurateTo2dot5ms(42),
21 timeAccurateTo10ms(43),
22 timeAccurateTo25ms(44),
23 timeAccurateTo100ms(45),
24 timeAccurateTo250ms(46),
25 timeAccurateTo1s(47),
26 timeAccurateTo10s(48),
27 timeAccurateToGT10s(49),

1 timeAccurateToUnknown(254)
2 }
3
4 Ieee8021AsV3TimeSourceValue ::= TEXTUAL-CONVENTION
5 STATUS current
6 DESCRIPTION
7 "The Ieee8021AsV3TimeSourceValue is an information only
8 attribute indicating the type of source of time used by a
9 ClockTimeTransmitter. The value is not used in the selection of the
10 Grandmaster PTP Instance. The values of TimeSource are
11 given below and are specified in Table 8-2. These represent
12 categories. For example, the GPS entry includes not only the
13 GPS system of the U.S. Department of Defense but the European
14 Galileo system and other present and future GNSSs.
15
16 In the absence of a default value set by a user of this standard,
17 the default value of timeSource shall be INTERNAL_OSCILLATOR.
18
19 A more detailed description of timeSource can be found in
20 IEEE Std 1588-2019.
21
22 The following interpretation is placed on the value:
23 0x10: Atomic Clock,
24 0x20: GPS,
25 0x30: Terrestrial Radio,
26 0x40: PTP,
27 0x50: NTP,

1 0x60: Hand Set,
2 0x90: Other,
3 0xA0: Internal Oscillator "
4 REFERENCE "8.6.2.7, Table 8-2 and IEEE Std 1588-2019 7.6.2.8"
5 SYNTAX INTEGER {
6 atomicClock(16),
7 gps(32),
8 terrestrialRadio(48),
9 ptp(64),
10 ntp(80),
11 handSet(96),
12 other(144),
13 internalOscillator(160)
14 }
15
16 Ieee8021ASV3PtpTimeInterval ::= TEXTUAL-CONVENTION
17 STATUS current
18 DESCRIPTION
19 "The Ieee8021ASV3PtpTimeInterval type represents time intervals
20 in units of 2^-16 ns. Positive or negative time
21 intervals outside the maximum range of this data type shall
22 be encoded as the largest positive and negative values of
23 the data type respectively.
24 For example: 2.5 ns is expressed as:
25 (hex) 0x0000 0000 0002 8000"
26 REFERENCE "6.4.3.3"
27 SYNTAX OCTET STRING (SIZE (8))

1

2 Ieee8021ASV3PtpPortIdentity ::= TEXTUAL-CONVENTION

3 STATUS current

4 DESCRIPTION

5 "The Ieee8021ASV3PtpPortIdentity type identifies a port of a

6 PTP Instance.

7 The first 8 octets within this value specifies the

8 ClockIdentity.

9 The last 2 octets within this value specifies the port number."

10 REFERENCE "6.4.3.7"

11 SYNTAX OCTET STRING (SIZE (10))

12

13 Ieee8021ASV3ScaledNs ::= TEXTUAL-CONVENTION

14 STATUS current

15 DESCRIPTION

16 "The Ieee8021ASV3ScaledNs type represents signed values of

17 time and time interval in units of 2^{-16} ns.

18 Positive or negative values of time or time interval outside the
19 maximum range of this data type are encoded as the largest
20 positive or negative value of the data type, respectively.

21 For example: -2.5 ns is expressed as:

22 (hex) 0xFFFF FFFF FFFF FFFF FFFD 8000"

23 REFERENCE "6.4.3.1"

24 SYNTAX OCTET STRING (SIZE (12))

25

26 Ieee8021ASV3UScaledNs ::= TEXTUAL-CONVENTION

27 STATUS current

1 DESCRIPTION

2 "The Ieee8021ASV3UScaledNs type represents unsigned values of
3 time and time interval in units of 2^{16} ns.
4 Positive or negative values of time or time interval outside
5 the maximum range of this data type are encoded as the largest
6 positive or negative value of the data type, respectively.

7 For example: 2.5 ns is expressed as:

8 (hex) 0x0000 0000 0000 0000 0002 8000"

9 REFERENCE "6.4.3.2"

10 SYNTAX OCTET STRING (SIZE (12))

11

12 Ieee8021ASV3PTPInstanceIdentifier ::= TEXTUAL-CONVENTION

13 DISPLAY-HINT "d"

14 STATUS current

15 DESCRIPTION

16 "The entity of a single time-aware system that executes gPTP in
17 one gPTP domain is called a PTP Instance. A time-aware system
18 can contain multiple PTP Instances, which are each associated
19 with a different gPTP domain. There are two types of
20 PTP Instances, a PTP End Instance and a PTP Relay Instance."

21 REFERENCE "7.2.1"

22 SYNTAX Unsigned32

23

24 Ieee8021ASV3Timestamp ::= TEXTUAL-CONVENTION

25 STATUS current

26 DESCRIPTION

27 "The value of Ieee8021ASV3Timestamp is equal to the remainder

1 obtained upon dividing the respective timestamp, expressed
2 in units of 2^{16} ns, by 2^{48} ."

3 REFERENCE "14.8.53, 14.16.25 and Table 14-9"

4 SYNTAX OCTET STRING (SIZE (6))

5

6 -- =====

7 -- subtrees in the IEEE8021-AS-MIB

8 --

9 -- System Time-Aware Parameters/Capabilities for each instance of

10 -- gPTP domain. ieee8021AsV3InstanceListIndex that is of

11 -- ieee8021AsV3DomainIdentificationNumber object-type is used as Index.

12 --

13 -- =====

14

15 -- =====

16 -- The PTP Instance set is used to allow for dynamic creation and

17 -- deletion of PTP Instances and logical ports implementations that

18 -- support dynamic create/delete of devices.

19 -- =====

20

21 ieee8021AsV3PtpInstanceTable OBJECT-TYPE

22 SYNTAX SEQUENCE OF Ieee8021AsV3PtpInstanceEntry

23 MAX-ACCESS not-accessible

24 STATUS current

25 DESCRIPTION

26 "This table is used to allow for dynamic creation and deletion

27 of PTP Instances and logical ports implementations that support

1 dynamic create/delete of devices."

2 REFERENCE "14.1"

3 ::= { ieee8021AsV3MIBObjects 1 }

4

5 ieee8021AsV3PtpInstanceEntry OBJECT-TYPE

6 SYNTAX Ieee8021AsV3PtpInstanceEntry

7 MAX-ACCESS not-accessible

8 STATUS current

9 DESCRIPTION

10 "An entry that specifies a PTP Instance."

11 INDEX { ieee8021AsV3PtpInstance }

12 ::= { ieee8021AsV3PtpInstanceTable 1 }

13

14 Ieee8021AsV3PtpInstanceEntry ::=

15 SEQUENCE {

16 ieee8021AsV3PtpInstance Ieee8021ASV3PTPInstanceIdentifier,

17 ieee8021AsV3PtpInstanceName SnmpAdminString,

18 ieee8021AsV3PtpInstanceRowStatus RowStatus

19 }

20

21 ieee8021AsV3PtpInstance OBJECT-TYPE

22 SYNTAX Ieee8021ASV3PTPInstanceIdentifier

23 MAX-ACCESS not-accessible

24 STATUS current

25 DESCRIPTION

26 "The entity of a single time-aware system that executes gPTP in

27 one gPTP domain is called a PTP Instance. A time-aware system can

1 contain multiple PTP Instances, which are each associated with
2 a different gPTP domain. There are two types of PTP Instances,
3 a PTP End Instance and a PTP Relay Instance."

4 REFERENCE "7.2.1"

5 ::= { ieee8021AsV3PtpInstanceEntry 1 }

6

7 ieee8021AsV3PtpInstanceName OBJECT-TYPE

8 SYNTAX SnmpAdminString

9 MAX-ACCESS read-create

10 STATUS current

11 DESCRIPTION

12 "Name for identification of a PTP Instance."

13 DEFVAL { "" }

14 ::= { ieee8021AsV3PtpInstanceEntry 2 }

15

16 ieee8021AsV3PtpInstanceRowStatus OBJECT-TYPE

17 SYNTAX RowStatus

18 MAX-ACCESS read-create

19 STATUS current

20 DESCRIPTION

21 "This attribute is used to create and delete PTP Instances."

22 REFERENCE "14.1"

23 ::= { ieee8021AsV3PtpInstanceEntry 3 }

24

25 -- =====

26 -- The Default data set represents native time capability of a time-

27 -- aware system and is consistent with respective IEEE 1588 data set.

1 -- =====
2
3 ieee8021AsV3DefaultDSTable OBJECT-TYPE
4 SYNTAX SEQUENCE OF Ieee8021AsV3DefaultDSEntry
5 MAX-ACCESS not-accessible
6 STATUS current
7 DESCRIPTION
8 "The Default Parameter Data Set represents the native capabilities
9 of a PTP Instance, i.e., a PTP Relay Instance or a
10 PTP End Instance."
11 REFERENCE "14.2"
12 ::= { ieee8021AsV3MIBObjects 2 }
13
14 ieee8021AsV3DefaultDSEntry OBJECT-TYPE
15 SYNTAX Ieee8021AsV3DefaultDSEntry
16 MAX-ACCESS not-accessible
17 STATUS current
18 DESCRIPTION
19 "Default Data Set contains the profile Identifier for
20 this instance of gPTP domain."
21 INDEX { ieee8021AsV3PtpInstance }
22 ::= { ieee8021AsV3DefaultDSTable 1 }
23
24 Ieee8021AsV3DefaultDSEntry ::=
25 SEQUENCE {
26 ieee8021AsV3DefaultDSClockIdentity Ieee8021AsV3ClockIdentity,
27 ieee8021AsV3DefaultDSNumberPorts Unsigned32,

```
1 ieee8021AsV3DefaultDSClockQualityClockClass Ieee8021AsV3ClockClassValue,  
2 ieee8021AsV3DefaultDSClockQualityClockAccuracy Ieee8021AsV3ClockAccuracyValue,  
3 ieee8021AsV3DefaultDSClockQualityOffsetScaledLogVariance Unsigned32,  
4 ieee8021AsV3DefaultDSPriority1 Unsigned32,  
5 ieee8021AsV3DefaultDSPriority2 Unsigned32,  
6 ieee8021AsV3DefaultDSGmCapable TruthValue,  
7 ieee8021AsV3DefaultDSCurrentUtcOffset Integer32,  
8 ieee8021AsV3DefaultDSCurrentUtcOffsetValid TruthValue,  
9 ieee8021AsV3DefaultDSLeap59 TruthValue,  
10 ieee8021AsV3DefaultDSLeap61 TruthValue,  
11 ieee8021AsV3DefaultDSTimeTraceable TruthValue,  
12 ieee8021AsV3DefaultDSFrequencyTraceable TruthValue,  
13 ieee8021AsV3DefaultDSPtpTimescale TruthValue,  
14 ieee8021AsV3DefaultDSTimeSource Ieee8021AsV3TimeSourceValue,  
15 ieee8021AsV3DefaultDSDomainNumber Unsigned32,  
16 ieee8021AsV3DefaultDSSdoId Unsigned32,  
17 ieee8021AsV3DefaultDSExternalPortConfigurationEnabled TruthValue,  
18 ieee8021AsV3DefaultDSInstanceEnable TruthValue  
19 }  
20  
21 ieee8021AsV3DefaultDSClockIdentity OBJECT-TYPE  
22 SYNTAX Ieee8021AsV3ClockIdentity  
23 MAX-ACCESS read-only  
24 STATUS current  
25 DESCRIPTION  
26 "The value is the clockIdentity of the PTP Instance.  
27 The clockIdentity attribute shall be as specified in
```

1 IEEE Std 1588-2019."

2 REFERENCE "14.2.2 and IEEE Std 1588-2019 7.5.2.2"

3 ::= { ieee8021AsV3DefaultDSEntry 1 }

4

5

6 ieee8021AsV3DefaultDSNumberPorts OBJECT-TYPE

7 SYNTAX Unsigned32(1..65535)

8 MAX-ACCESS read-only

9 STATUS current

10 DESCRIPTION

11 "The number of ports of the PTP Instance. For an end

12 station the value is 1."

13 REFERENCE "14.2.3"

14 ::= { ieee8021AsV3DefaultDSEntry 2 }

15

16

17 ieee8021AsV3DefaultDSClockQualityClockClass OBJECT-TYPE

18 SYNTAX Ieee8021AsV3ClockClassValue

19 MAX-ACCESS read-only

20 STATUS current

21 DESCRIPTION

22 "The value is the clockClass of the PTP Instance, which

23 implements the clockClass specifications of 8.6.2.2."

24 REFERENCE "14.2.4.2"

25 ::= { ieee8021AsV3DefaultDSEntry 3 }

26

27

1 ieee8021AsV3DefaultDSClockQualityClockAccuracy OBJECT-TYPE

2 SYNTAX Ieee8021AsV3ClockAccuracyValue

3 MAX-ACCESS read-only

4 STATUS current

5 DESCRIPTION

6 "The value is the clockAccuracy of the PTP Instance, which

7 implements the clockAccuracy specifications of 8.6.2.3."

8 REFERENCE "14.2.4.3"

9 ::= { ieee8021AsV3DefaultDSEntry 4 }

10

11

12 ieee8021AsV3DefaultDSOffsetScaledLogVariance OBJECT-TYPE

13 SYNTAX Unsigned32(0..65535)

14 MAX-ACCESS read-only

15 STATUS current

16 DESCRIPTION

17 "The value is the offsetScaledLogVariance of the PTP Instance,

18 which implements the offsetScaledLogVariance specifications

19 of 8.6.2.4."

20 REFERENCE "14.2.4.4"

21 ::= { ieee8021AsV3DefaultDSEntry 5 }

22

23 ieee8021AsV3DefaultDSPriority1 OBJECT-TYPE

24 SYNTAX Unsigned32(0..255)

25 MAX-ACCESS read-write

26 STATUS current

27 DESCRIPTION

1 "The value is the priority1 attribute of the PTP Instance."
2 REFERENCE "14.2.5"
3 ::= { ieee8021AsV3DefaultDSEntry 6 }
4
5
6 ieee8021AsV3DefaultDSPriority2 OBJECT-TYPE
7 SYNTAX Unsigned32(0..255)
8 MAX-ACCESS read-write
9 STATUS current
10 DESCRIPTION
11 "The value is the priority2 attribute of the PTP Instance."
12 REFERENCE "14.2.6"
13 DEFVAL { 248 }
14 ::= { ieee8021AsV3DefaultDSEntry 7 }
15
16
17 ieee8021AsV3DefaultDSGmCapable OBJECT-TYPE
18 SYNTAX TruthValue
19 MAX-ACCESS read-only
20 STATUS current
21 DESCRIPTION
22 "The value is TRUE (1) if the PTP Instance is capable of being a
23 Grandmaster PTP Instance, and FALSE (2) if the PTP Instance is
24 not capable of being a Grandmaster PTP Instance."
25 REFERENCE "14.2.7"
26 ::= { ieee8021AsV3DefaultDSEntry 8 }
27

1
2 ieee8021AsV3DefaultDSCurrentUtcOffset OBJECT-TYPE

3 SYNTAX Integer32(-32768..32767)

4 UNITS "seconds"

5 MAX-ACCESS read-only

6 STATUS current

7 DESCRIPTION

8 "The value is the offset between TAI and UTC, relative to
9 the ClockTimeTransmitter entity of this PTP Instance. It is equal
10 to the global variable sysCurrentUtcOffset.

11 The value is in units of seconds.

12

13 The default value is selected as follows:

14 a)The value is the value obtained from a primary
15 reference if the value is known at the time of
16 initialization, else

17 b)The value is the current IERS defined value of
18 TAI - UTC (see IERS Bulletin C) when the PTP Instance
19 is designed.currentUtcOffsetValid"

20 REFERENCE "14.2.8"

21 ::= { ieee8021AsV3DefaultDSEntry 9 }

22

23

24 ieee8021AsV3DefaultDSCurrentUtcOffsetValid OBJECT-TYPE

25 SYNTAX TruthValue

26 MAX-ACCESS read-only

27 STATUS current

1 DESCRIPTION

2 "The default value is TRUE (1) if the value of
3 ieee8021AsV3DefaultDSCurrentUtcOffset is known to be
4 correct, otherwise it is set to FALSE (2)."

5 REFERENCE "14.2.9"

6 ::= { ieee8021AsV3DefaultDSEntry 10 }

7

8

9 ieee8021AsV3DefaultDSLeap59 OBJECT-TYPE

10 SYNTAX TruthValue

11 MAX-ACCESS read-only

12 STATUS current

13 DESCRIPTION

14 "A TRUE (1) value indicates that the last minute of the
15 current UTC day, relative to the ClockTimeTransmitter entity of
16 this PTP Instance, will contain 59 s. It is equal to the
17 global variable sysLeap59.

18

19 The value is selected as follows:

20 a)The value is obtained from a primary reference if
21 known at the time of initialization, else
22 b)The value is set to FALSE (2)."

23 REFERENCE "14.2.10"

24 ::= { ieee8021AsV3DefaultDSEntry 11 }

25

26

27 ieee8021AsV3DefaultDSLeap61 OBJECT-TYPE

1 SYNTAX TruthValue
2 MAX-ACCESS read-only
3 STATUS current
4 DESCRIPTION
5 "A TRUE (1) value indicates that the last minute of the
6 current UTC day, relative to the ClockTimeTransmitter entity of
7 this PTP Instance, will contain 61 s. It is equal to the global
8 variable sysLeap61.
9
10 The value is selected as follows:
11 a)The value is obtained from a primary reference if
12 known at the time of initialization, else
13 b)The value is set to FALSE (2)."br/>14 REFERENCE "14.2.11"
15 ::= { ieee8021AsV3DefaultDSEntry 12 }
16
17
18 ieee8021AsV3DefaultDSTimeTraceable OBJECT-TYPE
19 SYNTAX TruthValue
20 MAX-ACCESS read-only
21 STATUS current
22 DESCRIPTION
23 "The value is set to TRUE (1) if the timescale and the value
24 of currentUtcOffset, relative to the ClockTimeTransmitter entity of
25 this PTP Instance, are traceable to a primary reference
26 standard; otherwise the value is set to FALSE (2).
27 It is equal to the global variable sysTimeTraceable.

1
2 The value is selected as follows:
3 a)If the time and the value of currentUtcOffset are
4 traceable to a primary reference standard at the time of
5 initialization, the value is set to TRUE (1), else
6 b)The value is set to FALSE (2)."

7 REFERENCE "14.2.12"

8 ::= { ieee8021AsV3DefaultDSEntry 13 }

9

10

11 ieee8021AsV3DefaultDSFrequencyTraceable OBJECT-TYPE

12 SYNTAX TruthValue

13 MAX-ACCESS read-only

14 STATUS current

15 DESCRIPTION

16 "The value is set to TRUE (1) if the frequency determining the
17 timescale of the ClockTimeTransmitter Entity of this PTP Instance is
18 traceable to a primary standard; otherwise the value is set
19 to FALSE (2). It is equal to the global variable
20 sysFrequencyTraceable.

21

22 The value is selected as follows:

23 a)If the frequency is traceable to a primary reference
24 standard at the time of initialization the value is set
25 to TRUE (1), else
26 b)The value is set to FALSE (2)."

27 REFERENCE "14.2.13"

1 ::= { ieee8021AsV3DefaultDSEntry 14 }

2

3 ieee8021AsV3DefaultDSPtpTimescale OBJECT-TYPE

4 SYNTAX TruthValue

5 MAX-ACCESS read-only

6 STATUS current

7 DESCRIPTION

8 "The value is set to TRUE (1) if the clock timescale of the

9 ClockTimeTransmitter Entity of this PTP Instance is PTP and

10 FALSE (2) otherwise."

11 REFERENCE "14.2.14"

12 ::= { ieee8021AsV3DefaultDSEntry 15 }

13

14 ieee8021AsV3DefaultDSTimeSource OBJECT-TYPE

15 SYNTAX Ieee8021AsV3TimeSourceValue

16 MAX-ACCESS read-only

17 STATUS current

18 DESCRIPTION

19 "The value is the source of time used by the

20 Grandmaster PTP Instance clock."

21 REFERENCE "14.2.15"

22 ::= { ieee8021AsV3DefaultDSEntry 16 }

23

24 ieee8021AsV3DefaultDSDomainNumber OBJECT-TYPE

25 SYNTAX Unsigned32(0..127)

26 MAX-ACCESS read-write

27 STATUS current

1 DESCRIPTION

2 "The value is the domain number of the gPTP domain for this
3 instance of gPTP supported by the time-aware system."

4 REFERENCE "14.2.16"

5 ::= { ieee8021AsV3DefaultDSEntry 17 }

6

7 ieee8021AsV3DefaultDSSdId OBJECT-TYPE

8 SYNTAX Unsigned32(0..4095)

9 MAX-ACCESS read-only

10 STATUS current

11 DESCRIPTION

12 "The value is the sdId of the gPTP domain for this instance
13 of gPTP supported by the time-aware system.

14 For compatibility with IEEE Std 1588, the range of the
15 managed object is limited to 12 bits; in addition, only the
16 single value 0x100 is specified in this standard for the
17 gPTP domain of a PTP Instance."

18 REFERENCE "14.2.17"

19 ::= { ieee8021AsV3DefaultDSEntry 18 }

20

21 ieee8021AsV3DefaultDSExternalPortConfigurationEnabled OBJECT-TYPE

22 SYNTAX TruthValue

23 MAX-ACCESS read-write

24 STATUS current

25 DESCRIPTION

26 "The value is the externalPortConfigurationEnabled attribute
27 of the PTP Instance."

1 REFERENCE "14.2.18"

2 ::= { ieee8021AsV3DefaultDSEntry 19 }

3

4 ieee8021AsV3DefaultDSInstanceEnable OBJECT-TYPE

5 SYNTAX TruthValue

6 MAX-ACCESS read-write

7 STATUS current

8 DESCRIPTION

9 "The value is the instanceEnable attribute of the PTP Instance."

10 REFERENCE "14.2.19"

11 ::= { ieee8021AsV3DefaultDSEntry 20 }

12

13 -- =====

14 -- The Current data set represents this system's topological location

15 -- relative to the known Grandmaster PTP Instance.

16 -- This data set is consistent with respective IEEE 1588 data set.

17 -- =====

18

19 ieee8021AsV3CurrentDSTable OBJECT-TYPE

20 SYNTAX SEQUENCE OF Ieee8021AsV3CurrentDSEntry

21 MAX-ACCESS not-accessible

22 STATUS current

23 DESCRIPTION

24 "The Current Parameter Data Set represents the position of a local

25 system and other information, relative to the

26 Grandmaster PTP Instance."

27 REFERENCE "14.3"

```
1 ::= { ieee8021AsV3MIBObjects 3 }

2

3 ieee8021AsV3CurrentDSEntry OBJECT-TYPE

4 SYNTAX Ieee8021AsV3CurrentDSEntry

5 MAX-ACCESS not-accessible

6 STATUS current

7 DESCRIPTION

8 "Current Data Set for a specific PTP Instance."

9 INDEX { ieee8021AsV3PtpInstance }

10 ::= { ieee8021AsV3CurrentDSTable 1 }

11

12 Ieee8021AsV3CurrentDSEntry ::=

13 SEQUENCE {

14     ieee8021AsV3CurrentDSStepsRemoved          Unsigned32,
15     ieee8021AsV3CurrentDSOffsetFromTimeTransmitter   Ieee8021ASV3PtpTimeInterval,
16     ieee8021AsV3CurrentDSLastGmPhaseChange      Ieee8021ASV3ScaledNs,
17     ieee8021AsV3CurrentDSLastGmFreqChange       Float64TC,
18     ieee8021AsV3CurrentDSGmTimebaseIndicator    Unsigned32,
19     ieee8021AsV3CurrentDSGmChangeCount         Counter32,
20     ieee8021AsV3CurrentDSTimeOfLastGmChangeEvent TimeStamp,
21     ieee8021AsV3CurrentDSTimeOfLastGmPhaseChangeEvent TimeStamp,
22     ieee8021AsV3CurrentDSTimeOfLastGmFreqChangeEvent TimeStamp
23 }

24

25 ieee8021AsV3CurrentDSStepsRemoved OBJECT-TYPE

26 SYNTAX Unsigned32(0..65535)

27 MAX-ACCESS read-only
```

1 STATUS current
2 DESCRIPTION
3 "The value is the number of gPTP communication paths
4 traversed between this PTP Instance and the
5 Grandmaster PTP Instance, as specified in 10.3.3."
6 REFERENCE "14.3.2"
7 ::= { ieee8021AsV3CurrentDSEntry 1 }
8
9

10 ieee8021AsV3CurrentDSOffsetFromTimeTransmitter OBJECT-TYPE

11 SYNTAX Ieee8021ASV3PtpTimeInterval
12 MAX-ACCESS read-only
13 STATUS current
14 DESCRIPTION
15 "The value is an implementation-specific representation of
16 the current value of the time difference between a timeReceiver
17 and the Grandmaster Clock, as computed by the timeReceiver, and
18 as specified in 10.2.10."
19 REFERENCE "14.3.3"
20 ::= { ieee8021AsV3CurrentDSEntry 2 }
21

22 ieee8021AsV3CurrentDSLastGmPhaseChange OBJECT-TYPE

23 SYNTAX Ieee8021ASV3ScaledNs
24 MAX-ACCESS read-only
25 STATUS current
26 DESCRIPTION
27 "The value is the phase change that occurred on the most

1 recent change in either Grandmaster PTP Instance or
2 gmTimeBaseIndicator."
3 REFERENCE "14.3.4"
4 ::= { ieee8021AsV3CurrentDSEntry 3 }
5
6 ieee8021AsV3CurrentDSLastGmFreqChange OBJECT-TYPE
7 SYNTAX Float64TC
8 MAX-ACCESS read-only
9 STATUS current
10 DESCRIPTION
11 "The value is the frequency change that occurred on the most
12 recent change in either Grandmaster PTP Instance or
13 gmTimeBaseIndicator."
14 REFERENCE "14.3.5"
15 ::= { ieee8021AsV3CurrentDSEntry 4 }
16
17 ieee8021AsV3CurrentDSGmTimebaseIndicator OBJECT-TYPE
18 SYNTAX Unsigned32(0..65535)
19 MAX-ACCESS read-only
20 STATUS current
21 DESCRIPTION
22 "The value is the value of timeBaseIndicator of the
23 current Grandmaster PTP Instance."
24 REFERENCE "14.3.6"
25 ::= { ieee8021AsV3CurrentDSEntry 5 }
26
27 ieee8021AsV3CurrentDSGmChangeCount OBJECT-TYPE

1 SYNTAX Counter32
2 MAX-ACCESS read-only
3 STATUS current
4 DESCRIPTION
5 "This statistics counter tracks the number of times the
6 Grandmaster PTP Instance has changed in a gPTP domain.
7 This counter increments when the PortAnnounceInformation
8 state machine enters the SUPERIOR_TIME_TRANSMITTER_PORT state
9 or the INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state."
10 REFERENCE "14.3.7"
11 ::= { ieee8021AsV3CurrentDSEntry 6 }
12
13 ieee8021AsV3CurrentDSTimeOfLastGmChangeEvent OBJECT-TYPE
14 SYNTAX TimeStamp
15 UNITS "0.01 seconds"
16 MAX-ACCESS read-only
17 STATUS current
18 DESCRIPTION
19 "This timestamp takes the value of sysUpTime (see RFC3418) when
20 the most recent Grandmaster PTP Instance change occurred in
21 a gPTP domain.
22 This timestamp is updated when the PortAnnounceInformation
23 state machine enters the SUPERIOR_TIME_TRANSMITTER_PORT state or
24 the INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state."
25 REFERENCE "14.3.8"
26 ::= { ieee8021AsV3CurrentDSEntry 7 }
27

1

2 ieee8021AsV3CurrentDSTimeOfLastGmPhaseChangeEvent OBJECT-TYPE

3 SYNTAX TimeStamp

4 UNITS "0.01 seconds"

5 MAX-ACCESS read-only

6 STATUS current

7 DESCRIPTION

8 "This timestamp takes the value of sysUpTime (see RFC3418)

9 when the most recent change in Grandmaster Clock phase

10 occurred, due to a change of either the

11 Grandmaster PTP Instance or the Grandmaster Clock

12 time base. This timestamp is updated when one of the

13 following occurs:

14 a)The PortAnnounceInformation state machine enters the

15 SUPERIOR_TIME_TRANSMITTER_PORT state or the

16 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state, or

17 b)The gmTimebaseIndicator managed object changes and the

18 lastGmPhaseChange field of the most recently received

19 Follow_Up information TLV is nonzero."

20 REFERENCE "14.3.9"

21 ::= { ieee8021AsV3CurrentDSEntry 8 }

22

23 ieee8021AsV3CurrentDSTimeOfLastGmFreqChangeEvent OBJECT-TYPE

24 SYNTAX TimeStamp

25 UNITS "0.01 seconds"

26 MAX-ACCESS read-only

27 STATUS current

1 DESCRIPTION

2 "This timestamp takes the value of sysUpTime (see RFC3418)
3 when the most recent change in Grandmaster Clock frequency
4 occurred, due to a change of either the Grandmaster PTP
5 Instance or the Grandmaster Clock time base. This timestamp
6 is updated when one of the following occurs:
7 a)The PortAnnounceInformation state machine enters the
8 SUPERIOR_TIME_TRANSMITTER_PORT state or the
9 INFERIOR_TIME_TRANSMITTER_OR_OTHER_PORT state, or
10 b)The gmTimebaseIndicator managed object changes and the
11 lastGmFreqChange field of the most recently received
12 Follow_Up information TLV is nonzero."

13 REFERENCE "14.3.10"

14 ::= { ieee8021AsV3CurrentDSEntry 9 }

15

16 -- =====

17 -- The Parent data set represents the upstream (toward
18 -- Grandmaster PTP Instance) system's timing parameters as measured
19 -- at this system.
20 -- This data set is consistent with the respective IEEE 1588 data set.

21 -- =====

22 ieee8021AsV3ParentDSTable OBJECT-TYPE

23 SYNTAX SEQUENCE OF Ieee8021AsV3ParentDSEntry

24 MAX-ACCESS not-accessible

25 STATUS current

26 DESCRIPTION

27 "The Parent Parameter Data Set represents capabilities of the

1 upstream system, toward the Grandmaster PTP Instance, as
2 measured at a local system."

3 REFERENCE "14.4"

4 ::= { ieee8021AsV3MIBObjects 4 }

5

6 ieee8021AsV3ParentDSEntry OBJECT-TYPE

7 SYNTAX Ieee8021AsV3ParentDSEntry

8 MAX-ACCESS not-accessible

9 STATUS current

10 DESCRIPTION

11 "Parent Data Set for a specific PTP Instance."

12 INDEX { ieee8021AsV3PtpInstance }

13 ::= { ieee8021AsV3ParentDSTable 1 }

14

15 Ieee8021AsV3ParentDSEntry ::=

16 SEQUENCE {

17 ieee8021AsV3ParentDSParentClockIdentity Ieee8021AsV3ClockIdentity,

18 ieee8021AsV3ParentDSParentPortNumber Unsigned32,

19 ieee8021AsV3ParentDSCumulativeRateRatio Integer32,

20 ieee8021AsV3ParentDSGrandmasterIdentity Ieee8021AsV3ClockIdentity,

21 ieee8021AsV3ParentDSGrandmasterClockQualityclockClass

Ieee8021AsV3ClockClassValue,

23 ieee8021AsV3ParentDSGrandmasterClockQualityclockAccuracy

24 Ieee8021AsV3ClockAccuracyValue,

25 ieee8021AsV3ParentDSGrandmasterClockQualityoffsetScaledLogVar

26 Unsigned32,

27 ieee8021AsV3ParentDSGrandmasterPriority1 Unsigned32,

1 ieee8021AsV3ParentDSGrandmasterPriority2 Unsigned32
2 }

3

4 ieee8021AsV3ParentDSParentClockIdentity OBJECT-TYPE

5 SYNTAX Ieee8021AsV3ClockIdentity

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "The value is the first of the parentPortIdentity attribute
10 for this instance of gPTP domain, which is a set made of
11 Ieee8021AsV3ClockIdentity and portNumber."

12 REFERENCE "14.4.2"

13 ::= { ieee8021AsV3ParentDSEntry 1 }

14

15 ieee8021AsV3ParentDSParentPortNumber OBJECT-TYPE

16 SYNTAX Unsigned32(0..65535)

17 MAX-ACCESS read-only

18 STATUS current

19 DESCRIPTION

20 "The value is the second of the parentPortIdentity attribute
21 for this instance of gPTP domain, which is a set made of
22 Ieee8021AsV3ClockIdentity and portNumber."

23 REFERENCE "14.4.2"

24 ::= { ieee8021AsV3ParentDSEntry 2 }

25

26 ieee8021AsV3ParentDSCumulativeRateRatio OBJECT-TYPE

27 SYNTAX Integer32

1 MAX-ACCESS read-only
2 STATUS current
3 DESCRIPTION
4 "The value is an estimate of the ratio of the frequency of
5 the Grandmaster Clock to the frequency of the LocalClock
6 entity of this PTP Instance.
7 CumulativeRateRatio is expressed as the fractional
8 frequency offset multiplied by 2^{41} , i.e., the quantity
9 $(\text{rateRatio} - 1.0)(2^{41})$, where rateRatio is computed by
10 the PortSyncSyncReceive state machine."
11 REFERENCE "14.4.3"
12 ::= { ieee8021AsV3ParentDSEntry 3 }
13
14 ieee8021AsV3ParentDSGrandmasterIdentity OBJECT-TYPE
15 SYNTAX Ieee8021AsV3ClockIdentity
16 MAX-ACCESS read-only
17 STATUS current
18 DESCRIPTION
19 "The value is the clockIdentity attribute of the
20 Grandmaster PTP Instance."
21 REFERENCE "14.4.4"
22 ::= { ieee8021AsV3ParentDSEntry 4 }
23
24 ieee8021AsV3ParentDSGrandmasterClockQualityclockClass OBJECT-TYPE
25 SYNTAX Ieee8021AsV3ClockClassValue
26 MAX-ACCESS read-only
27 STATUS current

1 DESCRIPTION

2 "The value is the clockClass of the Grandmaster PTP Instance."

3 REFERENCE "14.4.5.2"

4 ::= { ieee8021AsV3ParentDSEntry 5 }

5

6 ieee8021AsV3ParentDSGrandmasterClockQualityclockAccuracy OBJECT-TYPE

7 SYNTAX Ieee8021AsV3ClockAccuracyValue

8 MAX-ACCESS read-only

9 STATUS current

10 DESCRIPTION

11 "The value is the clockAccuracy of the Grandmaster PTP Instance."

12 REFERENCE "14.4.5.3"

13 ::= { ieee8021AsV3ParentDSEntry 6 }

14

15 ieee8021AsV3ParentDSGrandmasterClockQualityoffsetScaledLogVar

16 OBJECT-TYPE

17 SYNTAX Unsigned32(0..65535)

18 MAX-ACCESS read-only

19 STATUS current

20 DESCRIPTION

21 "The value is the offsetScaledLogVariance of the

22 Grandmaster PTP Instance."

23 REFERENCE "14.4.5.4"

24 ::= { ieee8021AsV3ParentDSEntry 7 }

25

26 ieee8021AsV3ParentDSGrandmasterPriority1 OBJECT-TYPE

27 SYNTAX Unsigned32(0..255)

1 MAX-ACCESS read-only
2 STATUS current
3 DESCRIPTION
4 "The value is the priority1 attribute of the
5 Grandmaster PTP Instance."
6 REFERENCE "14.4.6"
7 ::= { ieee8021AsV3ParentDSEntry 8 }
8
9 ieee8021AsV3ParentDSGrandmasterPriority2 OBJECT-TYPE
10 SYNTAX Unsigned32(0..255)
11 MAX-ACCESS read-only
12 STATUS current
13 DESCRIPTION
14 "The value is the priority2 attribute of the
15 Grandmaster PTP Instance."
16 REFERENCE "14.4.7"
17 ::= { ieee8021AsV3ParentDSEntry 9 }
18
19 -- =====
20 -- TimePropertiesDS represents the Grandmaster PTP Instance's
21 -- parameters, as measured at this system and are derived from
22 -- IEEE 802.1AS protocol.
23 -- This data set is consistent with respective IEEE 1588 data set.
24 -- =====
25 ieee8021AsV3TimePropertiesDSTable OBJECT-TYPE
26 SYNTAX SEQUENCE OF Ieee8021AsV3TimePropertiesDSEntry
27 MAX-ACCESS not-accessible

1 STATUS current

2 DESCRIPTION

3 "The Time Properties Parameter Data Set represents capabilities of

4 the Grandmaster PTP Instance, as measured at a local system"

5 REFERENCE "14.5"

6 ::= { ieee8021AsV3MIBObjects 5 }

7

8 ieee8021AsV3TimePropertiesDSEntry OBJECT-TYPE

9 SYNTAX Ieee8021AsV3TimePropertiesDSEntry

10 MAX-ACCESS not-accessible

11 STATUS current

12 DESCRIPTION

13 "Time Properties Data Set contains the profile Identifier for

14 this instance of gPTP domain."

15 INDEX { ieee8021AsV3PtpInstance }

16 ::= { ieee8021AsV3TimePropertiesDSTable 1 }

17

18 Ieee8021AsV3TimePropertiesDSEntry ::=

19 SEQUENCE {

20 ieee8021AsV3TimePropertiesDSCurrentUtcOffset Integer32,

21 ieee8021AsV3TimePropertiesDSCurrentUtcOffsetValid TruthValue,

22 ieee8021AsV3TimePropertiesDSLeap59 TruthValue,

23 ieee8021AsV3TimePropertiesDSLeap61 TruthValue,

24 ieee8021AsV3TimePropertiesDSTimeTraceable TruthValue,

25 ieee8021AsV3TimePropertiesDSFrequencyTraceable TruthValue,

26 ieee8021AsV3TimePropertiesDSPtpTimescale TruthValue,

27 ieee8021AsV3TimePropertiesDSTimeSource Ieee8021AsV3TimeSourceValue

1 }

2

3 ieee8021AsV3TimePropertiesDSCurrentUtcOffset OBJECT-TYPE

4 SYNTAX Integer32(-32768..32767)

5 UNITS "seconds"

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "The value is currentUtcOffset for the current

10 Grandmaster PTP Instance. It is equal to the value of

11 the global variable currentUtcOffset. The value is in

12 units of seconds."

13 REFERENCE "14.5.2"

14 ::= { ieee8021AsV3TimePropertiesDSEntry 1 }

15

16 ieee8021AsV3TimePropertiesDSCurrentUtcOffsetValid OBJECT-TYPE

17 SYNTAX TruthValue

18 MAX-ACCESS read-only

19 STATUS current

20 DESCRIPTION

21 "The value is currentUtcOffsetValid for the current

22 Grandmaster PTP Instance. It is equal to the global

23 variable currentUtcOffsetValid."

24 REFERENCE "14.5.3"

25 ::= { ieee8021AsV3TimePropertiesDSEntry 2 }

26

27 ieee8021AsV3TimePropertiesDSLeap59 OBJECT-TYPE

1 SYNTAX TruthValue
2 MAX-ACCESS read-only
3 STATUS current
4 DESCRIPTION
5 "The value is leap59 for the current Grandmaster PTP Instance.
6 It is equal to the global variable leap59."
7 REFERENCE "14.5.4"
8 ::= { ieee8021AsV3TimePropertiesDSEntry 3 }
9
10 ieee8021AsV3TimePropertiesDSLeap61 OBJECT-TYPE
11 SYNTAX TruthValue
12 MAX-ACCESS read-only
13 STATUS current
14 DESCRIPTION
15 "The value is leap61 for the current Grandmaster PTP Instance.
16 It is equal to the global variable leap61."
17 REFERENCE "14.5.5"
18 ::= { ieee8021AsV3TimePropertiesDSEntry 4 }
19
20 ieee8021AsV3TimePropertiesDSTimeTraceable OBJECT-TYPE
21 SYNTAX TruthValue
22 MAX-ACCESS read-only
23 STATUS current
24 DESCRIPTION
25 "The value is timeTraceable for the current
26 Grandmaster PTP Instance. It is equal to the global
27 variable timeTraceable."

1 REFERENCE "14.5.6"
2 ::= { ieee8021AsV3TimePropertiesDSEntry 5 }
3
4 ieee8021AsV3TimePropertiesDSFrequencyTraceable OBJECT-TYPE
5 SYNTAX TruthValue
6 MAX-ACCESS read-only
7 STATUS current
8 DESCRIPTION
9 "The value is frequencyTraceable for the current
10 Grandmaster PTP Instance. It is equal to the global
11 variable frequencyTraceable."
12 REFERENCE "14.5.7"
13 ::= { ieee8021AsV3TimePropertiesDSEntry 6 }
14
15 ieee8021AsV3TimePropertiesDSPtpTimescale OBJECT-TYPE
16 SYNTAX TruthValue
17 MAX-ACCESS read-only
18 STATUS current
19 DESCRIPTION
20 "The value is ptpTimescale for the current
21 Grandmaster PTP Instance."
22 REFERENCE "14.5.8"
23 ::= { ieee8021AsV3TimePropertiesDSEntry 7 }
24
25 ieee8021AsV3TimePropertiesDSTimeSource OBJECT-TYPE
26 SYNTAX Ieee8021AsV3TimeSourceValue
27 MAX-ACCESS read-only

1 STATUS current
2 DESCRIPTION
3 "The value is timeSource for the current
4 Grandmaster PTP Instance. It is equal to the global
5 variable timeSource"
6 REFERENCE "14.5.9"
7 ::= { ieee8021AsV3TimePropertiesDSEntry 8 }
8
9 -- ======
10 -- The Path Trace Parameter Data set represents the current path
11 -- trace information available at the PTP Instance.
12 -- ======
13
14 ieee8021AsV3PathTraceDSTable OBJECT-TYPE
15 SYNTAX SEQUENCE OF Ieee8021AsV3PathTraceDSEntry
16 MAX-ACCESS not-accessible
17 STATUS current
18 DESCRIPTION
19 "The pathTraceDS represents the current path trace information
20 available at the PTP Instance."
21 REFERENCE "14.6"
22 ::= { ieee8021AsV3MIBObjects 6 }
23
24 ieee8021AsV3PathTraceDSEntry OBJECT-TYPE
25 SYNTAX Ieee8021AsV3PathTraceDSEntry
26 MAX-ACCESS not-accessible
27 STATUS current

1 DESCRIPTION

2 "Path Trace Data Set for a specific PTP Instance."

3 INDEX { ieee8021AsV3PtpInstance }

4 ::= { ieee8021AsV3PathTraceDSTable 1 }

5

6 Ieee8021AsV3PathTraceDSEntry ::=

7 SEQUENCE {

8 ieee8021AsV3PathTraceDSEnable TruthValue

9 }

10

11 ieee8021AsV3PathTraceDSEnable OBJECT-TYPE

12 SYNTAX TruthValue

13 MAX-ACCESS read-only

14 STATUS current

15 DESCRIPTION

16 "The value is TRUE.

17 NOTE: This member is included for compatibility with

18 IEEE Std 1588. In IEEE Std 1588, the path trace mechanism

19 is optional, and the pathTraceDS.enable member is

20 configurable (its value in IEEE Std 1588 is TRUE (1) or

21 FALSE (2), depending on whether the path trace mechanism is

22 operational or not operational, respectively. However, the

23 pathTrace mechanism is mandatory in this standard, and the

24 value of enable is always TRUE (1)."

25 REFERENCE "14.6.3"

26 ::= { ieee8021AsV3PathTraceDSEntry 2 }

27

```
1 ieee8021AsV3PathTraceDSArrayTable OBJECT-TYPE
2   SYNTAX   SEQUENCE OF Ieee8021AsV3PathTraceDSArrayEntry
3   MAX-ACCESS not-accessible
4   STATUS    current
5   DESCRIPTION
6     "This object contains an array of ClockIdentity values contained
7     in the pathTrace array, which represents the current path trace
8     information, and which is carried in the path trace TLV per
9     PTP Instance."
10  REFERENCE "14.6.2"
11 ::= { ieee8021AsV3MIBObjects 7 }
12
13 ieee8021AsV3PathTraceDSArrayEntry OBJECT-TYPE
14   SYNTAX   Ieee8021AsV3PathTraceDSArrayEntry
15   MAX-ACCESS not-accessible
16   STATUS    current
17   DESCRIPTION
18     "Path Trace Data Set Table Array for a specific PTP Instance."
19   INDEX { ieee8021AsV3PtpInstance, ieee8021AsV3PathTraceDSArrayIndex }
20 ::= { ieee8021AsV3PathTraceDSArrayTable 1 }
21
22 Ieee8021AsV3PathTraceDSArrayEntry :=
23   SEQUENCE {
24     ieee8021AsV3PathTraceDSArrayIndex Unsigned32,
25     ieee8021AsV3PathTraceDSArrayList Ieee8021AsV3ClockIdentity
26   }
27
```

1 ieee8021AsV3PathTraceDSArrayIndex OBJECT-TYPE

2 SYNTAX Unsigned32 (1..179)

3 MAX-ACCESS not-accessible

4 STATUS current

5 DESCRIPTION

6 "Index of the Path Trace Data Set Array."

7 REFERENCE "10.3.9.23"

8 ::= { ieee8021AsV3PathTraceDSArrayEntry 1 }

9

10 ieee8021AsV3PathTraceDSArrayList OBJECT-TYPE

11 SYNTAX Ieee8021AsV3ClockIdentity

12 MAX-ACCESS read-only

13 STATUS current

14 DESCRIPTION

15 "The value is the array of ClockIdentity values contained

16 in the pathTrace array, which represents the current

17 path trace information, and which is carried in the path

18 trace TLV."

19 REFERENCE "14.6.2"

20 ::= { ieee8021AsV3PathTraceDSArrayEntry 2 }

21

22

23 -- ****

24 -- The Acceptable TimeTransmitter Table Parameter Data Set represents the

25 -- acceptable timeTransmitter table used when an EPON port is used by a PTP

26 -- Instance of a time-aware system.

27 -- ****

1

2 ieee8021AsV3AcceptableTimeTransmitterTableDSTable OBJECT-TYPE

3 SYNTAX SEQUENCE OF Ieee8021AsV3AcceptableTimeTransmitterTableDSEntry

4 MAX-ACCESS not-accessible

5 STATUS current

6 DESCRIPTION

7 "The acceptableTimeTransmitterTableDS represents the acceptable timeTransmitter

8 table used when an EPON port is used by a PTP Instance of a

9 time-aware system."

10 REFERENCE "14.7"

11 ::= { ieee8021AsV3MIBObjects 8 }

12

13 ieee8021AsV3AcceptableTimeTransmitterTableDSEntry OBJECT-TYPE

14 SYNTAX Ieee8021AsV3AcceptableTimeTransmitterTableDSEntry

15 MAX-ACCESS not-accessible

16 STATUS current

17 DESCRIPTION

18 "Acceptable TimeTransmitter Table Data Set represents the acceptable timeTransmitter

19 table used when an EPON port is used by a PTP Instance of a

20 time-aware system."

21 INDEX { ieee8021AsV3PtpInstance }

22 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSTable 1 }

23

24 Ieee8021AsV3AcceptableTimeTransmitterTableDSEntry ::=

25 SEQUENCE {

26 ieee8021AsV3AcceptableTimeTransmitterTableDSMaxTableSize Unsigned32,

27 ieee8021AsV3AcceptableTimeTransmitterTableDSActualTableSize Unsigned32

1 }

2

3 ieee8021AsV3AcceptableTimeTransmitterTableDSMaxTableSize OBJECT-TYPE

4 SYNTAX Unsigned32(0..65535)

5 MAX-ACCESS read-only

6 STATUS current

7 DESCRIPTION

8 "The value is the maximum size of the AcceptableTimeTransmitterTable.

9 It is equal to the maxTableSize member of the

10 AcceptableTimeTransmitterTable structure."

11 REFERENCE "14.7.2"

12 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSEntry 1 }

13

14 ieee8021AsV3AcceptableTimeTransmitterTableDSActualTableSize OBJECT-TYPE

15 SYNTAX Unsigned32(0..65535)

16 MAX-ACCESS read-write

17 STATUS current

18 DESCRIPTION

19 "The value is the actual size of the AcceptableTimeTransmitterTable.

20 It is equal to the actualTableSize member of the

21 AcceptableTimeTransmitterTable structure, i.e., the current number

22 of elements in the acceptable timeTransmitter array. The actual

23 table size is less than or equal to the max table size."

24 REFERENCE "14.7.3"

25 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSEntry 2 }

26

27 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayTable OBJECT-TYPE

```

1  SYNTAX   SEQUENCE OF Ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry
2  MAX-ACCESS not-accessible
3  STATUS    current
4  DESCRIPTION
5      "The acceptableTimeTransmitterTableDS represents the acceptable timeTransmitter table
6      used when an EPON port is used by a PTP Instance of a time-aware
7      system."
8  REFERENCE  "14.7"
9 ::= { ieee8021AsV3MIBObjects 9 }
10
11 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry OBJECT-TYPE
12  SYNTAX   Ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry
13  MAX-ACCESS not-accessible
14  STATUS    current
15  DESCRIPTION
16      "Each element of this array is an AcceptableTimeTransmitter structure per
17      PTP Instance."
18  INDEX { ieee8021AsV3PtpInstance, ieee8021AsV3AcceptableTimeTransmitterTableDSArrayIndex }
19  ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSArrayTable 1 }
20
21 Ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry ::=

22 SEQUENCE {
23     ieee8021AsV3AcceptableTimeTransmitterTableDSArrayIndex          Unsigned32,
24     ieee8021AsV3AcceptableTimeTransmitterTableDSArrayPortIdentity  Ieee8021ASV3PtpPortIdentity,
25     ieee8021AsV3AcceptableTTTableDSArrayAlternatePriority1        Unsigned32
26 }
27

```

1 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayIndex OBJECT-TYPE

2 SYNTAX Unsigned32(0..65535)

3 MAX-ACCESS not-accessible

4 STATUS current

5 DESCRIPTION

6 "Index of the Acceptable TimeTransmitter Table Data Set Array."

7 REFERENCE "14.7.4"

8 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry 1 }

9

10 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayPortIdentity OBJECT-TYPE

11 SYNTAX Ieee8021ASV3PtpPortIdentity

12 MAX-ACCESS read-write

13 STATUS current

14 DESCRIPTION

15 "The acceptablePortIdentity member is the PortIdentity of

16 an acceptable timeTransmitter port."

17 REFERENCE "14.7.4"

18 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry 2 }

19

20 ieee8021AsV3AcceptableTTTableDSArrayAlternatePriority1 OBJECT-TYPE

21 SYNTAX Unsigned32(0..255)

22 MAX-ACCESS read-write

23 STATUS current

24 DESCRIPTION

25 "The alternatePriority1 member contains an alternate value

26 for the priority1 attribute of the acceptable timeTransmitter port."

27 REFERENCE "14.7.4"

1 ::= { ieee8021AsV3AcceptableTimeTransmitterTableDSArrayEntry 3 }

2

3 -- =====

4 -- The Port Parameter Data Set (portDS) represents PTP Port

5 -- time-aware capabilities for a PTP Instance of a time-aware

6 -- system.

7 -- =====

8 ieee8021AsV3PortDSTable OBJECT-TYPE

9 SYNTAX SEQUENCE OF Ieee8021AsV3PortDSEntry

10 MAX-ACCESS not-accessible

11 STATUS current

12 DESCRIPTION

13 "For the single PTP Port of a PTP End Instance and for each

14 PTP Port of a PTP Relay Instance , the portDS is maintained

15 as the basis for making protocol decisions and providing

16 values for message fields.

17 The number of such data sets is the same as the value of

18 defaultDS.numberPorts."

19 REFERENCE "14.8"

20 ::= { ieee8021AsV3MIBObjects 10 }

21

22 ieee8021AsV3PortDSEntry OBJECT-TYPE

23 SYNTAX Ieee8021AsV3PortDSEntry

24 MAX-ACCESS not-accessible

25 STATUS current

26 DESCRIPTION

27 "A list of objects pertaining to a PTP Port of a PTP Instance."

```
1 INDEX { ieee8021AsV3PtpInstance,
2     ieee8021AsV3PortDSIndex }
3 ::= { ieee8021AsV3PortDSTable 1 }
4
5 Ieee8021AsV3PortDSEntry ::=

6 SEQUENCE {
7     ieee8021AsV3BridgeBasePort      IEEE8021BridgePortNumber,
8     ieee8021AsV3PortDSIndex        InterfaceIndexOrZero,
9     ieee8021AsV3PortDSClockIdentity    Ieee8021AsV3ClockIdentity,
10    ieee8021AsV3PortDSPortNumber     Unsigned32,
11    ieee8021AsV3PortDSPortState      INTEGER,
12    ieee8021AsV3PortDSPtpPortEnabled   TruthValue,
13    ieee8021AsV3PortDSDelayMechanism   INTEGER,
14    ieee8021AsV3PortDSIsMeasuringDelay   TruthValue,
15    ieee8021AsV3PortDSAsCapable      TruthValue,
16    ieee8021AsV3PortDSMeanLinkDelay    Ieee8021ASV3PtpTimeInterval,
17    ieee8021AsV3PortDSMeanLinkDelayThresh  Ieee8021ASV3PtpTimeInterval,
18    ieee8021AsV3PortDSDelayAsym      Ieee8021ASV3PtpTimeInterval,
19    ieee8021AsV3PortDSNbrRateRatio     Integer32,
20    ieee8021AsV3PortDSInitialLogAnnounceInterval Integer32,
21    ieee8021AsV3PortDSCurrentLogAnnounceInterval Integer32,
22    ieee8021AsV3PortDSUseMgtSettableLogAnnounceInterval TruthValue,
23    ieee8021AsV3PortDSMgtSettableLogAnnounceInterval Integer32,
24    ieee8021AsV3PortDSAnnounceReceiptTimeout   Unsigned32,
25    ieee8021AsV3PortDSInitialLogSyncInterval Integer32,
26    ieee8021AsV3PortDSCurrentLogSyncInterval Integer32,
27    ieee8021AsV3PortDSUseMgtSettableLogSyncInterval TruthValue,
```

1 ieee8021AsV3PortDSMgtSettableLogSyncInterval Integer32,
2 ieee8021AsV3PortDSSyncReceiptTimeout Unsigned32,
3 ieee8021AsV3PortDSSyncReceiptTimeoutTimeInterval Ieee8021ASV3UScaledNs,
4 ieee8021AsV3PortDSInitialLogPdelayReqInterval Integer32,
5 ieee8021AsV3PortDSCurrentLogPdelayReqInterval Integer32,
6 ieee8021AsV3PortDSUseMgtSettableLogPdelayReqInterval TruthValue,
7 ieee8021AsV3PortDSMgtSettableLogPdelayReqInterval Integer32,
8 ieee8021AsV3PortDSInitialLogGtpCapableMessageInterval Integer32,
9 ieee8021AsV3PortDSCurrentLogGtpCapableMessageInterval Integer32,
10 ieee8021AsV3PortDSUseMgtSettableLogGtpCapableMessageInterval TruthValue,
11 ieee8021AsV3PortDSMgtSettableLogGtpCapableMessageInterval Integer32,
12 ieee8021AsV3PortDSInitialComputeNbrRateRatio TruthValue,
13 ieee8021AsV3PortDSCurrentComputeNbrRateRatio TruthValue,
14 ieee8021AsV3PortDSUseMgtSettableComputeNbrRateRatio TruthValue,
15 ieee8021AsV3PortDSMgtSettableComputeNbrRateRatio TruthValue,
16 ieee8021AsV3PortDSInitialComputeMeanLinkDelay TruthValue,
17 ieee8021AsV3PortDSCurrentComputeMeanLinkDelay TruthValue,
18 ieee8021AsV3PortDSUseMgtSettableComputeMeanLinkDelay TruthValue,
19 ieee8021AsV3PortDSMgtSettableComputeMeanLinkDelay TruthValue,
20 ieee8021AsV3PortDSAllowedLostRsp Unsigned32,
21 ieee8021AsV3PortDSAllowedFaults Unsigned32,
22 ieee8021AsV3PortDSGPtpCapableReceiptTimeout Unsigned32,
23 ieee8021AsV3PortDSVersionNumber Unsigned32,
24 ieee8021AsV3PortDSNup Float64TC,
25 ieee8021AsV3PortDSNdown Float64TC,
26 ieee8021AsV3PortDSOneStepTxOper TruthValue,
27 ieee8021AsV3PortDSOneStepReceive TruthValue,

```

1     ieee8021AsV3PortDSOneStepTransmit      TruthValue,
2     ieee8021AsV3PortDSInitialOneStepTxOper   TruthValue,
3     ieee8021AsV3PortDSCurrentOneStepTxOper   TruthValue,
4     ieee8021AsV3PortDSUseMgtSettableOneStepTxOper   TruthValue,
5     ieee8021AsV3PortDSMgtSettableOneStepTxOper   TruthValue,
6     ieee8021AsV3PortDSSyncLocked      TruthValue,
7     ieee8021AsV3PortDSPdelayTruncTST1    Ieee8021ASV3Timestamp,
8     ieee8021AsV3PortDSPdelayTruncTST2    Ieee8021ASV3Timestamp,
9     ieee8021AsV3PortDSPdelayTruncTST3    Ieee8021ASV3Timestamp,
10    ieee8021AsV3PortDSPdelayTruncTST4    Ieee8021ASV3Timestamp,
11    ieee8021AsV3PortDSMinorVersionNumber Unsigned32
12  }
13

```

14 ieee8021AsV3BridgeBasePort OBJECT-TYPE

15 SYNTAX IEEE8021BridgePortNumber

16 MAX-ACCESS not-accessible

17 STATUS current

18 DESCRIPTION

19 "This object identifies the bridge port number of the port for

20 which this entry contains bridge management information.

21 For end stations, this port number shall be (1)."

22 ::= { ieee8021AsV3PortDSEntry 1 }

23

24 ieee8021AsV3PortDSIndex OBJECT-TYPE

25 SYNTAX InterfaceIndexOrZero

26 MAX-ACCESS not-accessible

27 STATUS current

1 DESCRIPTION

2 "This object identifies the gPTP interface group within
3 the system for which this entry contains information. It
4 is the value of the instance of the IfIndex object,
5 defined in the IF-MIB, for the gPTP interface group
6 corresponding to this port, or the value 0 if the port
7 has not been bound to an underlying frame source and
8 sink.

9

10 For a given media port of a Bridge or an end station,
11 there can be one or more PTP Port, and depends whether
12 a media port supports point to point link (e.g. IEEE
13 802.3 Ethernet) or point to multi-point (e.g. CSN, IEEE
14 802.3 EPON) links on the media port."

15 ::= { ieee8021AsV3PortDSEntry 2 }

16

17 ieee8021AsV3PortDSClockIdentity OBJECT-TYPE

18 SYNTAX Ieee8021AsV3ClockIdentity

19 MAX-ACCESS read-only

20 STATUS current

21 DESCRIPTION

22 "The value is the first of the portIdentity attribute
23 of the local port, which is a set made of
24 Ieee8021AsV3ClockIdentity and portNumber."

25 REFERENCE "14.8.2"

26 ::= { ieee8021AsV3PortDSEntry 3 }

27

1
2 ieee8021AsV3PortDSPortNumber OBJECT-TYPE
3 SYNTAX Unsigned32(0..65535)
4 MAX-ACCESS read-only
5 STATUS current
6 DESCRIPTION
7 "The value is the second of the portIdentity attribute
8 of the local port, which is a set made of
9 Ieee8021AsV3ClockIdentity and portNumber."
10 REFERENCE "14.8.2"
11 ::= { ieee8021AsV3PortDSEntry 4 }
12
13 ieee8021AsV3PortDSPortState OBJECT-TYPE
14 SYNTAX INTEGER {
15 disabledPort(3),
16 timeTransmitterPort(6),
17 passivePort(7),
18 timeReceiverPort(9)
19 }
20 MAX-ACCESS read-only
21 STATUS current
22 DESCRIPTION
23 "The value is the value of the PTP Port state of this
24 PTP Port (see Table 10-2) and is taken from the enumeration
25 in Table 14-7. It is equal to the value of the global
26 variable selectedState."
27 REFERENCE "14.8.3"

1 ::= { ieee8021AsV3PortDSEntry 5 }

2

3 ieee8021AsV3PortDSPtpPortEnabled OBJECT-TYPE

4 SYNTAX TruthValue

5 MAX-ACCESS read-write

6 STATUS current

7 DESCRIPTION

8 "The value is equal to the value of the Boolean ptpNetEnabled.

9 Setting this managed object causes the Boolean ptpNetEnabled

10 to have the same value."

11 REFERENCE "14.8.4"

12 ::= { ieee8021AsV3PortDSEntry 6 }

13

14 ieee8021AsV3PortDSDelayMechanism OBJECT-TYPE

15 SYNTAX INTEGER {

16 p2p(2),

17 commonp2p(3),

18 special(4)

19 }

20 MAX-ACCESS read-write

21 STATUS current

22 DESCRIPTION

23 "The value indicates the mechanism for measuring mean

24 propagation delay and neighbor rate ratio on the link

25 attached to this PTP Port, and is taken from the enumeration

26 in Table 14-8. If the domain number is not 0, portDS.delay

27 mechanism must not be P2P."

1 REFERENCE "14.8.5"
2 ::= { ieee8021AsV3PortDSEntry 7 }
3
4 ieee8021AsV3PortDSIsMeasuringDelay OBJECT-TYPE
5 SYNTAX TruthValue
6 MAX-ACCESS read-only
7 STATUS current
8 DESCRIPTION
9 "The value is equal to the value of the Boolean isMeasuringDelay."
10 REFERENCE "14.8.6"
11 ::= { ieee8021AsV3PortDSEntry 8 }
12
13
14 ieee8021AsV3PortDSAsCapable OBJECT-TYPE
15 SYNTAX TruthValue
16 MAX-ACCESS read-only
17 STATUS current
18 DESCRIPTION
19 "The value is equal to the value of the Boolean asCapable."
20 REFERENCE "14.8.7"
21 ::= { ieee8021AsV3PortDSEntry 9 }
22
23 ieee8021AsV3PortDSMeanLinkDelay OBJECT-TYPE
24 SYNTAX Ieee8021ASV3PtpTimeInterval
25 MAX-ACCESS read-only
26 STATUS current
27 DESCRIPTION

1 "The value is equal to the value of the per-PTP Port global
2 variable meanLinkDelay. It is an estimate of the current
3 one-way propagation time on the link attached to this
4 PTP Port, measured as specified for the respective medium.
5 The value is zero for PTP Port attached to IEEE 802.3
6 EPON links and for the timeTransmitter port of an IEEE 802.11 link,
7 because one-way propagation delay is not measured on the
8 latter and not directly measured on the former.
9 NOTE: The underlying per-PTP Port, global variable
10 meanLinkDelay is of type UScaledNS, which is a 96-Bit value.
11 meanLinkDelay values that are larger than the maximum value
12 that can be represented by the TimeInterval data type, i.e.,
13 0xFFFF FFFF FFFF FFFF (where the units are 2^{16} ns), used
14 for this managed object are set to this largest value."
15 REFERENCE "14.8.8"
16 ::= { ieee8021AsV3PortDSEntry 10 }
17
18
19 ieee8021AsV3PortDSMeanLinkDelayThresh OBJECT-TYPE
20 SYNTAX Ieee8021ASV3PtpTimeInterval
21 MAX-ACCESS read-write
22 STATUS current
23 DESCRIPTION
24 "The value is equal to the value of the per-PTP Port global
25 variable meanLinkDelayThresh. It is the propagation time
26 threshold above which a PTP Port is considered not capable of
27 participating in the IEEE 802.1AS protocol.

1 Setting this managed object causes the per PTP Port global
2 variable meanLinkDelayThresh to have the same value.
3 NOTE: The underlying per-PTP Port, global variable
4 meanLinkDelayThresh is of type UScaledNS, which is a 96-Bit
5 value. meanLinkDelayThresh values that are larger than the
6 maximum value that can be represented by the TimeInterval
7 data type, i.e., 0xFFFF FFFF FFFF FFFF (where the units are
8 2^{-16} ns), used for this managed object are set to this
9 largest value."

10 REFERENCE "14.8.9"

11 ::= { ieee8021AsV3PortDSEntry 11 }

12

13 ieee8021AsV3PortDSDelayAsym OBJECT-TYPE

14 SYNTAX Ieee8021ASV3PtpTimeInterval

15 MAX-ACCESS read-write

16 STATUS current

17 DESCRIPTION

18 "The value is the asymmetry in the propagation delay on
19 the link attached to this PTP Port relative to the
20 Grandmaster Clock time base, as defined in 10.2.5.9 and
21 8.3. If propagation delay asymmetry is not modeled, then
22 delayAsymmetry is 0.

23 NOTE: The underlying per-port global variable delayAsymmetry

24 is of type ScaledNS, which is a 96-Bit value.

25 delayAsymmetry values that are larger than the maximum value
26 that can be represented by the TimeInterval data type, i.e.,
27 0x7FFF FFFF FFFF FFFF, (where the units are 2^{-16} ns),

1 used for this managed object are set to this largest value.
2 delayAsymmetry values that are less than the minimum value
3 that can be represented by the TimeInterval data type, i.e.,
4 0x8000 0000 0000 0001 written in twos complement form (where
5 the units are 2^{-16} ns), used for this managed object are
6 set to this smallest value."
7 REFERENCE "14.8.10 and 8.3"
8 ::= { ieee8021AsV3PortDSEntry 12 }
9

10 ieee8021AsV3PortDSNbrRateRatio OBJECT-TYPE
11 SYNTAX Integer32
12 MAX-ACCESS read-only
13 STATUS current
14 DESCRIPTION
15 "The value is an estimate of the ratio of the frequency of
16 the LocalClock entity of the PTP Instance at the other end
17 of the link attached to this PTP Port, to the frequency of
18 the LocalClock entity of this PTP Instance. neighborRateRatio
19 is expressed as the fractional frequency offset multiplied
20 by 2^{41} , i.e., the quantity (neighborRateRatio -1.0)(2^{41})."
21 REFERENCE "14.8.11"
22 ::= { ieee8021AsV3PortDSEntry 13 }

23
24
25 ieee8021AsV3PortDSInitialLogAnnounceInterval OBJECT-TYPE
26 SYNTAX Integer32(-128..127)
27 MAX-ACCESS read-write

1 STATUS current

2 DESCRIPTION

3 "If useMgtSettableLogAnnounceInterval is FALSE (2), the

4 value is the logarithm to base 2 of the announce interval

5 used when (a) the PTP Port is initialized, or (b) a message

6 interval request TLV is received with the logAnnounceInterval

7 field set to 126."

8 REFERENCE "14.8.12"

9 DEFVAL { 0 }

10 ::= { ieee8021AsV3PortDSEntry 14 }

11

12 ieee8021AsV3PortDSCurrentLogAnnounceInterval OBJECT-TYPE

13 SYNTAX Integer32(-128..127)

14 MAX-ACCESS read-only

15 STATUS current

16 DESCRIPTION

17 "The value is the logarithm to the base 2 of the

18 current announce interval."

19 REFERENCE "14.8.13"

20 ::= { ieee8021AsV3PortDSEntry 15 }

21

22 ieee8021AsV3PortDSUseMgtSettableLogAnnounceInterval OBJECT-TYPE

23 SYNTAX TruthValue

24 MAX-ACCESS read-write

25 STATUS current

26 DESCRIPTION

27 "The managed object is a Boolean that determines the

1 source of the announce interval. If the value is TRUE (1),
2 the value of currentLogAnnounceInterval is set equal to the
3 value of mgtSettableLogAnnounceInterval. If the value is
4 FALSE (2), the value of currentLogAnnounceInterval is
5 determined by the AnnounceIntervalSetting state machine. The
6 default value of useMgtSettableLogAnnounceInterval is
7 FALSE (2) for domain 0 and TRUE (1) for domains other than
8 domain 0."

9 REFERENCE "14.8.14"

10 ::= { ieee8021AsV3PortDSEntry 16 }

11

12 ieee8021AsV3PortDSMgtSettableLogAnnounceInterval OBJECT-TYPE

13 SYNTAX Integer32(-128..127)

14 MAX-ACCESS read-write

15 STATUS current

16 DESCRIPTION

17 "The value is the logarithm to base 2 of the announce interval
18 used if useMgtSettableLogAnnounceInterval is TRUE (1).

19 The value is not used if useMgtSettableLogAnnounceInterval is
20 FALSE (2)."

21 REFERENCE "14.8.15"

22 ::= { ieee8021AsV3PortDSEntry 17 }

23

24 ieee8021AsV3PortDSAnnounceReceiptTimeout OBJECT-TYPE

25 SYNTAX Unsigned32(0..255)

26 MAX-ACCESS read-write

27 STATUS current

1 DESCRIPTION

2 "The value is the number of Announce message transmission
3 intervals that a timeReceiver port waits without receiving an
4 Announce message, before assuming that the timeTransmitter is no
5 longer transmitting Announce messages and the BTCA needs
6 to be run, if appropriate."

7 REFERENCE "14.8.16"

8 DEFVAL { 3 }

9 ::= { ieee8021AsV3PortDSEntry 18 }

10

11

12 ieee8021AsV3PortDSInitialLogSyncInterval OBJECT-TYPE

13 SYNTAX Integer32(-128..127)

14 MAX-ACCESS read-write

15 STATUS current

16 DESCRIPTION

17 "If useMgtSettableLogSyncInterval is FALSE (2), the
18 value is the logarithm to base 2 of the sync interval used
19 when (a) the PTP Port is initialized, or (b) a message
20 interval request TLV is received with the logTimeSyncInterval
21 field set to 126."

22 REFERENCE "14.8.17"

23 ::= { ieee8021AsV3PortDSEntry 19 }

24

25 ieee8021AsV3PortDSCurrentLogSyncInterval OBJECT-TYPE

26 SYNTAX Integer32(-128..127)

27 MAX-ACCESS read-only

1 STATUS current

2 DESCRIPTION

3 "The value is the logarithm to the base 2 of the current
4 time-synchronization transmission interval."

5 REFERENCE "14.8.18"

6 ::= { ieee8021AsV3PortDSEntry 20 }

7

8 ieee8021AsV3PortDSUseMgtSettableLogSyncInterval OBJECT-TYPE

9 SYNTAX TruthValue

10 MAX-ACCESS read-write

11 STATUS current

12 DESCRIPTION

13 "The managed object is a Boolean that determines the source
14 of the sync interval. If the value is TRUE (1), the value
15 of currentLogSyncInterval is set equal to the value of
16 mgtSettableLogSyncInterval. If the value of the managed
17 object is FALSE (2), the value of currentLogSyncInterval is
18 determined by the SyncIntervalSetting state machine. The
19 default value of useMgtSettableLogSyncInterval is FALSE (2)
20 for domain 0 and TRUE (1) for domains other than domain 0."

21 REFERENCE "14.8.19"

22 ::= { ieee8021AsV3PortDSEntry 21 }

23

24 ieee8021AsV3PortDSMgtSettableLogSyncInterval OBJECT-TYPE

25 SYNTAX Integer32(-128..127)

26 MAX-ACCESS read-write

27 STATUS current

1 DESCRIPTION

2 "The value is the logarithm to base 2 of the sync interval
3 if useMgtSettableLogSyncInterval is TRUE (1). The value is
4 not used if useMgtSettableLogSyncInterval is FALSE (2)."

5 REFERENCE "14.8.20"

6 ::= { ieee8021AsV3PortDSEntry 22 }

7

8 ieee8021AsV3PortDSSyncReceiptTimeout OBJECT-TYPE

9 SYNTAX Unsigned32(0..255)

10 MAX-ACCESS read-write

11 STATUS current

12 DESCRIPTION

13 "The value is the number of time-synchronization transmission
14 intervals that a timeReceiver port waits without receiving
15 synchronization information, before assuming that the timeTransmitter
16 is no longer transmitting synchronization information and
17 that the BTCA needs to be run, if appropriate."

18 REFERENCE "14.8.21"

19 DEFVAL { 3 }

20 ::= { ieee8021AsV3PortDSEntry 23 }

21

22

23 ieee8021AsV3PortDSSyncReceiptTimeoutTimeInterval OBJECT-TYPE

24 SYNTAX Ieee8021ASV3UScaledNs

25 UNITS "2**-16 ns"

26 MAX-ACCESS read-only

27 STATUS current

1 DESCRIPTION

2 "The value is equal to the value of the per-PTP Port global
3 variable syncReceiptTimeoutTimeInterval. It is the time
4 interval after which sync receipt timeout occurs if
5 time-synchronization information has not been received during
6 the interval."

7 REFERENCE "14.8.22"

8 ::= { ieee8021AsV3PortDSEntry 24 }

9

10 ieee8021AsV3PortDSInitialLogPdelayReqInterval OBJECT-TYPE

11 SYNTAX Integer32(-128..127)

12 MAX-ACCESS read-write

13 STATUS current

14 DESCRIPTION

15 "For full-duplex IEEE 802.3 media and for CSN media that
16 use the peer-to-peer delay mechanism to measure path delay,
17 the value is the logarithm to base 2 of the Pdelay_Req
18 message transmission interval used when (a) the PTP Port is
19 initialized, or (b) a message interval request TLV is
20 received with the logLinkDelayInterval field set to 126.

21 For all other media, the value is 127."

22 REFERENCE "14.8.23"

23 ::= { ieee8021AsV3PortDSEntry 25 }

24

25 ieee8021AsV3PortDSCurrentLogPdelayReqInterval OBJECT-TYPE

26 SYNTAX Integer32(-128..127)

27 MAX-ACCESS read-only

1 STATUS current

2 DESCRIPTION

3 "For full-duplex IEEE 802.3 media and for CSN media that

4 use the peer-to-peer delay mechanism to measure path delay,

5 the value is the logarithm to the base 2 of the current

6 Pdelay_Req message transmission interval.

7 For all other media, the value is 127."

8 REFERENCE "14.8.24"

9 ::= { ieee8021AsV3PortDSEntry 26 }

10

11 ieee8021AsV3PortDSUseMgtSettableLogPdelayReqInterval OBJECT-TYPE

12 SYNTAX TruthValue

13 MAX-ACCESS read-write

14 STATUS current

15 DESCRIPTION

16 "The managed object is a Boolean that determines the source

17 of the mean time interval between successive Pdelay_Req

18 messages. If the value is TRUE (1), the value of

19 currentLogPdelayReqInterval is set equal to the value of

20 mgtSettableLogPdelayReqInterval. If the value of the managed

21 object is FALSE (2), the value of currentLogPdelayReqInterval

22 is determined by the LinkDelayIntervalSetting state machine.

23 The default value of useMgtSettableLogPdelayReqInterval is

24 FALSE (2)."

25 REFERENCE "14.8.25"

26 DEFVAL { false }

27 ::= { ieee8021AsV3PortDSEntry 27 }

1
2 ieee8021AsV3PortDSMgtSettableLogPdelayReqInterval OBJECT-TYPE

3 SYNTAX Integer32 (-128..127)

4 MAX-ACCESS read-write

5 STATUS current

6 DESCRIPTION

7 "The value is the logarithm to base 2 of the mean time

8 interval between successive Pdelay_Req messages if

9 useMgtSettableLogPdelayReqInterval is TRUE (1). The

10 value is not used if useMgtSettableLogPdelayReqInterval

11 is FALSE (2)."

12 REFERENCE "14.8.26"

13 ::= { ieee8021AsV3PortDSEntry 28 }

14

15 ieee8021AsV3PortDSInitialLogGptpCapableMessageInterval OBJECT-TYPE

16 SYNTAX Integer32 (-128..127)

17 MAX-ACCESS read-write

18 STATUS current

19 DESCRIPTION

20 "The value is the logarithm to base 2 of the gPTP capable

21 message interval used when (a) the PTP Port is initialized,

22 or (b) a gPtpCapableMessage interval request TLV is received

23 with the logGptpCapableMessageInterval field set to 126."

24 REFERENCE "14.8.27"

25 ::= { ieee8021AsV3PortDSEntry 29 }

26

27 ieee8021AsV3PortDSCurrentLogGptpCapableMessageInterval OBJECT-TYPE

1 SYNTAX Integer32 (-128..127)

2 MAX-ACCESS read-only

3 STATUS current

4 DESCRIPTION

5 "The value is the logarithm to the base 2 of the current

6 gPTP capable message interval."

7 REFERENCE "14.8.28"

8 ::= { ieee8021AsV3PortDSEntry 30 }

9

10 ieee8021AsV3PortDSUseMgtSettableLogGptpCapableMessageInterval OBJECT-TYPE

11 SYNTAX TruthValue

12 MAX-ACCESS read-write

13 STATUS current

14 DESCRIPTION

15 "The managed object is a Boolean that determines the source

16 of the gPTP capable message interval. If the value is

17 TRUE (1), the value of currentLogGptpCapableMessageInterval

18 is set equal to the value of

19 mgtSettableLogGptpCapableMessageInterval. If the value of

20 the managed object is FALSE (2), the value of

21 currentLogGptpCapableMessageInterval is determined by the

22 GptpCapableMessageIntervalSetting state machine.

23 The default value of

24 useMgtSettableLogGptpCapableMessageInterval is FALSE (2)."

25 REFERENCE "14.8.29"

26 DEFVAL { false }

27 ::= { ieee8021AsV3PortDSEntry 31 }

1

2 ieee8021AsV3PortDSMgtSettableLogGptpCapableMessageInterval OBJECT-TYPE

3 SYNTAX Integer32 (-128..127)

4 MAX-ACCESS read-write

5 STATUS current

6 DESCRIPTION

7 "The value is the logarithm to base 2 of the

8 gPtpCapableMessageInterval if

9 useMgtSettableLogGptpCapableMessageInterval is TRUE (1).

10 The value is not used if

11 useMgtSettableLogGptpCapableMessageInterval is FALSE (2)."

12 REFERENCE "14.8.30"

13 ::= { ieee8021AsV3PortDSEntry 32 }

14

15 ieee8021AsV3PortDSInitialComputeNbrRateRatio OBJECT-TYPE

16 SYNTAX TruthValue

17 MAX-ACCESS read-write

18 STATUS current

19 DESCRIPTION

20 "If useMgtSettableComputeNeighborRateRatio is FALSE (2),

21 then for full-duplex IEEE 802.3 media and for CSN media that

22 use the peer-to-peer delay mechanism to measure path delay,

23 the value is the initial value of computeNeighborRateRatio."

24 REFERENCE "14.8.31"

25 ::= { ieee8021AsV3PortDSEntry 33 }

26

27 ieee8021AsV3PortDSCurrentComputeNbrRateRatio OBJECT-TYPE

1 SYNTAX TruthValue
2 MAX-ACCESS read-only
3 STATUS current
4 DESCRIPTION
5 "For full-duplex IEEE 802.3 media and for CSN media that
6 use the peer-to-peer delay mechanism to measure path delay,
7 the value is the current value of computeNeighborRateRatio."
8 REFERENCE "14.8.32"
9 ::= { ieee8021AsV3PortDSEntry 34 }
10
11 ieee8021AsV3PortDSUseMgtSettableComputeNbrRateRatio OBJECT-TYPE
12 SYNTAX TruthValue
13 MAX-ACCESS read-write
14 STATUS current
15 DESCRIPTION
16 "The managed object is a Boolean that determines the source
17 of the value of computeNeighborRateRatio. If the value is
18 TRUE (1), the value of computeNeighborRateRatio is set equal
19 to the value of mgtSettableComputeNeighborRateRatio. If the
20 value of the managed object is FALSE (2), the value of
21 currentComputeNeighborRateRatio is determined by the
22 LinkDelayIntervalSetting state machine.
23 The default value of useMgtSettableComputeNbrRateRatio is
24 FALSE (2)."
25 REFERENCE "14.8.33"
26 DEFVAL { false }
27 ::= { ieee8021AsV3PortDSEntry 35 }

1

2 ieee8021AsV3PortDSMgtSettableComputeNbrRateRatio OBJECT-TYPE

3 SYNTAX TruthValue

4 MAX-ACCESS read-write

5 STATUS current

6 DESCRIPTION

7 "ComputeNeighborRateRatio is configured to this value if

8 useMgtSettableComputeNeighborRateRatio is TRUE (1). The

9 value is not used if useMgtSettableComputeNeighborRateRatio

10 is FALSE (2)."

11 REFERENCE "14.8.34"

12 ::= { ieee8021AsV3PortDSEntry 36 }

13

14 ieee8021AsV3PortDSInitialComputeMeanLinkDelay OBJECT-TYPE

15 SYNTAX TruthValue

16 MAX-ACCESS read-write

17 STATUS current

18 DESCRIPTION

19 "If useMgtSettableComputeMeanLinkDelay is FALSE (2) then,

20 for full-duplex IEEE 802.3 media and for CSN media that use

21 the peer-to-peer delay mechanism to measure path delay,

22 the value is the initial value of computeMeanLinkDelay."

23 REFERENCE "14.8.35"

24 ::= { ieee8021AsV3PortDSEntry 37 }

25

26 ieee8021AsV3PortDSCurrentComputeMeanLinkDelay OBJECT-TYPE

27 SYNTAX TruthValue

1 MAX-ACCESS read-only
2 STATUS current
3 DESCRIPTION
4 "For full-duplex IEEE 802.3 media and for CSN media that
5 use the peer-to-peer delay mechanism to measure path delay,
6 the value is the current value of computeMeanLinkDelay."
7 REFERENCE "14.8.36"
8 ::= { ieee8021AsV3PortDSEntry 38 }
9
10 ieee8021AsV3PortDSUseMgtSettableComputeMeanLinkDelay OBJECT-TYPE
11 SYNTAX TruthValue
12 MAX-ACCESS read-write
13 STATUS current
14 DESCRIPTION
15 "The managed object is a Boolean that determines the source
16 of the value of computeMeanLinkDelay. If the value is
17 TRUE (1), the value of computeMeanLinkDelay is set equal to
18 the value of mgtSettableComputeMeanLinkDelay. If the value
19 of the managed object is FALSE (2), the value of
20 currentComputeMeanLinkDelay is determined by the
21 LinkDelayIntervalSetting state machine.
22 The default value of useMgtSettableComputeMeanLinkDelay
23 is FALSE (2)."
24 REFERENCE "14.8.37"
25 DEFVAL { false }
26 ::= { ieee8021AsV3PortDSEntry 39 }
27

1 ieee8021AsV3PortDSMgtSettableComputeMeanLinkDelay OBJECT-TYPE
2 SYNTAX TruthValue
3 MAX-ACCESS read-write
4 STATUS current
5 DESCRIPTION
6 "ComputeMeanLinkDelay is configured to this value if
7 useMgtSettableComputeMeanLinkDelay is TRUE (1). The
8 value is not used if useMgtSettableComputeMeanLinkDelay
9 is FALSE (2)."
10 REFERENCE "14.8.38"
11 ::= { ieee8021AsV3PortDSEntry 40 }
12
13 ieee8021AsV3PortDSAllowedLostRsp OBJECT-TYPE
14 SYNTAX Unsigned32(1..255)
15 MAX-ACCESS read-write
16 STATUS current
17 DESCRIPTION
18 "The value is equal to the value of the per-PTP Port global
19 variable allowedLostResponses. It is the number of Pdelay_Req
20 messages without valid responses above which a PTP Port
21 is considered to be not exchanging peer delay messages with
22 its neighbor.
23 Setting this managed object causes the per-PTP Port global
24 variable allowedLostResponses to have the same value."
25 REFERENCE "14.8.39 and 11.5.3"
26 DEFVAL { 9 }
27 ::= { ieee8021AsV3PortDSEntry 41 }

1

2 ieee8021AsV3PortDSAllowedFaults OBJECT-TYPE

3 SYNTAX Unsigned32(1..255)

4 MAX-ACCESS read-write

5 STATUS current

6 DESCRIPTION

7 "The value is equal to the value of the per-PTP-Port global

8 variable allowedFaults. It is the number of faults above

9 which asCapable is set to FALSE (1), i.e., a PTP Port is

10 considered not capable of interoperating with its

11 neighbor via the IEEE 802.1AS protocol.

12 Setting this managed object causes the per-PTP Port global

13 variable allowedFaults to have the same value."

14 REFERENCE "14.8.40"

15 DEFVAL { 9 }

16 ::= { ieee8021AsV3PortDSEntry 42 }

17

18 ieee8021AsV3PortDSGPtpCapableReceiptTimeout OBJECT-TYPE

19 SYNTAX Unsigned32(1..255)

20 MAX-ACCESS read-write

21 STATUS current

22 DESCRIPTION

23 "The value is the number of transmission intervals that a

24 PTP Port waits without receiving the gPTP capable TLV, before

25 assuming that the neighbor PTP Port is no longer invoking

26 the gPTP protocol."

27 REFERENCE "14.8.41"

1 DEFVAL { 9 }

2 ::= { ieee8021AsV3PortDSEntry 43 }

3

4 ieee8021AsV3PortDSVersionNumber OBJECT-TYPE

5 SYNTAX Unsigned32(0..16)

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "This value is set to versionPTP as specified in 10.6.2.2.4."

10 REFERENCE "14.8.42"

11 ::= { ieee8021AsV3PortDSEntry 44 }

12

13 ieee8021AsV3PortDSNup OBJECT-TYPE

14 SYNTAX Float64TC

15 MAX-ACCESS read-write

16 STATUS current

17 DESCRIPTION

18 "For an OLT port of an IEEE 802.3 EPON link, the value is
19 the effective index of refraction for the EPON upstream
20 wavelength light of the optical path. The default value is
21 1.46770 for 1 Gb/s upstream links, and 1.46773 for
22 10 Gb/s upstream links.
23 For all other PTP Ports, the value is 0."

24 REFERENCE "14.8.43"

25 ::= { ieee8021AsV3PortDSEntry 45 }

26

27 ieee8021AsV3PortDSNdown OBJECT-TYPE

1 SYNTAX Float64TC
2 MAX-ACCESS read-write
3 STATUS current
4 DESCRIPTION
5 "For an OLT port of an IEEE 802.3 EPON link, the value is
6 the effective index of refraction for the EPON downstream
7 wavelength light of the optical path. The default value is
8 1.46805 for 1 Gb/s downstream links, and 1.46851 for
9 10 Gb/s downstream links.
10 For all other PTP Ports, the value is 0."
11 REFERENCE "14.8.44"
12 ::= { ieee8021AsV3PortDSEntry 46 }
13
14 ieee8021AsV3PortDSOneStepTxOper OBJECT-TYPE
15 SYNTAX TruthValue
16 MAX-ACCESS read-only
17 STATUS current
18 DESCRIPTION
19 "The value is equal to the value of the per-PTP Port global
20 variable oneStepTxOper. Its value is TRUE (1) if the
21 PTP Port is sending one-step Sync messages, and FALSE (2)
22 if the PTP Port is sending two-step Sync and Follow-Up
23 messages."
24 REFERENCE "14.8.45"
25 ::= { ieee8021AsV3PortDSEntry 47 }
26
27 ieee8021AsV3PortDSOneStepReceive OBJECT-TYPE

1 SYNTAX TruthValue
2 MAX-ACCESS read-only
3 STATUS current
4 DESCRIPTION
5 "The value is equal to the value of the per-PTP Port global
6 variable oneStepReceive. Its value is TRUE (1) if the
7 PTP Port is capable of receiving and processing one-step
8 Sync messages."
9 REFERENCE "14.8.46"
10 ::= { ieee8021AsV3PortDSEntry 48 }
11
12 ieee8021AsV3PortDSOneStepTransmit OBJECT-TYPE
13 SYNTAX TruthValue
14 MAX-ACCESS read-only
15 STATUS current
16 DESCRIPTION
17 "The value is equal to the value of the per-PTP Port global
18 variable oneStepTransmit. Its value is TRUE (1) if the
19 PTP Port is capable of transmitting one-step Sync messages."
20 REFERENCE "14.8.47"
21 ::= { ieee8021AsV3PortDSEntry 49 }
22
23 ieee8021AsV3PortDSInitialOneStepTxOper OBJECT-TYPE
24 SYNTAX TruthValue
25 MAX-ACCESS read-write
26 STATUS current
27 DESCRIPTION

1 "If useMgtSettableOneStepTxOper is FALSE (2), the value is
2 used to initialize currentOneStepTxOper when the PTP Port is
3 initialized. If useMgtSettableOneStepTxOper is TRUE (1),
4 the value of initialOneStepTxOper is not used."

5 REFERENCE "14.8.48"

6 DEFVAL { false }

7 ::= { ieee8021AsV3PortDSEntry 50 }

8

9 ieee8021AsV3PortDSCurrentOneStepTxOper OBJECT-TYPE

10 SYNTAX TruthValue

11 MAX-ACCESS read-only

12 STATUS current

13 DESCRIPTION

14 "The value is TRUE (1) if it is desired, either via
15 management or via a received Signaling message, that the
16 PTP Port transmit one-step Sync messages. The value is
17 FALSE (2) if it is not desired, either via management or via
18 a received Signaling message, that the PTP Port transmit
19 one-step Sync messages.

20 NOTE: The PTP Port will send one-step Sync messages only if
21 currentOneStepTxOper and oneStepTransmit are both TRUE (1)."

22 REFERENCE "14.8.49"

23 ::= { ieee8021AsV3PortDSEntry 51 }

24

25 ieee8021AsV3PortDSUseMgtSettableOneStepTxOper OBJECT-TYPE

26 SYNTAX TruthValue

27 MAX-ACCESS read-write

1 STATUS current

2 DESCRIPTION

3 "The managed object is a Boolean that determines the source

4 of currentOneStepTxOper. If the value is TRUE (1), the

5 value of currentOneStepTxOper is set equal to the value of

6 mgtSettableOneStepTxOper. If the value is FALSE (2), the

7 value of currentOneStepTxOper is determined by the

8 OneStepTxOperSetting state machine.

9 The default value of useMgtSettableOneStepTxOper is TRUE (1)."

10 REFERENCE "14.8.50"

11 DEFVAL { true }

12 ::= { ieee8021AsV3PortDSEntry 52 }

13

14 ieee8021AsV3PortDSMgtSettableOneStepTxOper OBJECT-TYPE

15 SYNTAX TruthValue

16 MAX-ACCESS read-write

17 STATUS current

18 DESCRIPTION

19 "If useMgtSettableOneStepTxOper is TRUE (1),

20 currentOneStepTxOper is set equal to the value of

21 mgtSettableOneStepTxOper. The value of mgtSettableOneStepTxOper

22 is not used if useMgtSettableOneStepTxOper is FALSE (2).

23 The default value of mgtSettableOneStepTxOper is FALSE (2)

24 for domains other than domain 0."

25 REFERENCE "14.8.51"

26 ::= { ieee8021AsV3PortDSEntry 53 }

27

1 ieee8021AsV3PortDSSyncLocked OBJECT-TYPE
2 SYNTAX TruthValue
3 MAX-ACCESS read-only
4 STATUS current
5 DESCRIPTION
6 "The value is equal to the value of the per-PTP Port global
7 variable syncLocked. Its value is TRUE (1) if the PTP Port
8 will transmit a Sync as soon as possible after the timeReceiver
9 PTP Port receives a Sync."
10 REFERENCE "14.8.52"
11 ::= { ieee8021AsV3PortDSEntry 54 }
12
13 ieee8021AsV3PortDSPdelayTruncTST1 OBJECT-TYPE
14 SYNTAX Ieee8021ASV3Timestamp
15 MAX-ACCESS read-only
16 STATUS current
17 DESCRIPTION
18 "For full-duplex IEEE 802.3 media and for CSN media that use
19 the peer-to-peer delay mechanism to measure path delay, the
20 first value, T1, of the four elements of this array is as
21 described in Table 14-9. For all other media, the values are
22 zero. This object corresponds to the timestamp t1 in
23 Figure 11-1, and expressed in units of 2^-16 ns (i.e., the
24 value of this array element is equal to the remainder obtained
25 upon dividing the respective timestamp , expressed in units
26 of 2^-16 ns, by 2^48).
27 At any given time, the timestamp values stored in the T1, T2,

1 T3, T4 PdelayTruncTS are for the same, and most recently
2 completed, peer delay message exchange."

3 REFERENCE "14.8.53"

4 ::= { ieee8021AsV3PortDSEntry 55 }

5

6 ieee8021AsV3PortDSPdelayTruncTST2 OBJECT-TYPE

7 SYNTAX Ieee8021ASV3Timestamp

8 MAX-ACCESS read-only

9 STATUS current

10 DESCRIPTION

11 "For full-duplex IEEE 802.3 media and for CSN media that use

12 the peer-to-peer delay mechanism to measure path delay, the

13 second value, T2, of the four elements of this array is as

14 described in Table 14-9. For all other media, the values are

15 zero. This object corresponds to the timestamp t2 in

16 Figure 11-1, and expressed in units of 2^{16} ns (i.e., the

17 value of this array element is equal to the remainder obtained

18 upon dividing the respective timestamp , expressed in units

19 of 2^{16} ns, by 2^{48}).

20 At any given time, the timestamp values stored in the T1, T2,

21 T3, T4 PdelayTruncTS are for the same, and most recently

22 completed, peer delay message exchange."

23 REFERENCE "14.8.53"

24 ::= { ieee8021AsV3PortDSEntry 56 }

25

26 ieee8021AsV3PortDSPdelayTruncTST3 OBJECT-TYPE

27 SYNTAX Ieee8021ASV3Timestamp

1 MAX-ACCESS read-only

2 STATUS current

3 DESCRIPTION

4 "For full-duplex IEEE 802.3 media and for CSN media that use

5 the peer-to-peer delay mechanism to measure path delay, the

6 third value, T3, of the four elements of this array is as

7 described in Table 14-9. For all other media, the values are

8 zero. This object corresponds to the timestamp t3 in

9 Figure 11-1, and expressed in units of 2^{-16} ns (i.e., the

10 value of this array element is equal to the remainder obtained

11 upon dividing the respective timestamp , expressed in units

12 of 2^{-16} ns, by 2^{48}).

13 At any given time, the timestamp values stored in the T1, T2,

14 T3, T4 PdelayTruncTS are for the same, and most recently

15 completed, peer delay message exchange."

16 REFERENCE "14.8.53"

17 ::= { ieee8021AsV3PortDSEntry 57 }

18

19 ieee8021AsV3PortDSPdelayTruncTST4 OBJECT-TYPE

20 SYNTAX Ieee8021ASV3Timestamp

21 MAX-ACCESS read-only

22 STATUS current

23 DESCRIPTION

24 "For full-duplex IEEE 802.3 media and for CSN media that use

25 the peer-to-peer delay mechanism to measure path delay, the

26 fourth value, T4, of the four elements of this array is as

27 described in Table 14-9. For all other media, the values are

1 zero. This object corresponds to the timestamp t4 in
2 Figure 11-1, and expressed in units of 2^{16} ns (i.e., the
3 value of this array element is equal to the remainder obtained
4 upon dividing the respective timestamp , expressed in units
5 of 2^{16} ns, by 2^{48}).

6 At any given time, the timestamp values stored in the T1, T2,
7 T3, T4 PdelayTruncTS are for the same, and most recently
8 completed, peer delay message exchange."

9 REFERENCE "14.8.53"

10 ::= { ieee8021AsV3PortDSEntry 58 }

11

12 ieee8021AsV3PortDSMinorVersionNumber OBJECT-TYPE

13 SYNTAX Unsigned32 (0..15)

14 MAX-ACCESS read-only

15 STATUS current

16 DESCRIPTION

17 "This value is set to minorVersionPTP as specified in 10.6.2.2.3."

18 REFERENCE "14.8.54"

19 ::= { ieee8021AsV3PortDSEntry 59 }

20

21 -- =====

22 -- The Description Port Parameter Data Set contains the

23 -- profileIdentifier for this PTP profile, as specified in F.2

24 -- =====

25

26 ieee8021AsV3DescriptionPortDSTable OBJECT-TYPE

27 SYNTAX SEQUENCE OF Ieee8021AsV3DescriptionPortDSEntry

1 MAX-ACCESS not-accessible
2 STATUS current
3 DESCRIPTION
4 "The descriptionPortDS contains the profileIdentifier for
5 this PTP profile, as specified in F.2."
6 REFERENCE "14.9"
7 ::= { ieee8021AsV3MIBObjects 11 }
8
9 ieee8021AsV3DescriptionPortDSEntry OBJECT-TYPE
10 SYNTAX Ieee8021AsV3DescriptionPortDSEntry
11 MAX-ACCESS not-accessible
12 STATUS current
13 DESCRIPTION
14 "The descriptionPortDS contains the profileIdentifier for
15 this PTP profile"
16 INDEX { ieee8021AsV3PtpInstance,
17 ieee8021AsV3DescriptionPortDSAsIndex }
18 ::= { ieee8021AsV3DescriptionPortDSTable 1 }
19
20 Ieee8021AsV3DescriptionPortDSEntry ::=
21 SEQUENCE {
22 ieee8021AsV3DescriptionPortDSAsIndex
23 InterfaceIndexOrZero,
24 ieee8021AsV3DescriptionPortDSPProfileIdentifier
25 Ieee8021AsV3GPtpProfileIdentifier }
26
27 ieee8021AsV3DescriptionPortDSAsIndex OBJECT-TYPE

1 SYNTAX InterfaceIndexOrZero

2 MAX-ACCESS not-accessible

3 STATUS current

4 DESCRIPTION

5 "This object identifies the gPTP interface group within

6 the system for which this entry contains information. It

7 is the value of the instance of the IfIndex object,

8 defined in the IF-MIB, for the gPTP interface group

9 corresponding to this port, or the value 0 if the port

10 has not been bound to an underlying frame source and

11 sink.

12

13 For a given media port of a Bridge or an end station,

14 there can be one or more PTP Port, and depends whether

15 a media port supports point to point link (e.g. IEEE

16 802.3 Ethernet) or point to multi-point (e.g. CSN, IEEE

17 802.3 EPON) links on the media port."

18 REFERENCE "IEEE Std 802.1AS Description Port Parameter DS Group

19 PTP Port Index"

20 ::= { ieee8021AsV3DescriptionPortDSEntry 1 }

21

22 ieee8021AsV3DescriptionPortDSProfileIdentifier OBJECT-TYPE

23 SYNTAX Ieee8021AsV3GPtpProfileIdentifier

24 MAX-ACCESS read-only

25 STATUS current

26 DESCRIPTION

27 "The value is the profileIdentifier for this PTP profile."

1 REFERENCE "14.9.2 and F.2"
2 ::= { ieee8021AsV3DescriptionPortDSEntry 2 }
3
4 -- ======
5 -- The Port Parameter Statistics Data Set provides counters
6 -- associated with PTP Port capabilities at a given PTP Instance.
7 -- ======
8
9 ieee8021AsV3PortStatDSTable OBJECT-TYPE
10 SYNTAX SEQUENCE OF Ieee8021AsV3PortStatDSEntry
11 MAX-ACCESS not-accessible
12 STATUS current
13 DESCRIPTION
14 "The portStatisticsDS provides counters associated with PTP Port
15 capabilities at a given PTP Instance."
16 REFERENCE "14.10"
17 ::= { ieee8021AsV3MIBObjects 12 }
18
19 ieee8021AsV3PortStatDSEntry OBJECT-TYPE
20 SYNTAX Ieee8021AsV3PortStatDSEntry
21 MAX-ACCESS not-accessible
22 STATUS current
23 DESCRIPTION
24 "Port Statistics Data Set provides counters associated with
25 PTP Port capabilities at a given PTP Instance."
26 INDEX { ieee8021AsV3PtpInstance,
27 ieee8021AsV3PortDSIndex }

```

1 ::= { ieee8021AsV3PortStatDSTable 1 }

2

3 Ieee8021AsV3PortStatDSEntry ::=

4 SEQUENCE {

5   ieee8021AsV3PortStatRxSyncCount      Counter32,
6   ieee8021AsV3PortStatRxOneStepSyncCount Counter32,
7   ieee8021AsV3PortStatRxFollowUpCount    Counter32,
8   ieee8021AsV3PortStatRxPdelayRequestCount Counter32,
9   ieee8021AsV3PortStatRxPdelayRspCount   Counter32,
10  ieee8021AsV3PortStatRxPdelayRspFollowUpCount Counter32,
11  ieee8021AsV3PortStatRxAnnounceCount    Counter32,
12  ieee8021AsV3PortStatRxPtpPacketDiscardCount Counter32,
13  ieee8021AsV3PortStatSyncReceiptTimeoutCount Counter32,
14  ieee8021AsV3PortStatAnnounceReceiptTimeoutCount Counter32,
15  ieee8021AsV3PortStatPdelayAllowedLostRspExceededCount Counter32,
16  ieee8021AsV3PortStatTxSyncCount       Counter32,
17  ieee8021AsV3PortStatTxOneStepSyncCount Counter32,
18  ieee8021AsV3PortStatTxFollowUpCount   Counter32,
19  ieee8021AsV3PortStatTxPdelayRequestCount Counter32,
20  ieee8021AsV3PortStatTxPdelayRspCount   Counter32,
21  ieee8021AsV3PortStatTxPdelayRspFollowUpCount Counter32,
22  ieee8021AsV3PortStatTxAnnounceCount   Counter32

23 }

24

25 ieee8021AsV3PortStatRxSyncCount OBJECT-TYPE

26   SYNTAX   Counter32

27   MAX-ACCESS read-only

```

1 STATUS current

2 DESCRIPTION

3 "A counter that increments every time synchronization
4 information is received."

5 REFERENCE "14.10.2"

6 ::= { ieee8021AsV3PortStatDSEntry 1 }

7

8 ieee8021AsV3PortStatRxOneStepSyncCount OBJECT-TYPE

9 SYNTAX Counter32

10 MAX-ACCESS read-only

11 STATUS current

12 DESCRIPTION

13 "A counter that increments every time a one-step Sync
14 message is received."

15 REFERENCE "14.10.3"

16 ::= { ieee8021AsV3PortStatDSEntry 2 }

17

18 ieee8021AsV3PortStatRxFollowUpCount OBJECT-TYPE

19 SYNTAX Counter32

20 MAX-ACCESS read-only

21 STATUS current

22 DESCRIPTION

23 "A counter that increments every time a Follow_Up message
24 is received."

25 REFERENCE "14.10.4"

26 ::= { ieee8021AsV3PortStatDSEntry 3 }

27

1 ieee8021AsV3PortStatRxPdelayRequestCount OBJECT-TYPE

2 SYNTAX Counter32

3 MAX-ACCESS read-only

4 STATUS current

5 DESCRIPTION

6 "A counter that increments every time a Pdelay_Req message
7 is received."

8 REFERENCE "14.10.5"

9 ::= { ieee8021AsV3PortStatDSEntry 4 }

10

11 ieee8021AsV3PortStatRxPdelayRspCount OBJECT-TYPE

12 SYNTAX Counter32

13 MAX-ACCESS read-only

14 STATUS current

15 DESCRIPTION

16 "A counter that increments every time a Pdelay_Resp message
17 is received."

18 REFERENCE "14.10.6"

19 ::= { ieee8021AsV3PortStatDSEntry 5 }

20

21 ieee8021AsV3PortStatRxPdelayRspFollowUpCount OBJECT-TYPE

22 SYNTAX Counter32

23 MAX-ACCESS read-only

24 STATUS current

25 DESCRIPTION

26 "A counter that increments every time a Pdelay_Resp_Follow_Up
27 message is received."

1 REFERENCE "14.10.7"

2 ::= { ieee8021AsV3PortStatDSEntry 6 }

3

4 ieee8021AsV3PortStatRxAnnounceCount OBJECT-TYPE

5 SYNTAX Counter32

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "A counter that increments every time an Announce message

10 is received."

11 REFERENCE "14.10.8"

12 ::= { ieee8021AsV3PortStatDSEntry 7 }

13

14 ieee8021AsV3PortStatRxPtpPacketDiscardCount OBJECT-TYPE

15 SYNTAX Counter32

16 MAX-ACCESS read-only

17 STATUS current

18 DESCRIPTION

19 "A counter that increments every time a PTP message of the

20 respective PTP Instance is discarded."

21 REFERENCE "14.10.9"

22 ::= { ieee8021AsV3PortStatDSEntry 8 }

23

24 ieee8021AsV3PortStatSyncReceiptTimeoutCount OBJECT-TYPE

25 SYNTAX Counter32

26 MAX-ACCESS read-only

27 STATUS current

1 DESCRIPTION

2 "A counter that increments every time sync receipt timeout

3 occurs."

4 REFERENCE "14.10.10"

5 ::= { ieee8021AsV3PortStatDSEntry 9 }

6

7 ieee8021AsV3PortStatAnnounceReceiptTimeoutCount OBJECT-TYPE

8 SYNTAX Counter32

9 MAX-ACCESS read-only

10 STATUS current

11 DESCRIPTION

12 "A counter that increments every time announce receipt timeout

13 occurs."

14 REFERENCE "14.10.11"

15 ::= { ieee8021AsV3PortStatDSEntry 10 }

16

17 ieee8021AsV3PortStatPdelayAllowedLostRspExceededCount OBJECT-TYPE

18 SYNTAX Counter32

19 MAX-ACCESS read-only

20 STATUS current

21 DESCRIPTION

22 "A counter that increments every time the value of the

23 variable lostResponses exceeds the value of the variable

24 allowedLostResponses."

25 REFERENCE "14.10.12"

26 ::= { ieee8021AsV3PortStatDSEntry 11 }

27

```
1 ieee8021AsV3PortStatTxSyncCount OBJECT-TYPE
2   SYNTAX  Counter32
3   MAX-ACCESS read-only
4   STATUS   current
5   DESCRIPTION
6     "A counter that increments every time synchronization
7     information is transmitted."
8   REFERENCE "14.10.13"
9 ::= { ieee8021AsV3PortStatDSEntry 12 }
10
11 ieee8021AsV3PortStatTxOneStepSyncCount OBJECT-TYPE
12   SYNTAX  Counter32
13   MAX-ACCESS read-only
14   STATUS   current
15   DESCRIPTION
16     "A counter that increments every time a one-step Sync
17     message is transmitted."
18   REFERENCE "14.10.14"
19 ::= { ieee8021AsV3PortStatDSEntry 13 }
20
21 ieee8021AsV3PortStatTxFollowUpCount OBJECT-TYPE
22   SYNTAX  Counter32
23   MAX-ACCESS read-only
24   STATUS   current
25   DESCRIPTION
26     "A counter that increments every time a Follow_Up message
27     is transmitted."
```

1 REFERENCE "14.10.15"

2 ::= { ieee8021AsV3PortStatDSEntry 14 }

3

4 ieee8021AsV3PortStatTxPdelayRequestCount OBJECT-TYPE

5 SYNTAX Counter32

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "A counter that increments every time a Pdelay_Req message

10 is transmitted."

11 REFERENCE "14.10.16"

12 ::= { ieee8021AsV3PortStatDSEntry 15 }

13

14 ieee8021AsV3PortStatTxPdelayRspCount OBJECT-TYPE

15 SYNTAX Counter32

16 MAX-ACCESS read-only

17 STATUS current

18 DESCRIPTION

19 "A counter that increments every time a Pdelay_Resp message

20 is transmitted."

21 REFERENCE "14.10.17"

22 ::= { ieee8021AsV3PortStatDSEntry 16 }

23

24 ieee8021AsV3PortStatTxPdelayRspFollowUpCount OBJECT-TYPE

25 SYNTAX Counter32

26 MAX-ACCESS read-only

27 STATUS current

1 DESCRIPTION

2 "A counter that increments every time a
3 Pdelay_Resp_Follow_Up message is transmitted."

4 REFERENCE "14.10.18"

5 ::= { ieee8021AsV3PortStatDSEntry 17 }

6

7 ieee8021AsV3PortStatTxAnnounceCount OBJECT-TYPE

8 SYNTAX Counter32

9 MAX-ACCESS read-only

10 STATUS current

11 DESCRIPTION

12 "A counter that increments every time an Announce message is
13 transmitted."

14 REFERENCE "14.10.19"

15 ::= { ieee8021AsV3PortStatDSEntry 18 }

16

17 -- =====

18 -- The Acceptable TimeTransmitter Port Parameter Data Ser represents the

19 -- capability to enable/disable the acceptable timeTransmitter table

20 -- feature on a PTP Port.

21 -- =====

22

23 ieee8021AsV3AcceptableTimeTransmitterPortDSTable OBJECT-TYPE

24 SYNTAX SEQUENCE OF Ieee8021AsV3AcceptableTimeTransmitterPortDSEntry

25 MAX-ACCESS not-accessible

26 STATUS current

27 DESCRIPTION

1 "For the single PTP Port of a PTP End Instance and for each
2 PTP Port of a PTP Relay Instance, the acceptableTimeTransmitterPortDS
3 contains the single member acceptableTimeTransmitterTableEnabled, which
4 is used to enable/disable the Acceptable TimeTransmitter Table Feature.
5 The number of such data sets is the same as the value of
6 defaultDS.numberPorts."
7 REFERENCE "14.11"
8 ::= { ieee8021AsV3MIBObjects 13 }
9
10 ieee8021AsV3AcceptableTimeTransmitterPortDSEntry OBJECT-TYPE
11 SYNTAX Ieee8021AsV3AcceptableTimeTransmitterPortDSEntry
12 MAX-ACCESS not-accessible
13 STATUS current
14 DESCRIPTION
15 "The Acceptable TimeTransmitter Port Data Set represents the capability
16 to enable/disable the acceptable timeTransmitter table feature on a
17 PTP Port.
18 For the single PTP Port of a PTP End Instance and for each
19 PTP Port of a PTP Relay Instance, the acceptableTimeTransmitterPortDS
20 contains the single member acceptableTimeTransmitterTableEnabled, which
21 is used to enable/disable the Acceptable TimeTransmitter Table Feature.
22 The number of such data sets is the same as the value of
23 defaultDS.numberPorts."
24 INDEX { ieee8021AsV3PtpInstance,
25 ieee8021AsV3AcceptableTimeTransmitterPortDSAsIndex }
26 ::= { ieee8021AsV3AcceptableTimeTransmitterPortDSTable 1 }
27

```
1 Ieee8021AsV3AcceptableTimeTransmitterPortDSEntry ::=  
2 SEQUENCE {  
3   ieee8021AsV3AcceptableTimeTransmitterPortDSAsIndex  InterfaceIndexOrZero,  
4   ieee8021AsV3AcceptableTTPortDSAcceptableTTTableEnabled    TruthValue  
5 }  
6  
7 ieee8021AsV3AcceptableTimeTransmitterPortDSAsIndex OBJECT-TYPE  
8   SYNTAX  InterfaceIndexOrZero  
9   MAX-ACCESS not-accessible  
10  STATUS   current  
11  DESCRIPTION  
12    "An index to identify an entry in the Acceptable Time Transmitter  
13      Port Table Data Set."  
14  REFERENCE  "14.11"  
15 ::= { ieee8021AsV3AcceptableTimeTransmitterPortDSEntry 1 }  
16  
17 ieee8021AsV3AcceptableTTPortDSAcceptableTTTableEnabled OBJECT-TYPE  
18   SYNTAX  TruthValue  
19   MAX-ACCESS read-write  
20   STATUS   current  
21  DESCRIPTION  
22    "The value is equal to the value of the Boolean  
23      acceptableTimeTransmitterTableEnabled."  
24  REFERENCE  "14.11.2"  
25 ::= { ieee8021AsV3AcceptableTimeTransmitterPortDSEntry 2 }  
26  
27 -- =====
```

1 -- The External Port Configuration Port Data Set is used with

2 -- the external port configuration option to indicate the

3 -- desired state for the PTP Port.

4 -- =====

5 ieee8021AsV3ExternalPortConfigurationPortDSTable OBJECT-TYPE

6 SYNTAX SEQUENCE OF Ieee8021AsV3ExternalPortConfigurationPortDSEntry

7 MAX-ACCESS not-accessible

8 STATUS current

9 DESCRIPTION

10 "The externalPortConfigurationPortDS contains the single member

11 desiredState, which indicates the desired state for the PTP Port.

12 The number of such data sets is the same as the value of

13 defaultDS.numberPorts."

14 REFERENCE "14.12"

15 ::= { ieee8021AsV3MIBObjects 14 }

16

17 ieee8021AsV3ExternalPortConfigurationPortDSEntry OBJECT-TYPE

18 SYNTAX Ieee8021AsV3ExternalPortConfigurationPortDSEntry

19 MAX-ACCESS not-accessible

20 STATUS current

21 DESCRIPTION

22 "The externalPortConfigurationPortDS contains the single member

23 desiredState, which indicates the desired state for the PTP Port.

24 The number of such data sets is the same as the value of

25 defaultDS.numberPorts."

26 INDEX { ieee8021AsV3PtpInstance,

27 ieee8021AsV3ExternalPortConfigurationPortDSAsIndex }

```
1 ::= { ieee8021AsV3ExternalPortConfigurationPortDSTable 1 }

2

3 Ieee8021AsV3ExternalPortConfigurationPortDSEntry ::=

4 SEQUENCE {

5   ieee8021AsV3ExternalPortConfigurationPortDSAsIndex  InterfaceIndexOrZero,
6   ieee8021AsV3ExternalPortConfigurationPortDSDesiredState      INTEGER
7 }

8

9 ieee8021AsV3ExternalPortConfigurationPortDSAsIndex OBJECT-TYPE

10 SYNTAX  InterfaceIndexOrZero

11 MAX-ACCESS not-accessible

12 STATUS    current

13 DESCRIPTION

14 "An index to identify an entry in the External Port
15 Configuration Port Table Data Set."
16 REFERENCE "14.12"

17 ::= { ieee8021AsV3ExternalPortConfigurationPortDSEntry 1 }

18

19 ieee8021AsV3ExternalPortConfigurationPortDSDesiredState OBJECT-TYPE

20 SYNTAX      INTEGER {

21   disabledPort(3),
22   timeTransmitterPort(6),
23   passivePort(7),
24   timeReceiverPort(9)
25 }

26 MAX-ACCESS read-write

27 STATUS    current
```

1 DESCRIPTION

2 "When the value of defaultDS.externalPortConfigurationEnabled
3 is TRUE (1), the value of
4 externalPortConfigurationPortDS.desiredState is the desired
5 state of the PTP Port. This member sets the value of the
6 variable portStateInd. When a new value is written to the
7 member by management, the variable rcvdPortStateInd is set
8 to TRUE (1)."

9 REFERENCE "14.12.2"

10 ::= { ieee8021AsV3ExternalPortConfigurationPortDSEntry 2 }

11

12 -- =====

13 -- Asymmetry Measurement Mode Parameter Data Set

14 -- to enable/disable the feature on a PTP Port.

15 -- =====

16

17 ieee8021AsV3AsymMeasurementModeDSTable OBJECT-TYPE

18 SYNTAX SEQUENCE OF Ieee8021AsV3AsymMeasurementModeDSEntry

19 MAX-ACCESS not-accessible

20 STATUS current

21 DESCRIPTION

22 "The asymmetryMeasurementModeDS represents the capability to
23 enable/disable the Asymmetry Compensation Measurement Procedure
24 on a PTP Port (see Annex G). This data set is used instead of
25 the cmldsAsymmetryMeasurementModeDS, when only domain 0 is
26 present and CMLDS is not used."

27 REFERENCE "14.13"

```
1 ::= { ieee8021AsV3MIBObjects 15 }

2

3 ieee8021AsV3AsymMeasurementModeDSEntry OBJECT-TYPE

4 SYNTAX Ieee8021AsV3AsymMeasurementModeDSEntry
5 MAX-ACCESS not-accessible
6 STATUS current
7 DESCRIPTION
8 "The asymmetryMeasurementModeDS represents the capability to
9 enable/disable the Asymmetry Compensation Measurement Procedure
10 on a PTP Port (see Annex G). This data set is used instead of
11 the cmldsAsymmetryMeasurementModeDS, when only domain 0 is
12 present and CMLDS is not used. "
13 INDEX { ieee8021AsV3PtpInstance,
14         ieee8021AsV3AsymMeasurementModeDSAsIndex }
15 ::= { ieee8021AsV3AsymMeasurementModeDSTable 1 }

16

17 Ieee8021AsV3AsymMeasurementModeDSEntry ::=

18 SEQUENCE {
19     ieee8021AsV3AsymMeasurementModeDSAsIndex    InterfaceIndexOrZero,
20     ieee8021AsV3AsymMeasurementModeDSAsymMeasurementMode    TruthValue
21 }

22

23 ieee8021AsV3AsymMeasurementModeDSAsIndex OBJECT-TYPE

24 SYNTAX InterfaceIndexOrZero
25 MAX-ACCESS not-accessible
26 STATUS current
27 DESCRIPTION
```

1 "An index to identify an entry in the Asymmetry Measurement
2 Mode Data Set."
3 REFERENCE "14.13"
4 ::= { ieee8021AsV3AsymMeasurementModeDSEntry 1 }
5
6 ieee8021AsV3AsymMeasurementModeDSAsymMeasurementMode OBJECT-TYPE
7 SYNTAX TruthValue
8 MAX-ACCESS read-write
9 STATUS current
10 DESCRIPTION
11 "The value is equal to the value of the Boolean
12 asymmetryMeasurementMode. For full-duplex IEEE 802.3
13 media, the value is TRUE (1) if an asymmetry measurement
14 is being performed for the link attached to this PTP Port,
15 and FALSE (2) otherwise. For all other media, the value
16 shall be FALSE (2). Setting this managed object causes the
17 Boolean asymmetryMeasurementMode to have the same value.
18 NOTE: If an asymmetry measurement is being performed for a
19 link, asymmetryMeasurementMode must be TRUE (1) for the
20 PTP Ports at each end of the link."
21 REFERENCE "14.13.2"
22 ::= { ieee8021AsV3AsymMeasurementModeDSEntry 2 }
23
24 -- ======
25 -- The Common Services Port Parameter Data Set enables a
26 -- PTP Port of a PTP Instance to determine which port of the
27 -- respective common service corresponds to that PTP Port.

1 -- =====

2 ieee8021AsV3CommonServicesPortDSTable OBJECT-TYPE

3 SYNTAX SEQUENCE OF Ieee8021AsV3CommonServicesPortDSEntry

4 MAX-ACCESS not-accessible

5 STATUS current

6 DESCRIPTION

7 "At present, the only common service specified is the CMLDS, and"

8 "the only member of the commonServicesPortDS is the"

9 "cmldsLinkPortPortNumber. This member contains the port number"

10 "of the CMLDS Link Port that corresponds to this PTP Port."

11 REFERENCE "14.14"

12 ::= { ieee8021AsV3MIBObjects 16 }

13

14 ieee8021AsV3CommonServicesPortDSEntry OBJECT-TYPE

15 SYNTAX Ieee8021AsV3CommonServicesPortDSEntry

16 MAX-ACCESS not-accessible

17 STATUS current

18 DESCRIPTION

19 "At present, the only common service specified is the CMLDS, and"

20 "the only member of the commonServicesPortDS is the"

21 "cmldsLinkPortPortNumber. This member contains the port number"

22 "of the CMLDS Link Port that"

23 "corresponds to this PTP Port."

24 INDEX { ieee8021AsV3PtpInstance,

25 ieee8021AsV3CommonServicesPortDSAsIndex }

26 ::= { ieee8021AsV3CommonServicesPortDSTable 1 }

27

```
1 Ieee8021AsV3CommonServicesPortDSEntry ::=  
2   SEQUENCE {  
3     ieee8021AsV3CommonServicesPortDSAsIndex    InterfaceIndexOrZero,  
4     ieee8021AsV3CommonServicesPortDSCmlsLinkPortPortNumber      Unsigned32  
5   }  
6  
7 ieee8021AsV3CommonServicesPortDSAsIndex OBJECT-TYPE  
8   SYNTAX    InterfaceIndexOrZero  
9   MAX-ACCESS not-accessible  
10  STATUS    current  
11  DESCRIPTION  
12    "An index to identify an entry in the Common Services Port  
13    Data Set."  
14  REFERENCE  "14.14"  
15  ::= { ieee8021AsV3CommonServicesPortDSEntry 1 }  
16  
17 ieee8021AsV3CommonServicesPortDSCmlsLinkPortPortNumber OBJECT-TYPE  
18  SYNTAX      Unsigned32 (0..65535)  
19  MAX-ACCESS  read-only  
20  STATUS      current  
21  DESCRIPTION  
22    "The value is the portNumber attribute of the  
23    cmlsLinkPortDS.portIdentity of the Link Port that  
24    corresponds to this PTP Port."  
25  REFERENCE  "14.14.2"  
26  ::= { ieee8021AsV3CommonServicesPortDSEntry 2 }  
27
```

1 -- =====

2 -- The Common Mean Link Delay Service Default Parameter Data Set

3 -- describes the per-time-aware-system attributes of the Common

4 -- Mean Link Delay Service.

5 -- =====

6

7 ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSTable OBJECT-TYPE

8 SYNTAX SEQUENCE OF Ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry

9 MAX-ACCESS not-accessible

10 STATUS current

11 DESCRIPTION

12 "The cmldsDefaultDS describes the per-time-aware-system attributes

13 of the Common Mean Link Delay Service."

14 REFERENCE "14.15"

15 ::= { ieee8021AsV3MIBObjects 17 }

16

17 ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry OBJECT-TYPE

18 SYNTAX Ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry

19 MAX-ACCESS not-accessible

20 STATUS current

21 DESCRIPTION

22 "The cmldsDefaultDS describes the per-time-aware-system attributes

23 of the Common Mean Link Delay Service."

24 INDEX { ieee8021AsV3CmldsDefaultDSAsIndex }

25 ::= { ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSTable 1 }

26

27 Ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry ::=

```
1 SEQUENCE {  
2     ieee8021AsV3CmldsDefaultDSAsIndex InterfaceIndexOrZero,  
3     ieee8021AsV3CmldsDefaultDSClockIdentity Ieee8021AsV3ClockIdentity,  
4     ieee8021AsV3CmldsDefaultDSNumberLinkPorts Unsigned32  
5 }  
6  
7  
8 ieee8021AsV3CmldsDefaultDSAsIndex OBJECT-TYPE  
9     SYNTAX    InterfaceIndexOrZero  
10    MAX-ACCESS not-accessible  
11    STATUS    current  
12    DESCRIPTION  
13        "An index to identify an entry in the Common Mean Link  
14        Delay Default Data Set."  
15    REFERENCE "14.15"  
16 ::= { ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry 1 }  
17  
18 ieee8021AsV3CmldsDefaultDSClockIdentity OBJECT-TYPE  
19     SYNTAX    Ieee8021AsV3ClockIdentity  
20     MAX-ACCESS read-only  
21     STATUS    current  
22     DESCRIPTION  
23        "The value is the clockIdentity that will be used to  
24        identify the Common Mean Link Delay Service."  
25    REFERENCE "14.15.2"  
26 ::= { ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry 2 }  
27
```

1 ieee8021AsV3CmldsDefaultDSNumberLinkPorts OBJECT-TYPE
2 SYNTAX Unsigned32 (0..65535)
3 MAX-ACCESS read-only
4 STATUS current
5 DESCRIPTION
6 "The value is the number of Link Ports of the time-aware
7 system on which the Common Mean Link Delay Service is
8 implemented. For an end station the value is 1."
9 REFERENCE "14.15.3"
10 ::= { ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSEntry 3 }
11
12 -- ======
13 -- The Common Mean Link Delay Service Link Port Parameter Data Set
14 -- represents time-aware Link Port capabilities for the Common Mean
15 -- Link Delay Service of a Link Port of a time-aware system.
16 -- ======
17
18 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSTable OBJECT-TYPE
19 SYNTAX SEQUENCE OF Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry
20 MAX-ACCESS not-accessible
21 STATUS current
22 DESCRIPTION
23 "For every Link Port of the Common Mean Link Delay Service of a
24 time-aware system, the cmldsLinkPortDS is maintained as the
25 basis for making protocol decisions and providing values for
26 message fields. The number of such data sets is the same as
27 the value of cmldsDefaultDS.numberLinkPorts."

1 REFERENCE "14.16"

2 ::= { ieee8021AsV3MIBObjects 18 }

3

4 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry OBJECT-TYPE

5 SYNTAX Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry

6 MAX-ACCESS not-accessible

7 STATUS current

8 DESCRIPTION

9 "For every Link Port of the Common Mean Link Delay Service of a

10 time-aware system, the cmldsLinkPortDS is maintained as the

11 basis for making protocol decisions and providing values for

12 message fields. The number of such data sets is the same as

13 the value of cmldsDefaultDS.numberLinkPorts."

14 INDEX { ieee8021AsV3BridgeBasePort,

15 ieee8021AsV3CmldsLinkPortDSAsIndex }

16 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSTable 1 }

17

18 Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry ::=

19 SEQUENCE {

20 ieee8021AsV3CmldsLinkPortDSAsIndex InterfaceIndexOrZero,

21 ieee8021AsV3CmldsLinkPortDSClockIdentity Ieee8021AsV3ClockIdentity,

22 ieee8021AsV3CmldsLinkPortDSPortNumber Unsigned32,

23 ieee8021AsV3CmldsLinkPortDSCmldsLinkPortEnabled TruthValue,

24 ieee8021AsV3CmldsLinkPortDSIsMeasuringDelay TruthValue,

25 ieee8021AsV3CmldsLinkPortDSAsCapableAcrossDomains TruthValue,

26 ieee8021AsV3CmldsLinkPortDSMeanLinkDelay Ieee8021ASV3PtpTimeInterval,

27 ieee8021AsV3CmldsLinkPortDSMeanLinkDelayThresh Ieee8021ASV3PtpTimeInterval,

```
1    ieee8021AsV3CmlsLinkPortDSDelayAsym      Ieee8021ASV3PtpTimeInterval,  
2    ieee8021AsV3CmlsLinkPortDSNbrRateRatio   Integer32,  
3    ieee8021AsV3CmlsLinkPortDSInitialLogPdelayReqInterval Integer32,  
4    ieee8021AsV3CmlsLinkPortDSCurrentLogPdelayReqInterval Integer32,  
5    ieee8021AsV3CmlsLinkPortDSUseMgtSettableLogPdelayReqInterval TruthValue,  
6    ieee8021AsV3CmlsLinkPortDSMgtSettableLogPdelayReqInterval Integer32,  
7    ieee8021AsV3CmlsLinkPortDSInitialComputeNbrRateRatio  TruthValue,  
8    ieee8021AsV3CmlsLinkPortDSCurrentComputeNbrRateRatio  TruthValue,  
9    ieee8021AsV3CmlsLinkPortDSUseMgtSettableComputeNbrRateRatio  TruthValue,  
10   ieee8021AsV3CmlsLinkPortDSMgtSettableComputeNbrRateRatio  TruthValue,  
11   ieee8021AsV3CmlsLinkPortDSInitialComputeMeanLinkDelay  TruthValue,  
12   ieee8021AsV3CmlsLinkPortDSCurrentComputeMeanLinkDelay  TruthValue,  
13   ieee8021AsV3CmlsLinkPortDSUseMgtSettableComputeMeanLinkDelay  TruthValue,  
14   ieee8021AsV3CmlsLinkPortDSMgtSettableComputeMeanLinkDelay  TruthValue,  
15   ieee8021AsV3CmlsLinkPortDSAAllowedLostRsp  Unsigned32,  
16   ieee8021AsV3CmlsLinkPortDSAllowedFaults        Unsigned32,  
17   ieee8021AsV3CmlsLinkPortDSVersionNumber       Unsigned32,  
18   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST1 Ieee8021ASV3Timestamp,  
19   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST2 Ieee8021ASV3Timestamp,  
20   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST3 Ieee8021ASV3Timestamp,  
21   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST4 Ieee8021ASV3Timestamp,  
22   ieee8021AsV3CmlsLinkPortDSMinorVersionNumber Unsigned32  
23   }  
24  
25 ieee8021AsV3CmlsLinkPortDSAsIndex OBJECT-TYPE  
26   SYNTAX   InterfaceIndexOrZero  
27   MAX-ACCESS not-accessible
```

1 STATUS current
2 DESCRIPTION
3 "An index to identify an entry in the Comon Mean Link
4 Delay Link Port Data Set."
5 REFERENCE "14.16"
6 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 1 }
7
8 ieee8021AsV3CmlsLinkPortDSClockIdentity OBJECT-TYPE
9 SYNTAX Ieee8021AsV3ClockIdentity
10 MAX-ACCESS read-only
11 STATUS current
12 DESCRIPTION
13 "The value is the first of the portIdentity attribute
14 of the local port, which is a set made of
15 Ieee8021AsV3ClockIdentity and portNumber."
16 REFERENCE "14.16.2"
17 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 2 }
18
19 ieee8021AsV3CmlsLinkPortDSPortNumber OBJECT-TYPE
20 SYNTAX Unsigned32(0..65535)
21 MAX-ACCESS read-only
22 STATUS current
23 DESCRIPTION
24 "The value is the second of the portIdentity attribute
25 of the local port, which is a set made of
26 Ieee8021AsV3ClockIdentity and portNumber."
27 REFERENCE "14.16.2"

1 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 3 }

2

3 ieee8021AsV3CmldsLinkPortDSCmldsLinkPortEnabled

4 OBJECT-TYPE

5 SYNTAX TruthValue

6 MAX-ACCESS read-only

7 STATUS current

8 DESCRIPTION

9 "The value is equal to the value of the Boolean

10 cmldsLinkPortEnabled."

11 REFERENCE "14.16.3"

12 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 4 }

13

14 ieee8021AsV3CmldsLinkPortDSIsMeasuringDelay

15 OBJECT-TYPE

16 SYNTAX TruthValue

17 MAX-ACCESS read-only

18 STATUS current

19 DESCRIPTION

20 "The value is equal to the value of the Boolean

21 isMeasuringDelay."

22 REFERENCE "14.16.4"

23 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 5 }

24

25 ieee8021AsV3CmldsLinkPortDSAsCapableAcrossDomains

26 OBJECT-TYPE

27 SYNTAX TruthValue

1 MAX-ACCESS read-only
2 STATUS current
3 DESCRIPTION
4 "The value is equal to the value of the Boolean
5 asCapableAcrossDomains."
6 REFERENCE "14.16.5"
7 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 6 }
8
9 ieee8021AsV3CmldsLinkPortDSMeanLinkDelay
10 OBJECT-TYPE
11 SYNTAX Ieee8021ASV3PtpTimeInterval
12 MAX-ACCESS read-only
13 STATUS current
14 DESCRIPTION
15 "The value is equal to the value of the per-port global
16 variable meanLinkDelay. It is an estimate of the current
17 one-way propagation time on the link attached to this Link
18 Port, measured as specified for the respective medium. The
19 value is zero for Link Ports attached to IEEE 802.3 EPON
20 links and for the timeTransmitter port of an IEEE 802.11 link,
21 because one-way propagation delay is not measured on the
22 latter and not directly measured on the former.
23 NOTE: The underlying per-port global variable meanLinkDelay is
24 of type UScaledNS, which is a 96-Bit value. meanLinkDelay
25 values that are larger than the maximum value that can be
26 represented by the TimeInterval data type, i.e.,
27 0xFFFF FFFF FFFF FFFF (where the units are 2^{16} ns), used

1 for this managed object are set to this largest value."

2 REFERENCE "14.16.6"

3 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 7 }

4

5 ieee8021AsV3CmldsLinkPortDSMeanLinkDelayThresh

6 OBJECT-TYPE

7 SYNTAX Ieee8021ASV3PtpTimeInterval

8 MAX-ACCESS read-write

9 STATUS current

10 DESCRIPTION

11 "The value is equal to the value of the per-Link-Port global

12 variable meanLinkDelayThresh. It is the propagation time

13 threshold above which a Link Port (and therefore any PTP Ports

14 that use the CMLDS on this Link Port) is considered not

15 capable of participating in the IEEE 802.1AS protocol.

16 Setting this managed object causes the per-Link-Port global

17 variable meanLinkDelayThresh to have the same value.

18 NOTE: The underlying per-port global variable

19 meanLinkDelayThresh is of type UScaledNS, which is a 96-Bit

20 value. meanLinkDelayThresh values that are larger than the

21 maximum value that can be represented by the TimeInterval

22 data type, i.e., 0xFFFF FFFF FFFF FFFF (where the units are

23 2^{-16} ns), used for this managed object are set to this

24 largest value."

25 REFERENCE "14.16.7"

26 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 8 }

27

1 ieee8021AsV3CmldsLinkPortDSDelayAsym

2 OBJECT-TYPE

3 SYNTAX Ieee8021ASV3PtpTimeInterval

4 MAX-ACCESS read-write

5 STATUS current

6 DESCRIPTION

7 "The value is the asymmetry in the propagation delay on the
8 link attached to this Link Port relative to the local clock.

9 If propagation delay asymmetry is not modeled, then
10 delayAsymmetry is 0.

11 NOTE: The underlying per-port global variable delayAsymmetry
12 is of type ScaledNS, which is a 96-Bit value.

13 delayAsymmetry values that are larger than the maximum value
14 that can be represented by the TimeInterval data type, i.e.,
15 0xFFFF FFFF FFFF FFFF, (where the units are 2^{-16} ns),
16 used for this managed object are set to this largest value.

17 delayAsymmetry values that are less than the minimum value
18 that can be represented by the TimeInterval data type, i.e.,
19 0x8000 0000 0000 0001 written in twos complement form (where
20 the units are 2^{-16} ns), used for this managed object are
21 set to this smallest value."

22 REFERENCE "14.16.8"

23 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 9 }

24

25 ieee8021AsV3CmldsLinkPortDSNbrRateRatio

26 OBJECT-TYPE

27 SYNTAX Integer32

1 MAX-ACCESS read-only
2 STATUS current
3 DESCRIPTION
4 "The value is an estimate of the ratio of the frequency of the
5 LocalClock entity of the time-aware system at the other end
6 of the link attached to this Link Port, to the frequency of
7 the LocalClock entity of this time-aware system.
8 neighborRateRatio is expressed as the fractional frequency
9 offset multiplied by 2^41, i.e., the quantity
10 (neighborRateRatio -1.0)(2^41)."
11 REFERENCE "14.16.9"
12 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 10 }
13
14 ieee8021AsV3CmldsLinkPortDSInitialLogPdelayReqInterval OBJECT-TYPE
15 SYNTAX Integer32(-128..127)
16 MAX-ACCESS read-write
17 STATUS current
18 DESCRIPTION
19 "If useMgtSettableLogPdelayReqInterval is FALSE (2) then,
20 for full-duplex IEEE 802.3 media and for CSN media that use
21 the peer-to-peer delay mechanism to measure path delay, the
22 value is the logarithm to base 2 of the Pdelay_Req message
23 transmission interval used when (a) the Link Port is
24 initialized, or (b) a message interval request TLV is
25 received with the logLinkDelayInterval field set to 126.
26 For all other media, the value is 127."
27 REFERENCE "14.16.10"

1 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 11 }

2

3 ieee8021AsV3CmldsLinkPortDSCurrentLogPdelayReqInterval OBJECT-TYPE

4 SYNTAX Integer32(-128..127)

5 MAX-ACCESS read-only

6 STATUS current

7 DESCRIPTION

8 "For full-duplex IEEE 802.3 media and for CSN media that use

9 the peer-to-peer delay mechanism to measure path delay,

10 the value is the logarithm to the base 2 of the current

11 Pdelay_Req message transmission interval.

12 For all other media, the value is 127."

13 REFERENCE "14.16.11"

14 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 12 }

15

16 ieee8021AsV3CmldsLinkPortDSUseMgtSettableLogPdelayReqInterval OBJECT-TYPE

17 SYNTAX TruthValue

18 MAX-ACCESS read-write

19 STATUS current

20 DESCRIPTION

21 "The managed object is a Boolean that determines the source

22 of the sync interval and mean time interval between

23 successive Pdelay_Req messages. If the value is TRUE (1),

24 the value of currentLogPdelayReqInterval is set equal to

25 the value of mgtSettableLogPdelayReqInterval. If the value

26 of the managed object is FALSE (2), the value of

27 currentLogPdelayReqInterval is determined by the

1 LinkDelayIntervalSetting state machine."

2 REFERENCE "14.16.12"

3 DEFVAL { false }

4 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 13 }

5

6 ieee8021AsV3CmldsLinkPortDSMgtSettableLogPdelayReqInterval OBJECT-TYPE

7 SYNTAX Integer32 (-128..127)

8 MAX-ACCESS read-write

9 STATUS current

10 DESCRIPTION

11 "The value is the logarithm to base 2 of the mean time

12 interval between successive Pdelay_Req messages if

13 useMgtSettableLogPdelayReqInterval is TRUE (1). The

14 value is not used if useMgtSettableLogPdelayReqInterval

15 is FALSE (2)."

16 REFERENCE "14.16.13"

17 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 14 }

18

19 ieee8021AsV3CmldsLinkPortDSInitialComputeNbrRateRatio OBJECT-TYPE

20 SYNTAX TruthValue

21 MAX-ACCESS read-write

22 STATUS current

23 DESCRIPTION

24 "If useMgtSettableComputeNeighborRateRatio is FALSE (2),

25 then for full-duplex IEEE 802.3 media and for CSN media

26 that use the peer-to-peer delay mechanism to measure path

27 delay, the value is the initial value of

1 computeNeighborRateRatio.

2 For all other media, the value is TRUE."

3 REFERENCE "14.16.14"

4 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 15 }

5

6 ieee8021AsV3CmldsLinkPortDSCurrentComputeNbrRateRatio OBJECT-TYPE

7 SYNTAX TruthValue

8 MAX-ACCESS read-only

9 STATUS current

10 DESCRIPTION

11 "For full-duplex IEEE 802.3 media and for CSN media that use

12 the peer-to-peer delay mechanism to measure path delay,

13 the value is the current value of computeNeighborRateRatio.

14 For all other media, the value is TRUE (1)."

15 REFERENCE "14.16.15"

16 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 16 }

17

18 ieee8021AsV3CmldsLinkPortDSUseMgtSettableComputeNbrRateRatio OBJECT-TYPE

19 SYNTAX TruthValue

20 MAX-ACCESS read-write

21 STATUS current

22 DESCRIPTION

23 "The managed object is a Boolean that determines the source

24 of the value of computeNeighborRateRatio. If the value is

25 TRUE (1), the value of computeNeighborRateRatio is set equal

26 to the value of mgtSettablecomputeNeighborRateRatio. If

27 the value of the managed object is FALSE (2), the

1 value of currentComputeNeighborRateRatio is determined by
2 the LinkDelayIntervalSetting state machine."
3 REFERENCE "14.16.16"
4 DEFVAL { false }
5 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 17 }
6
7 ieee8021AsV3CmldsLinkPortDSMgtSettableComputeNbrRateRatio OBJECT-TYPE
8 SYNTAX TruthValue
9 MAX-ACCESS read-write
10 STATUS current
11 DESCRIPTION
12 "computeNeighborRateRatio is configured to this value if
13 useMgtSettableComputeNeighborRateRatio is TRUE (1). The
14 value is not used if useMgtSettableComputeNeighborRateRatio
15 is FALSE (2)."
16 REFERENCE "14.16.17"
17 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 18 }
18
19 ieee8021AsV3CmldsLinkPortDSInitialComputeMeanLinkDelay OBJECT-TYPE
20 SYNTAX TruthValue
21 MAX-ACCESS read-write
22 STATUS current
23 DESCRIPTION
24 "If useMgtSettableComputeMeanLinkDelay is FALSE (2) then,
25 for full-duplex IEEE 802.3 media and for CSN media that use
26 the peer-to-peer delay mechanism to measure path delay,
27 the value is the initial value of computeMeanLinkDelay.

1 For all other media, the value is TRUE (1)."

2 REFERENCE "14.16.18"

3 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 19 }

4

5 ieee8021AsV3CmldsLinkPortDSCurrentComputeMeanLinkDelay OBJECT-TYPE

6 SYNTAX TruthValue

7 MAX-ACCESS read-only

8 STATUS current

9 DESCRIPTION

10 "For full-duplex IEEE 802.3 media and for CSN media that

11 use the peer-to-peer delay mechanism to measure path delay,

12 the value is the current value of computeMeanLinkDelay.

13 For all other media, the value is TRUE."

14 REFERENCE "14.16.19"

15 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 20 }

16

17 ieee8021AsV3CmldsLinkPortDSUseMgtSettableComputeMeanLinkDelay OBJECT-TYPE

18 SYNTAX TruthValue

19 MAX-ACCESS read-write

20 STATUS current

21 DESCRIPTION

22 "The managed object is a Boolean that determines the source

23 of the value of computeMeanLinkDelay. If the value is

24 TRUE (1), the value of computeMeanLinkDelay is set equal

25 to the value of mgtSettableComputeMeanLinkDelay. If the

26 value of the managed object is FALSE (2), the value of

27 currentComputeMeanLinkDelay is determined by the

1 LinkDelayIntervalSetting state machine."

2 REFERENCE "14.16.20"

3 DEFVAL { false }

4 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 21 }

5

6 ieee8021AsV3CmldsLinkPortDSMgtSettableComputeMeanLinkDelay OBJECT-TYPE

7 SYNTAX TruthValue

8 MAX-ACCESS read-write

9 STATUS current

10 DESCRIPTION

11 "computeMeanLinkDelay is configured to this value if

12 useMgtSettableComputeMeanLinkDelay is TRUE (1). The value

13 is not used if useMgtSettableComputeMeanLinkDelay is

14 FALSE (2)."

15 REFERENCE "14.16.21"

16 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 22 }

17

18 ieee8021AsV3CmldsLinkPortDSAllowedLostRsp

19 OBJECT-TYPE

20 SYNTAX Unsigned32(1..255)

21 MAX-ACCESS read-write

22 STATUS current

23 DESCRIPTION

24 "The value is equal to the value of the per-Link-Port

25 global variable allowedLostResponses. It is the number

26 of Pdelay_Req messages without valid responses

27 above which a Link Port is considered to be not

1 exchanging peer delay messages with its neighbor.

2 Setting this managed object causes the per-Link-Port global

3 variable allowedLostResponses to have the same value."

4 REFERENCE "14.16.22"

5 DEFVAL { 9 }

6 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 23 }

7

8

9 ieee8021AsV3CmldsLinkPortDSAllowedFaults OBJECT-TYPE

10 SYNTAX Unsigned32(1..255)

11 MAX-ACCESS read-write

12 STATUS current

13 DESCRIPTION

14 "The value is equal to the value of the per-Link-Port global

15 variable allowedFaults. It is the number of faults above

16 which asCapableAcrossDomains is set to FALSE (2), i.e., a

17 Link Port is considered not capable of interoperating

18 with its neighbor via the IEEE 802.1AS protocol.

19 Setting this managed object causes the per-Link-Port global

20 variable allowedFaults to have the same value."

21 REFERENCE "14.16.23"

22 DEFVAL { 9 }

23 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 24 }

24

25 ieee8021AsV3CmldsLinkPortDSVersionNumber OBJECT-TYPE

26 SYNTAX Unsigned32(0..15)

27 MAX-ACCESS read-only

1 STATUS current
2 DESCRIPTION
3 "This value is set to versionPTP as specified in 10.6.2.2.4."
4 REFERENCE "14.16.24"
5 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 25 }
6
7 ieee8021AsV3CmldsLinkPortDSPdelayTruncTST1
8 OBJECT-TYPE
9 SYNTAX Ieee8021ASV3Timestamp
10 MAX-ACCESS read-only
11 STATUS current
12 DESCRIPTION
13 "For full-duplex IEEE 802.3 media and for CSN media that use
14 the peer-to-peer delay mechanism to measure path delay, the
15 first value, T1, of the four elements of this array is as
16 described in Table 14-9. For all other media, the values are
17 zero. This object corresponds to the timestamp t1 modulo 2^32
18 in Figure 11-1, and expressed in units of 2^-16 ns (i.e., the
19 value of this array element is equal to the remainder obtained
20 upon dividing the respective timestamp, expressed in units
21 of 2^-16 ns, by 2^48).
22 At any given time, the timestamp values stored in the T1, T2,
23 T3, T4 PdelayTruncTS are for the same, and most recently
24 completed, peer delay message exchange.
25 NOTE: This managed object is used with the asymmetry
26 measurement compensation procedure, which is based on
27 line-swapping."
80

1 REFERENCE "14.16.25"

2 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 26 }

3

4 ieee8021AsV3CmldsLinkPortDSPdelayTruncTST2

5 OBJECT-TYPE

6 SYNTAX Ieee8021ASV3Timestamp

7 MAX-ACCESS read-only

8 STATUS current

9 DESCRIPTION

10 "For full-duplex IEEE 802.3 media and for CSN media that use

11 the peer-to-peer delay mechanism to measure path delay, the

12 second value, T2, of the four elements of this array is as

13 described in Table 14-9. For all other media, the values are

14 zero. This object corresponds to the timestamp t1 modulo 2^{32}

15 in Figure 11-1, and expressed in units of 2^{-16} ns (i.e., the

16 value of this array element is equal to the remainder obtained

17 upon dividing the respective timestamp, expressed in units

18 of 2^{-16} ns, by 2^{48}).

19 At any given time, the timestamp values stored in the T1, T2,

20 T3, T4 PdelayTruncTS are for the same, and most recently

21 completed, peer delay message exchange.

22 NOTE: This managed object is used with the asymmetry

23 measurement compensation procedure, which is based on

24 line-swapping."

25 REFERENCE "14.16.25"

26 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 27 }

27

1 ieee8021AsV3CmlsLinkPortDSPdelayTruncTST3

2 OBJECT-TYPE

3 SYNTAX Ieee8021ASV3Timestamp

4 MAX-ACCESS read-only

5 STATUS current

6 DESCRIPTION

7 "For full-duplex IEEE 802.3 media and for CSN media that use

8 the peer-to-peer delay mechanism to measure path delay, the

9 third value, T3, of the four elements of this array is as

10 described in Table 14-9. For all other media, the values are

11 zero. This object corresponds to the timestamp t1 modulo 2^{32}

12 in Figure 11-1, and expressed in units of 2^{-16} ns (i.e., the

13 value of this array element is equal to the remainder obtained

14 upon dividing the respective timestamp, expressed in units

15 of 2^{-16} ns, by 2^{48}).

16 At any given time, the timestamp values stored in the T1, T2,

17 T3, T4 PdelayTruncTS are for the same, and most recently

18 completed, peer delay message exchange.

19 NOTE: This managed object is used with the asymmetry

20 measurement compensation procedure, which is based on

21 line-swapping."

22 REFERENCE "14.16.25"

23 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 28 }

24

25 ieee8021AsV3CmlsLinkPortDSPdelayTruncTST4

26 OBJECT-TYPE

27 SYNTAX Ieee8021ASV3Timestamp

1 MAX-ACCESS read-only

2 STATUS current

3 DESCRIPTION

4 "For full-duplex IEEE 802.3 media and for CSN media that use

5 the peer-to-peer delay mechanism to measure path delay, the

6 fourth value, T4, of the four elements of this array is as

7 described in Table 14-9. For all other media, the values are

8 zero. This object corresponds to the timestamp t1 modulo 2^{32}

9 in Figure 11-1, and expressed in units of 2^{-16} ns (i.e., the

10 value of this array element is equal to the remainder obtained

11 upon dividing the respective timestamp, expressed in units

12 of 2^{-16} ns, by 2^{48}).

13 At any given time, the timestamp values stored in the T1, T2,

14 T3, T4 PdelayTruncTS are for the same, and most recently

15 completed, peer delay message exchange.

16 NOTE: This managed object is used with the asymmetry

17 measurement compensation procedure, which is based on

18 line-swapping."

19 REFERENCE "14.16.25"

20 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 29 }

21

22 ieee8021AsV3CmldsLinkPortDSMinorVersionNumber OBJECT-TYPE

23 SYNTAX Unsigned32 (0..15)

24 MAX-ACCESS read-only

25 STATUS current

26 DESCRIPTION

27 "This value is set to minorVersionPTP as specified in

1 10.6.2.2.3."

2 REFERENCE "14.16.26"

3 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSEntry 30 }

4

5 -- =====

6 -- The Common Mean Link Delay Service Link Port Parameter

7 -- Statistics Data Set provides counters associated with Link

8 -- Port capabilities at a given time-aware system.

9 -- =====

10

11 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSTable OBJECT-TYPE

12 SYNTAX SEQUENCE OF Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry

13 MAX-ACCESS not-accessible

14 STATUS current

15 DESCRIPTION

16 "For every Link Port of the Common Mean Link Delay Service of a

17 time-aware system, the following cmldsLinkPortStatisticsDS

18 provides counters. The number of such statistics sets is the

19 same as the value of cmldsDefaultDS.numberLinkPorts."

20 REFERENCE "14.17"

21 ::= { ieee8021AsV3MIBObjects 19 }

22

23 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry OBJECT-TYPE

24 SYNTAX Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry

25 MAX-ACCESS not-accessible

26 STATUS current

27 DESCRIPTION

1 "For every Link Port of the Common Mean Link Delay Service of a
 2 time-aware system, the following cmldsLinkPortStatisticsDS
 3 provides counters. The number of such statistics sets is the
 4 same as the value of cmldsDefaultDS.numberLinkPorts."
 5 INDEX { ieee8021AsV3BridgeBasePort,
 6 ieee8021AsV3CmldsLinkPortStatDSIndex }
 7 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSTable 1 }
 8
 9 Ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry ::=
 10 SEQUENCE {
 11 ieee8021AsV3CmldsLinkPortStatDSIndex InterfaceIndexOrZero,
 12 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRequestCount Counter32,
 13 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspCount Counter32,
 14 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspFollowUpCount Counter32,
 15 ieee8021AsV3CmldsLinkPortStatDSRxPtpPacketDiscardCount Counter32,
 16 ieee8021AsV3CmldsLinkPortStatDSPdelayAllowedLostRspExceededCount Counter32,
 17 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRequestCount Counter32,
 18 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspCount Counter32,
 19 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspFollowUpCount Counter32
 20 }
 21
 22 ieee8021AsV3CmldsLinkPortStatDSIndex OBJECT-TYPE
 23 SYNTAX InterfaceIndexOrZero
 24 MAX-ACCESS not-accessible
 25 STATUS current
 26 DESCRIPTION
 27 "An index to identify an entry in the Common Mean Link

1 Port Statistics Data Set."

2 REFERENCE "14.17"

3 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 1 }

4

5 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRequestCount OBJECT-TYPE

6 SYNTAX Counter32

7 MAX-ACCESS read-only

8 STATUS current

9 DESCRIPTION

10 "A counter that increments every time a Pdelay_Req message is

11 received."

12 REFERENCE "14.17.2"

13 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 2 }

14

15 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspCount OBJECT-TYPE

16 SYNTAX Counter32

17 MAX-ACCESS read-only

18 STATUS current

19 DESCRIPTION

20 "A counter that increments every time a Pdelay_Resp message is

21 received."

22 REFERENCE "14.17.3"

23 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 3 }

24

25 ieee8021AsV3CmldsLinkPortStatDSRxPdelayRspFollowUpCount OBJECT-TYPE

26 SYNTAX Counter32

27 MAX-ACCESS read-only

1 STATUS current

2 DESCRIPTION

3 "A counter that increments every time a Pdelay_Resp_Follow_Up
4 message is received."

5 REFERENCE "14.17.4"

6 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 4 }

7

8 ieee8021AsV3CmldsLinkPortStatDSRxPtpPacketDiscardCount

9 OBJECT-TYPE

10 SYNTAX Counter32

11 MAX-ACCESS read-only

12 STATUS current

13 DESCRIPTION

14 "A counter that increments every time a PTP message of the
15 Common Mean Link Delay Service is discarded, caused by the
16 occurrence of any of the conditions given in 14.17.5."

17 REFERENCE "14.17.5"

18 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 5 }

19

20 ieee8021AsV3CmldsLinkPortStatDSPdelayAllowedLostRspExceededCount

21 OBJECT-TYPE

22 SYNTAX Counter32

23 MAX-ACCESS read-only

24 STATUS current

25 DESCRIPTION

26 "A counter that increments every time the value of the variable
27 lostResponses exceeds the value of the variable

1 allowedLostResponses, in the RESET state of the
2 MDPdelayReq state machine."
3 REFERENCE "14.17.6"
4 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 6 }
5
6 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRequestCount
7 OBJECT-TYPE
8 SYNTAX Counter32
9 MAX-ACCESS read-only
10 STATUS current
11 DESCRIPTION
12 "A counter that increments every time a Pdelay_Req message is
13 transmitted."
14 REFERENCE "14.17.7"
15 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 7 }
16
17 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspCount
18 OBJECT-TYPE
19 SYNTAX Counter32
20 MAX-ACCESS read-only
21 STATUS current
22 DESCRIPTION
23 "A counter that increments every time a Pdelay_Resp message is
24 transmitted."
25 REFERENCE "14.17.8"
26 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 8 }
27

1 ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspFollowUpCount

2 OBJECT-TYPE

3 SYNTAX Counter32

4 MAX-ACCESS read-only

5 STATUS current

6 DESCRIPTION

7 "A counter that increments every time a Pdelay_Resp_Follow_Up

8 message is transmitted."

9 REFERENCE "14.17.9"

10 ::= { ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSEntry 9 }

11

12 -- =====

13 -- The Common Mean Link Delay Service Asymmetry Measurement Mode

14 -- Parameter Data Set represents the capability to enable/disable

15 -- the Asymmetry Compensation Measurement Procedure on a Link Port

16 -- (see Annex G).

17 -- =====

18

19 ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSTable OBJECT-TYPE

20 SYNTAX SEQUENCE OF
21 Ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry

22 MAX-ACCESS not-accessible

23 STATUS current

24 DESCRIPTION

25 "The Common Mean Link Delay Service Asymmetry Measurement Mode

26 Parameter Data Set represents the capability to enable/disable

27 the Asymmetry Compensation Measurement Procedure on a Link Port

28 (see Annex G)."

1 REFERENCE "14.18"
2 ::= { ieee8021AsV3MIBObjects 20 }
3
4 ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry OBJECT-TYPE
5 SYNTAX Ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry
6 MAX-ACCESS not-accessible
7 STATUS current
8 DESCRIPTION
9 "This table uses
10 ieee8021AsV3CmldsAsymmetryMeasurementModeDSAsIndex,
11 and corresponds to
12 ieee8021AsV3CommonMeanLinkDelayServiceAsymmetryMeasurementModeDSTable
13 entry."
14 INDEX { ieee8021AsV3BridgeBasePort,
15 ieee8021AsV3CmldsAsymMeasurementModeDSAsIndex }
16 ::= { ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSTable 1 }
17
18 Ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry ::=
19 SEQUENCE {
20 ieee8021AsV3CmldsAsymMeasurementModeDSAsIndex InterfaceIndexOrZero,
21 ieee8021AsV3CmldsAsymMeasurementModeDSAsymMeasurementMode TruthValue
22 }
23
24 ieee8021AsV3CmldsAsymMeasurementModeDSAsIndex OBJECT-TYPE
25 SYNTAX InterfaceIndexOrZero
26 MAX-ACCESS not-accessible
27 STATUS current

1 DESCRIPTION

2 "This object identifies the gPTP interface group within
3 the system for which this entry contains information. It
4 is the value of the instance of the IfIndex object,
5 defined in the IF-MIB, for the gPTP interface group
6 corresponding to this port, or the value 0 if the port
7 has not been bound to an underlying frame source and
8 sink.

9

10 For a given media port of a Bridge or an end station,
11 there can be one or more PTP Port, and depends whether
12 a media port supports point to point link (e.g. IEEE
13 802.3 Ethernet) or point to multi-point (e.g. CSN, IEEE
14 802.3 EPON) links on the media port."

15 REFERENCE "IEEE Std 802.1AS CommonMeanLinkDelayServiceAsymMeasurementModeParamDS
16 Group PTP Port Index"

17 ::= { ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry 1 }

18

19 ieee8021AsV3CmldsAsymMeasurementModeDSAsymMeasurementMode

20 OBJECT-TYPE

21 SYNTAX TruthValue

22 MAX-ACCESS read-write

23 STATUS current

24 DESCRIPTION

25 "The value is equal to the value of the Boolean
26 asymmetryMeasurementMode (see G.3). For full-duplex
27 IEEE 802.3 media, the value is TRUE (1) if an asymmetry
28 measurement is being performed for the link attached to

1 this Link Port, and FALSE (2) otherwise. For all other
2 media, the value shall be FALSE (2) (see 10.2.5.2).
3 Setting this managed object causes the Boolean
4 allowedFaults to have the same value.
5 NOTE: If an asymmetry measurement is being performed
6 for a link, asymmetryMeasurementMode must be TRUE (1)
7 for the Link Ports at each end of the link.
8 There is one Common Mean Link Delay Service Asymmetry
9 Measurement Mode Parameter Data Set Table for all PTP
10 Instances, per Link Port."
11 REFERENCE "14.18.2"
12 ::= { ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSEntry 2 }
13
14 -- ****=
15 -- IEEE 802.1ASV3 MIB - Conformance Information
16 -- ****=
17 ieee8021AsV3Groups OBJECT IDENTIFIER ::= { ieee8021AsV3Conformance 1 }
18 ieee8021AsV3Compliances OBJECT IDENTIFIER ::= { ieee8021AsV3Conformance 2 }
19
20 -- =====
21 -- units of conformance
22 -- =====
23
24 ieee8021AsV3PtpInstanceGroup OBJECT-GROUP
25 OBJECTS {
26 ieee8021AsV3PtpInstanceName,
27 ieee8021AsV3PtpInstanceRowStatus

```
1   }
2   STATUS    current
3   DESCRIPTION
4   "A collection of objects providing information for dynamic
5   creation and deletion of PTP Instances and logical ports."
6 ::= { ieee8021AsV3Groups 1 }
```

```
7
```

```
8 ieee8021AsV3DefaultDSGroup OBJECT-GROUP
```

```
9   OBJECTS {
10     ieee8021AsV3DefaultDSClockIdentity,
11     ieee8021AsV3DefaultDSNumberPorts,
12     ieee8021AsV3DefaultDSClockQualityClockClass,
13     ieee8021AsV3DefaultDSClockQualityClockAccuracy,
14     ieee8021AsV3DefaultDSClockQualityOffsetScaledLogVariance,
15     ieee8021AsV3DefaultDSPriority1,
16     ieee8021AsV3DefaultDSPriority2,
17     ieee8021AsV3DefaultDSGmCapable,
18     ieee8021AsV3DefaultDSCurrentUtcOffset,
19     ieee8021AsV3DefaultDSCurrentUtcOffsetValid,
20     ieee8021AsV3DefaultDSLeap59,
21     ieee8021AsV3DefaultDSLeap61,
22     ieee8021AsV3DefaultDSTimeTraceable,
23     ieee8021AsV3DefaultDSFrequencyTraceable,
24     ieee8021AsV3DefaultDSPtpTimescale,
25     ieee8021AsV3DefaultDSTimeSource,
26     ieee8021AsV3DefaultDSDomainNumber,
27     ieee8021AsV3DefaultDSSdoId,
```

```
1     ieee8021AsV3DefaultDSExternalPortConfigurationEnabled,  
2     ieee8021AsV3DefaultDSInstanceEnable  
3 }  
4 STATUS current  
5 DESCRIPTION  
6 "A collection of objects providing information on the Default  
7 Parameter Data Set representing the native capabilities of a  
8 PTP Instance, i.e., a PTP Relay Instance or a PTP End Instance."  
9 ::= { ieee8021AsV3Groups 2 }  
10  
11 ieee8021AsV3CurrentDSGroup OBJECT-GROUP  
12 OBJECTS {  
13     ieee8021AsV3CurrentDSStepsRemoved,  
14     ieee8021AsV3CurrentDSOffsetFromTimeTransmitter,  
15     ieee8021AsV3CurrentDSLastGmPhaseChange,  
16     ieee8021AsV3CurrentDSLastGmFreqChange,  
17     ieee8021AsV3CurrentDSGmTimebaseIndicator,  
18     ieee8021AsV3CurrentDSGmChangeCount,  
19     ieee8021AsV3CurrentDSTimeOfLastGmChangeEvent,  
20     ieee8021AsV3CurrentDSTimeOfLastGmPhaseChangeEvent,  
21     ieee8021AsV3CurrentDSTimeOfLastGmFreqChangeEvent  
22 }  
23 STATUS current  
24 DESCRIPTION  
25 "A collection of objects providing information on the Current  
26 Parameter Data Set representing the position of a local system  
27 and other information, relative to the Grandmaster PTP Instance."
```

```
1 ::= { ieee8021AsV3Groups 3 }

2

3 ieee8021AsV3ParentDSGroup OBJECT-GROUP

4 OBJECTS {

5   ieee8021AsV3ParentDSParentClockIdentity,
6   ieee8021AsV3ParentDSParentPortNumber,
7   ieee8021AsV3ParentDSCumulativeRateRatio,
8   ieee8021AsV3ParentDSGrandmasterIdentity,
9   ieee8021AsV3ParentDSGrandmasterClockQualityclockClass,
10  ieee8021AsV3ParentDSGrandmasterClockQualityclockAccuracy,
11  ieee8021AsV3ParentDSGrandmasterClockQualityoffsetScaledLogVar,
12  ieee8021AsV3ParentDSGrandmasterPriority1,
13  ieee8021AsV3ParentDSGrandmasterPriority2

14 }

15 STATUS current

16 DESCRIPTION

17 "A collection of objects providing information on the Parent
18 Parameter Data Set representing capabilities of the upstream
19 system, toward the Grandmaster PTP Instance, as measured at
20 a local system."

21 ::= { ieee8021AsV3Groups 4 }

22

23 ieee8021AsV3TimePropertiesDSGroup OBJECT-GROUP

24 OBJECTS {

25   ieee8021AsV3TimePropertiesDSCurrentUtcOffset,
26   ieee8021AsV3TimePropertiesDSCurrentUtcOffsetValid,
27   ieee8021AsV3TimePropertiesDSLeap59,
```

```
1     ieee8021AsV3TimePropertiesDSLeap61,  
2     ieee8021AsV3TimePropertiesDSTimeTraceable,  
3     ieee8021AsV3TimePropertiesDSFrequencyTraceable,  
4     ieee8021AsV3TimePropertiesDSPtpTimescale,  
5     ieee8021AsV3TimePropertiesDSTimeSource  
6 }  
7 STATUS current  
8 DESCRIPTION  
9 "A collection of objects providing information on the Time  
10 Properties Parameter Data Set representing capabilities of  
11 the Grandmaster PTP Instance, as measured at a local system."  
12 ::= { ieee8021AsV3Groups 5 }  
13  
14 ieee8021AsV3PathTraceDSGroup OBJECT-GROUP  
15 OBJECTS {  
16     ieee8021AsV3PathTraceDSEnable  
17 }  
18 STATUS current  
19 DESCRIPTION  
20 "A collection of objects providing information on the Path Trace  
21 Data Set representing the current path trace information  
22 available at the PTP Instance."  
23 ::= { ieee8021AsV3Groups 6 }  
24  
25 ieee8021AsV3PathTraceDSArrayTableGroup OBJECT-GROUP  
26 OBJECTS {  
27     ieee8021AsV3PathTraceDSArrayList
```

1 }
2 STATUS current
3 DESCRIPTION
4 "A collection of objects providing information of an array of
5 ClockIdentity values contained in the pathTrace array,
6 representing the current path trace information, and which is
7 carried in the path trace TLV per PTP Instance."
8 ::= { ieee8021AsV3Groups 7 }
9

10 ieee8021AsV3AcceptableTimeTransmitterTableDSGroup OBJECT-GROUP

11 OBJECTS {
12 ieee8021AsV3AcceptableTimeTransmitterTableDSMaxTableSize,
13 ieee8021AsV3AcceptableTimeTransmitterTableDSActualTableSize
14 }
15 STATUS current
16 DESCRIPTION
17 "A collection of objects providing information on the
18 Acceptable TimeTransmitter Table Data Set representing the acceptable
19 timeTransmitter table used when an EPON port is used by a PTP Instance
20 of a time-aware system."
21 ::= { ieee8021AsV3Groups 8 }
22

23 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayGroup OBJECT-GROUP

24 OBJECTS {
25 ieee8021AsV3AcceptableTimeTransmitterTableDSArrayPortIdentity,
26 ieee8021AsV3AcceptableTTTableDSArrayAlternatePriority1
27 }

```
1 STATUS    current
2 DESCRIPTION
3 "A collection of objects providing information on the
4 Acceptable TimeTransmitter Table Array Data Set representing the
5 acceptable timeTransmitter table used when an EPON port is used by a
6 PTP Instance of a time-aware system."
7 ::= { ieee8021AsV3Groups 9 }
8
9 ieee8021AsV3PortDSGroup OBJECT-GROUP
10 OBJECTS {
11     ieee8021AsV3PortDSClockIdentity,
12     ieee8021AsV3PortDSPortNumber,
13     ieee8021AsV3PortDSPortState,
14     ieee8021AsV3PortDSPtpPortEnabled,
15     ieee8021AsV3PortDSDelayMechanism,
16     ieee8021AsV3PortDSIsMeasuringDelay,
17     ieee8021AsV3PortDSAsCapable,
18     ieee8021AsV3PortDSMeanLinkDelay,
19     ieee8021AsV3PortDSMeanLinkDelayThresh,
20     ieee8021AsV3PortDSDelayAsym,
21     ieee8021AsV3PortDSNbrRateRatio,
22     ieee8021AsV3PortDSInitialLogAnnounceInterval,
23     ieee8021AsV3PortDSCurrentLogAnnounceInterval,
24     ieee8021AsV3PortDSUseMgtSettableLogAnnounceInterval,
25     ieee8021AsV3PortDSMgtSettableLogAnnounceInterval,
26     ieee8021AsV3PortDSAnnounceReceiptTimeout,
27     ieee8021AsV3PortDSInitialLogSyncInterval,
```

- 1 ieee8021AsV3PortDSCurrentLogSyncInterval,
- 2 ieee8021AsV3PortDSUseMgtSettableLogSyncInterval,
- 3 ieee8021AsV3PortDSMgtSettableLogSyncInterval,
- 4 ieee8021AsV3PortDSSyncReceiptTimeout,
- 5 ieee8021AsV3PortDSSyncReceiptTimeoutTimeInterval,
- 6 ieee8021AsV3PortDSInitialLogPdelayReqInterval,
- 7 ieee8021AsV3PortDSCurrentLogPdelayReqInterval,
- 8 ieee8021AsV3PortDSUseMgtSettableLogPdelayReqInterval,
- 9 ieee8021AsV3PortDSMgtSettableLogPdelayReqInterval,
- 10 ieee8021AsV3PortDSInitialLogGptpCapableMessageInterval,
- 11 ieee8021AsV3PortDSCurrentLogGptpCapableMessageInterval,
- 12 ieee8021AsV3PortDSUseMgtSettableLogGptpCapableMessageInterval,
- 13 ieee8021AsV3PortDSMgtSettableLogGptpCapableMessageInterval,
- 14 ieee8021AsV3PortDSInitialComputeNbrRateRatio,
- 15 ieee8021AsV3PortDSCurrentComputeNbrRateRatio,
- 16 ieee8021AsV3PortDSUseMgtSettableComputeNbrRateRatio,
- 17 ieee8021AsV3PortDSMgtSettableComputeNbrRateRatio,
- 18 ieee8021AsV3PortDSInitialComputeMeanLinkDelay,
- 19 ieee8021AsV3PortDSCurrentComputeMeanLinkDelay,
- 20 ieee8021AsV3PortDSUseMgtSettableComputeMeanLinkDelay,
- 21 ieee8021AsV3PortDSMgtSettableComputeMeanLinkDelay,
- 22 ieee8021AsV3PortDSAllowedLostRsp,
- 23 ieee8021AsV3PortDSAllowedFaults,
- 24 ieee8021AsV3PortDSGPtpCapableReceiptTimeout,
- 25 ieee8021AsV3PortDSVersionNumber,
- 26 ieee8021AsV3PortDSNup,
- 27 ieee8021AsV3PortDSNdown,

```
1     ieee8021AsV3PortDSOneStepTxOper,  
2     ieee8021AsV3PortDSOneStepReceive,  
3     ieee8021AsV3PortDSOneStepTransmit,  
4     ieee8021AsV3PortDSInitialOneStepTxOper,  
5     ieee8021AsV3PortDSCurrentOneStepTxOper,  
6     ieee8021AsV3PortDSUseMgtSettableOneStepTxOper,  
7     ieee8021AsV3PortDSMgtSettableOneStepTxOper,  
8     ieee8021AsV3PortDSSyncLocked,  
9     ieee8021AsV3PortDSPdelayTruncTST1,  
10    ieee8021AsV3PortDSPdelayTruncTST2,  
11    ieee8021AsV3PortDSPdelayTruncTST3,  
12    ieee8021AsV3PortDSPdelayTruncTST4,  
13    ieee8021AsV3PortDSMinorVersionNumber  
14  }  
15 STATUS current  
16 DESCRIPTION  
17 "A collection of objects providing information on PTP Port  
18 related variables in a time-aware Bridge or for a time-aware  
19 end station."  
20 ::= { ieee8021AsV3Groups 10 }  
21  
22 ieee8021AsV3DescriptionPortDSGroup OBJECT-GROUP  
23 OBJECTS {  
24     ieee8021AsV3DescriptionPortDSProfileIdentifier  
25 }  
26 STATUS current  
27 DESCRIPTION
```

1 "A collection of objects providing information on the
2 Description Port Data Set containing the profileIdentifier for
3 this PTP profile, as specified in F.2."
4 ::= { ieee8021AsV3Groups 11 }
5
6 ieee8021AsV3PortStatIfGroup OBJECT-GROUP
7 OBJECTS {
8 ieee8021AsV3PortStatRxSyncCount,
9 ieee8021AsV3PortStatRxOneStepSyncCount,
10 ieee8021AsV3PortStatRxFollowUpCount,
11 ieee8021AsV3PortStatRxPdelayRequestCount,
12 ieee8021AsV3PortStatRxPdelayRspCount,
13 ieee8021AsV3PortStatRxPdelayRspFollowUpCount,
14 ieee8021AsV3PortStatRxAnnounceCount,
15 ieee8021AsV3PortStatRxPtpPacketDiscardCount,
16 ieee8021AsV3PortStatSyncReceiptTimeoutCount,
17 ieee8021AsV3PortStatAnnounceReceiptTimeoutCount,
18 ieee8021AsV3PortStatPdelayAllowedLostRspExceededCount,
19 ieee8021AsV3PortStatTxSyncCount,
20 ieee8021AsV3PortStatTxOneStepSyncCount,
21 ieee8021AsV3PortStatTxFollowUpCount,
22 ieee8021AsV3PortStatTxPdelayRequestCount,
23 ieee8021AsV3PortStatTxPdelayRspCount,
24 ieee8021AsV3PortStatTxPdelayRspFollowUpCount,
25 ieee8021AsV3PortStatTxAnnounceCount
26 }
27 STATUS current

1 DESCRIPTION

2 "A collection of objects providing information on the Port
3 Statistics Data Set providing counters associated with PTP Port
4 capabilities at a given PTP Instance."

5 ::= { ieee8021AsV3Groups 12 }

6

7 ieee8021AsV3AcceptableTimeTransmitterPortDSGroup OBJECT-GROUP

8 OBJECTS {

9 ieee8021AsV3AcceptableTTPortDSAcceptableTTTableEnabled

10 }

11 STATUS current

12 DESCRIPTION

13 "A collection of objects providing information for the single
14 PTP Port of a PTP End Instance and for each PTP Port of a
15 PTP Relay Instance."

16 ::= { ieee8021AsV3Groups 13 }

17

18 ieee8021AsV3ExternalPortConfigurationPortDSGroup OBJECT-GROUP

19 OBJECTS {

20 ieee8021AsV3ExternalPortConfigurationPortDSDesiredState

21 }

22 STATUS current

23 DESCRIPTION

24 "A collection of objects providing information on the
25 External Port Configuration Port Data Set containing the
26 single member desiredState, which indicates the desired state
27 for the PTP Port."

```
1 ::= { ieee8021AsV3Groups 14 }

2

3 ieee8021AsV3AsymMeasurementModeDSGroup OBJECT-GROUP

4 OBJECTS {

5     ieee8021AsV3AsymMeasurementModeDSAsymMeasurementMode

6 }

7 STATUS current

8 DESCRIPTION

9 "A collection of objects providing information on the

10 Asymmetry Measurement Mode Data Set representing the capability

11 to enable/disable the Asymmetry Compensation Measurement

12 Procedure on a Link Port (see Annex G)."

13 ::= { ieee8021AsV3Groups 15 }

14

15 ieee8021AsV3CommonServicesPortDSGroup OBJECT-GROUP

16 OBJECTS {

17     ieee8021AsV3CommonServicesPortDSCmldsLinkPortPortNumber

18 }

19 STATUS current

20 DESCRIPTION

21 "A collection of objects providing information on the

22 Common Services Port Data Set."

23 ::= { ieee8021AsV3Groups 16 }

24

25 ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSGroup OBJECT-GROUP

26 OBJECTS {

27     ieee8021AsV3CmldsDefaultDSClockIdentity,
```

```
1     ieee8021AsV3CmldsDefaultDSNumberLinkPorts
2   }
3   STATUS    current
4   DESCRIPTION
5   "A collection of objects providing information on the
6   CMLDs Default Data Set describing the per-time-aware-system
7   attributes of the Common Mean Link Delay Service."
8 ::= { ieee8021AsV3Groups 17 }

9
10 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSGroup OBJECT-GROUP
11   OBJECTS {
12     ieee8021AsV3CmldsLinkPortDSClockIdentity,
13     ieee8021AsV3CmldsLinkPortDSPortNumber,
14     ieee8021AsV3CmldsLinkPortDSCmldsLinkPortEnabled,
15     ieee8021AsV3CmldsLinkPortDSIsMeasuringDelay,
16     ieee8021AsV3CmldsLinkPortDSAsCapableAcrossDomains,
17     ieee8021AsV3CmldsLinkPortDSMeanLinkDelay,
18     ieee8021AsV3CmldsLinkPortDSMeanLinkDelayThresh,
19     ieee8021AsV3CmldsLinkPortDSDelayAsym,
20     ieee8021AsV3CmldsLinkPortDSNbrRateRatio,
21     ieee8021AsV3CmldsLinkPortDSInitialLogPdelayReqInterval,
22     ieee8021AsV3CmldsLinkPortDSCurrentLogPdelayReqInterval,
23     ieee8021AsV3CmldsLinkPortDSUseMgtSettableLogPdelayReqInterval,
24     ieee8021AsV3CmldsLinkPortDSMgtSettableLogPdelayReqInterval,
25     ieee8021AsV3CmldsLinkPortDSInitialComputeNbrRateRatio,
26     ieee8021AsV3CmldsLinkPortDSCurrentComputeNbrRateRatio,
27     ieee8021AsV3CmldsLinkPortDSUseMgtSettableComputeNbrRateRatio,
```

```
1    ieee8021AsV3CmlsLinkPortDSMgtSettableComputeNbrRateRatio,  
2    ieee8021AsV3CmlsLinkPortDSInitialComputeMeanLinkDelay,  
3    ieee8021AsV3CmlsLinkPortDSCurrentComputeMeanLinkDelay,  
4    ieee8021AsV3CmlsLinkPortDSUseMgtSettableComputeMeanLinkDelay,  
5    ieee8021AsV3CmlsLinkPortDSMgtSettableComputeMeanLinkDelay,  
6    ieee8021AsV3CmlsLinkPortDSAllowedLostRsp,  
7    ieee8021AsV3CmlsLinkPortDSAllowedFaults,  
8    ieee8021AsV3CmlsLinkPortDSVersionNumber,  
9    ieee8021AsV3CmlsLinkPortDSPdelayTruncTST1,  
10   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST2,  
11   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST3,  
12   ieee8021AsV3CmlsLinkPortDSPdelayTruncTST4,  
13   ieee8021AsV3CmlsLinkPortDSMinorVersionNumber  
14 }
```

```
15 STATUS current
```

```
16 DESCRIPTION
```

```
17 "A collection of objects providing information for every  
18 Link Port of the Common Mean Link Delay Service of a  
19 time-aware system."
```

```
20 ::= { ieee8021AsV3Groups 18 }
```

```
21
```

```
22 ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSGroup OBJECT-GROUP
```

```
23 OBJECTS {
```

```
24   ieee8021AsV3CmlsLinkPortStatDSRxPdelayRequestCount,  
25   ieee8021AsV3CmlsLinkPortStatDSRxPdelayRspCount,  
26   ieee8021AsV3CmlsLinkPortStatDSRxPdelayRspFollowUpCount,  
27   ieee8021AsV3CmlsLinkPortStatDSRxPtpPacketDiscardCount,
```

```
1     ieee8021AsV3CmldsLinkPortStatDSPdelayAllowedLostRspExceededCount,
2     ieee8021AsV3CmldsLinkPortStatDSTxPdelayRequestCount,
3     ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspCount,
4     ieee8021AsV3CmldsLinkPortStatDSTxPdelayRspFollowUpCount
5   }
6 STATUS current
7 DESCRIPTION
8 "A collection of objects providing information for every
9 Link Port Statistics of the Common Mean Link Delay Service of a
10 time-aware system."
11 ::= { ieee8021AsV3Groups 19 }
12
13 ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSGroup OBJECT-GROUP
14 OBJECTS {
15   ieee8021AsV3CmldsAsymMeasurementModeDSAsymMeasurementMode
16 }
17 STATUS current
18 DESCRIPTION
19 "A collection of objects providing information on the
20 Common Mean Link Delay Service Asymmetry Measurement Mode
21 Parameter Data Set representing the capability to enable/disable
22 the Asymmetry Compensation Measurement Procedure on a Link Port
23 (see Annex G)."
24 ::= { ieee8021AsV3Groups 20 }
25
26 -- -----
27 -- compliance statements
```

1 -- =====
2
3 ieee8021AsV3Compliance MODULE-COMPLIANCE
4 STATUS current
5 DESCRIPTION
6 "The compliance statement for devices supporting
7 IEEE Std 802.1AS-2020."
8
9 MODULE -- this module
10
11 GROUP ieee8021AsV3PtpInstanceGroup
12 DESCRIPTION
13 "Implementation of this group is optional."
14
15 GROUP ieee8021AsV3DefaultDSGroup
16 DESCRIPTION
17 "Implementation of this group is optional."
18
19 GROUP ieee8021AsV3CurrentDSGroup
20 DESCRIPTION
21 "Implementation of this group is optional."
22
23 GROUP ieee8021AsV3ParentDSGroup
24 DESCRIPTION
25 "Implementation of this group is optional."
26
27 GROUP ieee8021AsV3TimePropertiesDSGroup

1 DESCRIPTION

2 "Implementation of this group is optional."

3

4 GROUP ieee8021AsV3PathTraceDSGroup

5 DESCRIPTION

6 "Implementation of this group is optional."

7

8 GROUP ieee8021AsV3PathTraceDSArrayTableGroup

9 DESCRIPTION

10 "Implementation of this group is optional."

11

12 GROUP ieee8021AsV3AcceptableTimeTransmitterTableDSGroup

13 DESCRIPTION

14 "Implementation of this group is optional."

15

16 GROUP ieee8021AsV3AcceptableTimeTransmitterTableDSArrayGroup

17 DESCRIPTION

18 "Implementation of this group is optional."

19

20 GROUP ieee8021AsV3PortDSGroup

21 DESCRIPTION

22 "Implementation of this group is optional."

23

24 GROUP ieee8021AsV3DescriptionPortDSGroup

25 DESCRIPTION

26 "Implementation of this group is optional."

27

1 GROUP ieee8021AsV3PortStatIfGroup

2 DESCRIPTION

3 "Implementation of this group is optional."

4

5 GROUP ieee8021AsV3AcceptableTimeTransmitterPortDSGroup

6 DESCRIPTION

7 "Implementation of this group is optional."

8

9 GROUP ieee8021AsV3ExternalPortConfigurationPortDSGroup

10 DESCRIPTION

11 "Implementation of this group is optional."

12

13 GROUP ieee8021AsV3AsymMeasurementModeDSGroup

14 DESCRIPTION

15 "Implementation of this group is optional."

16

17 GROUP ieee8021AsV3CommonServicesPortDSGroup

18 DESCRIPTION

19 "Implementation of this group is optional."

20

21 GROUP ieee8021AsV3CommonMeanLinkDelayServiceDefaultDSGroup

22 DESCRIPTION

23 "Implementation of this group is optional."

24

25 GROUP ieee8021AsV3CommonMeanLinkDelayServiceLinkPortDSGroup

26 DESCRIPTION

27 "Implementation of this group is optional."

1
2 GROUP ieee8021AsV3CommonMeanLinkDelayServiceLinkPortStatDSGroup
3 DESCRIPTION
4 "Implementation of this group is optional."
5
6 GROUP ieee8021AsV3CommonMeanLinkDelayServiceAsymMeasurementModeDSGroup
7 DESCRIPTION
8 "Implementation of this group is optional."
9
10 ::= { ieee8021AsV3Compliances 1 }
11
12 END
13

1 16. Media-dependent layer specification for CSN

2 16.1 Overview

3 Accurate synchronized time is distributed throughout a gPTP domain through time measurements between
 4 adjacent PTP Relay Instances or PTP End Instances in a packet network. Time is communicated from the
 5 root of the clock spanning tree (i.e., the Grandmaster PTP Instance) toward the leaves of the tree (i.e., from
 6 leaf-facing “timeTransmitter” ports to root-facing “timeReceiver” ports) through measurements made across
 7 the links connecting the PTP Instances. While the semantics of time transfer are consistent across the time-
 8 aware packet network, the method for communicating synchronized time from a timeTransmitter port to its
 9 immediate downstream link partner varies depending on the type of link interconnecting the two PTP
 10 Instances.

11 This clause specifies the protocol that provides accurate synchronized time across links of a *coordinated*
 12 *shared network* (CSN) as part of a packet network.

13 16.2 Coordinated Shared Network characteristics

14 A CSN is a contention-free, time-division, multiplexed-access network of devices sharing a common
 15 medium and supporting reserved bandwidth based on priority or flow (QoS). One of the nodes of the CSN
 16 acts as the network coordinator, granting transmission opportunities to the other CSN nodes of the network.
 17 A CSN physically is a shared medium, in that a CSN node has a single physical port connected to the half-
 18 duplex medium, but is logically a fully connected one-hop mesh network, in that every CSN node can
 19 transmit frames to every other CSN node over the shared medium.

20 A CSN supports two types of transmission: unicast transmission for point-to-point (CSN node-to-node)
 21 transmission and multicast/broadcast transmission for point-to-multipoint (CSN node-to-other/all-nodes)
 22 transmission. Figure 16-1 illustrates a CSN acting as a backbone for PTP Instances.

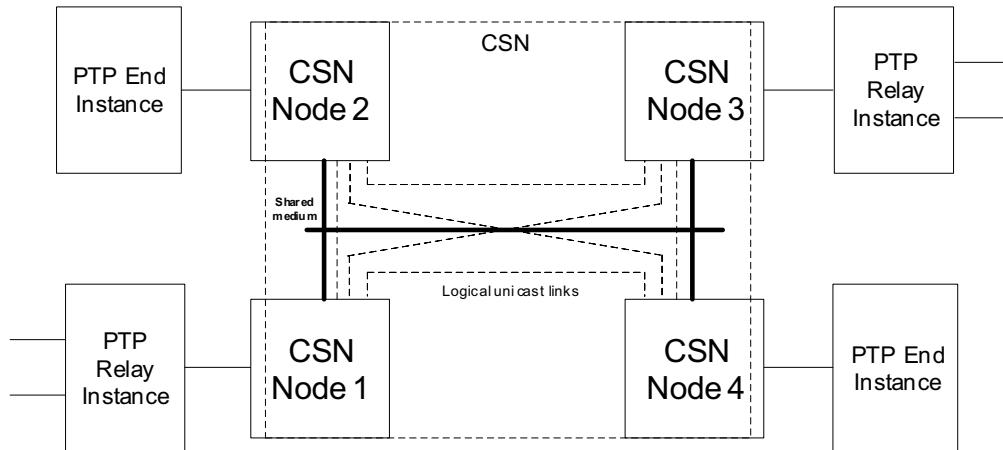


Figure 16-1—Example of CSN backbone in a TSN LAN

23 NOTE—In this clause, the term *node* is used to refer to a CSN node (i.e., it does not refer to a PTP Relay Instance or
 24 PTP End Instance). A CSN node is a 2-port PTP Relay Instance that forwards data packets between a segment external
 25 to the CSN (which can connect to an upstream or downstream PTP Instance) and the CSN, all at the data link layer.
 26 Nonetheless, to avoid confusion the term *node* is usually preceded by *CSN*, except when it is obvious that CSN nodes are
 27 being referenced.

16.3 Layering for CSN links

2 One PortSync entity and one MD entity are together associated with each CSN logical port (CSN node-to-node link) as illustrated in Figure 16-2. The PortSync entities is described in 10.1.2. The MD entity translates media-independent primitives to MD primitives as necessary for communicating synchronized time over the CSN links. The CSN MD entity shall implement the MDSyncSendSM and MDSyncReceiveSM states machines of 11.2.14 and 11.2.15.

7 The CSN MD entity either implements the MDPdelayReq and MDPdelayResp state machines of 11.2.19 and 11.2.20 to measure the propagation delay on a CSN link, or measures it through a CSN-native method 8 and populates the variables described in 16.4.3.3.

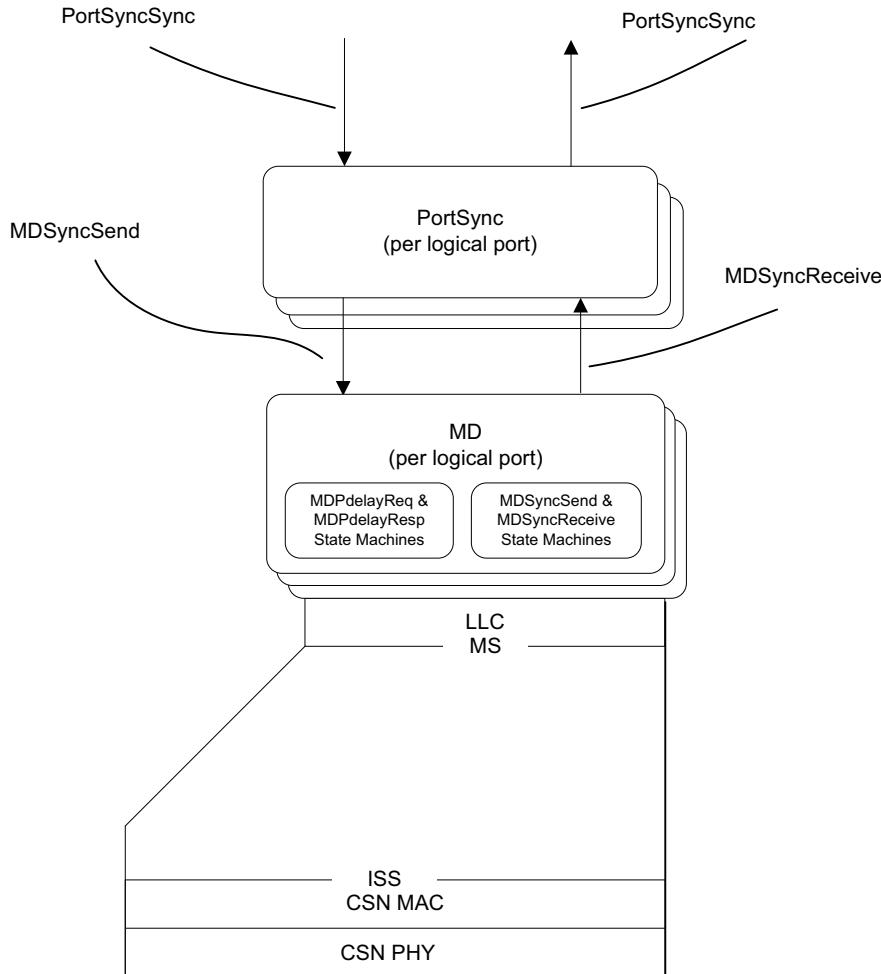


Figure 16-2—Media-dependent and lower entities in CSN nodes

1 16.4 Path delay measurement over a CSN backbone

2 16.4.1 General

3 The Path Delay over a CSN backbone is calculated for the following path types:

- 4 a) Between the upstream PTP Relay Instance and the ingress CSN node
- 5 b) Between the ingress and egress CSN nodes
- 6 c) Between the egress CSN node and the downstream PTP Instance (PTP Relay Instance or PTP End
- 7 Instance)

8 To maintain the synchronization, residence time on each PTP Instance and the propagation delay between
 9 PTP Instances is measured, requiring precise timestamping on both CSN node ingress and egress ports as
 10 illustrated in Figure 16-4 (the numbered paths in Figure 16-4 refer to the numbered paths in Figure 16-3). In
 11 Figure 16-4, ti_1 is the syncEventEgressTimestamp for the Sync message at the upstream PTP Relay
 12 Instance, ti_2 is the syncEventIngressTimestamp for the Sync message at the ingress CSN time-aware node,
 13 te_1 is the syncEventEgressTimestamp for the Sync message at the egress CSN time-aware node, and te_2 is
 14 the syncEventIngressTimestamp for the Sync message at the downstream PTP Relay Instance or PTP End
 15 Instance.

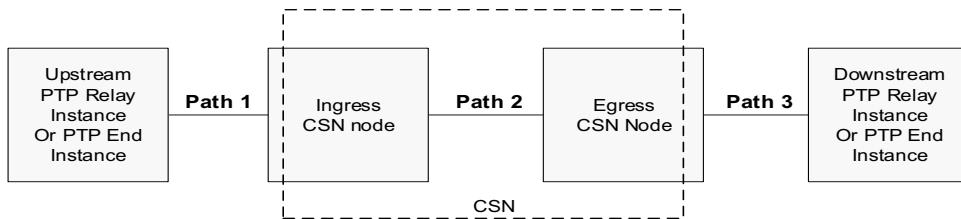


Figure 16-3—Path types over CSN as IEEE 802.1AS backbone

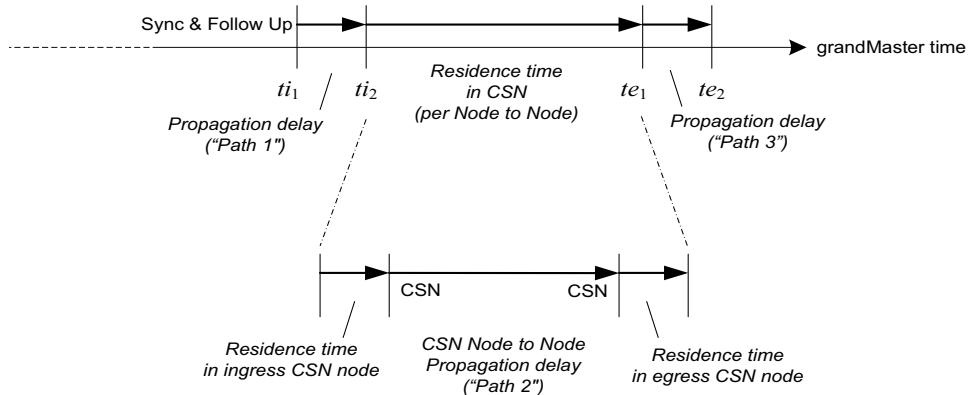


Figure 16-4—Propagation delay and residence time over a CSN backbone

16 16.4.2 Path delay measurement between CSN node and neighbor PTP Instance

17 The path delay measurement between a CSN node and a neighbor PTP Instance is made as specified for
 18 the respective medium. This path delay measurement is made for the link between the CSN node and the
 19 neighbor PTP Instance.

1 16.4.3 Path delay measurement between CSN nodes

2 16.4.3.1 General

3 The path delay between the two nodes of a CSN is the propagation delay for the logical link that connects
 4 those two nodes. The method of measuring the path delay between two CSN nodes has two variations,
 5 which are described in 16.4.3.2 and 16.4.3.3. The specific method to be used for a specific link technology is
 6 specified in 16.6.

7 16.4.3.2 Path delay measurement without network clock reference

8 Each CSN node has a free-running local clock. The path delay measurement uses the peer-to-peer delay
 9 mechanism protocol, messages, and state machines described in Clause 11 for full-duplex point-to-point
 10 links, as illustrated by Figure 16-5. The criteria of 11.2.17 for determining whether the peer-to-peer delay
 11 mechanism is **transport-specific peer-to-peer delay mechanism** or the CMLDS apply here.

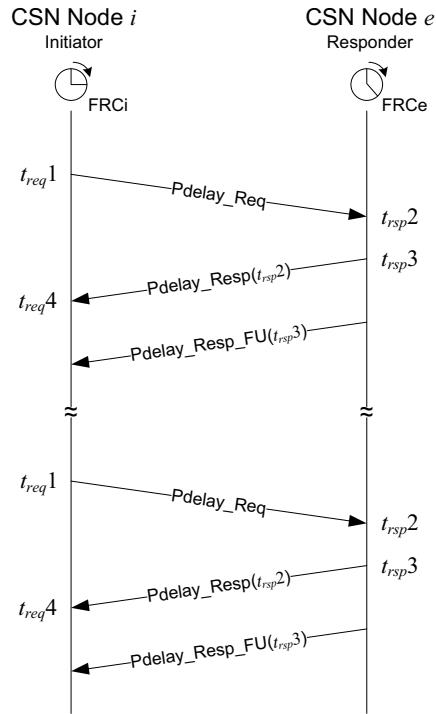


Figure 16-5—CSN node-to-node path delay measurement

12 The computation of the neighborRateRatio and meanLinkDelay between two CSN nodes is done using the
 13 timestamps at the initiator and information conveyed in the successive Pdelay_Resp and
 14 Pdelay_Resp_Follow_Up messages. Any scheme that uses this information is acceptable, as long as the
 15 performance requirements of B.2.4 are met. As one example, the neighborRateRatio is computed as the ratio
 16 between a time interval measured by the local clock of the responder and its associated time interval
 17 measured by the local clock of the initiator, using a set of received Pdelay_Resp and
 18 Pdelay_Resp_Follow_Up messages and a second set of received Pdelay_Resp and
 19 Pdelay_Resp_Follow_Up messages some number of Pdelay_Req message transmission intervals later, i.e.,
 20 as shown in Equation (16-1).

$$1 \quad \frac{(t_{rsp}3)_N - (t_{rsp}3)_0}{(t_{req}4)_N - (t_{req}4)_0} \quad (16-1)$$

2 where

- 3 $(t_{rsp}3)_k$ is the time relative to the local clock of the responder that the k^{th} Pdelay_Resp message is sent
- 4 $(t_{req}4)_k$ is the time relative to the local clock of the initiator that the k^{th} Pdelay_Resp message is
- 5 received
- 6 N is the number of Pdelay_Req message transmission intervals separating the first set of received
- 7 Pdelay_Resp and Pdelay_Resp_Follow_Up messages and the second set

8 The successive sets of received Pdelay_Resp and Pdelay_Resp_Follow_Up messages are indexed from 0 to
9 N with the first set indexed 0. The meanLinkDelay between the PTP Instance and the CSN node is computed
10 as shown in Equation (16-2).

$$11 \quad \frac{(t_{req}4 - t_{req}1)r - (t_{rsp}3 - t_{rsp}2)}{2} \quad (16-2)$$

12 where

- 13 r is equal to neighborRateRatio
- 14 $t_{req}1$ is the time relative to the local clock of the initiator that the Pdelay_Req message for this message
- 15 exchange is sent
- 16 $t_{rsp}2$ is the time relative to the local clock of the responder that the Pdelay_Req message for this message
- 17 exchange is received
- 18 $t_{rsp}3$ is the time relative to the local clock of the responder that the Pdelay_Resp message for this
- 19 message exchange is sent
- 20 $t_{req}4$ is the time relative to the local clock of the initiator that the Pdelay_Resp message for this message
- 21 exchange is received

22 NOTE—The difference between mean propagation delay relative to the Grandmaster Clock time base and relative to the
23 time base of the CSN node at the other end of the attached link (i.e., the responder CSN node) is usually negligible. To
24 see this, note that the former can be obtained from the latter by multiplying the latter by the ratio of the Grandmaster
25 Clock frequency to the frequency of the LocalClock entity of the CSN node at the other end of the link. This ratio differs
26 from 1 by 200 ppm or less. For example, for a worst-case frequency offset of the LocalClock entity of the CSN node at
27 the other end of the link, relative to the Grandmaster Clock, of 200 ppm, and a measured propagation time of 100 ns, the
28 difference in mean propagation delay relative to the two time bases is 20 ps.

29 Although the propagation delay between two CSN nodes is constant, a Pdelay_Req message is still sent
30 periodically by each CSN node to each other active CSN node to measure the neighborRateRatio between
31 the node and each other node. Each CSN node shall implement the state machines described in 11.2.19 and
32 11.2.20.

33 16.4.3.3 Native CSN path delay measurement

34 Some CSN technologies feature a native mechanism that provides a path delay measurement with accuracy
35 similar to the accuracy the peer delay protocol provides. For these CSNs, the path delay can be provided
36 using the native measurement method rather than using the Pdelay protocol defined in 11.2.19 and 11.2.20.
37 Such a situation is described in more detail as follows. The CSN MD entity populates the following per-PTP
38 Port and MD-entity global variables (described respectively in 10.2.5 and 11.2.13) as indicated:

- 39 — asCapable (10.2.5.1) is set to TRUE.
- 40 — neighborRateRatio (10.2.5.7) is set to the value provided by the native CSN measurement.
- 41 — meanLinkDelay (10.2.5.8) is set to the value provided by the native CSN measurement.
- 42 — computeNeighborRateRatio (10.2.5.10) is set to FALSE.

- 1 — computeMeanLinkDelay (10.2.5.11) is set to FALSE.
- 2 — isMeasuringDelay (11.2.13.6) is set to TRUE to indicate that the CSN MD entity is measuring path delay (in this case, using its internal mechanism).
- 3 — domainNumber (8.1) is set to the **domainNumber** of this gPTP domain.

5 16.4.3.4 Intrinsic CSN path delay measurement

6 If the CSN features a native mechanism that causes each CSN node's local clock to be fully synchronized to the local clocks of other nodes of the CSN such that the synchronized CSN time complies with the requirements specified in B.1, the CSN nodes need not implement the path delay mechanism but rather can treat the path delay as part of the residence time of the distributed system. The propagation of the Sync messages in this case is described in 16.5.3.

11 16.5 Synchronization messages

12 16.5.1 General

13 The CSN shall propagate synchronized time over the CSN to CSN end stations and to downstream non-CSN links, using Sync (and, if the message is two-step, the associated Follow_Up) messages, as illustrated in Figure 16-6.

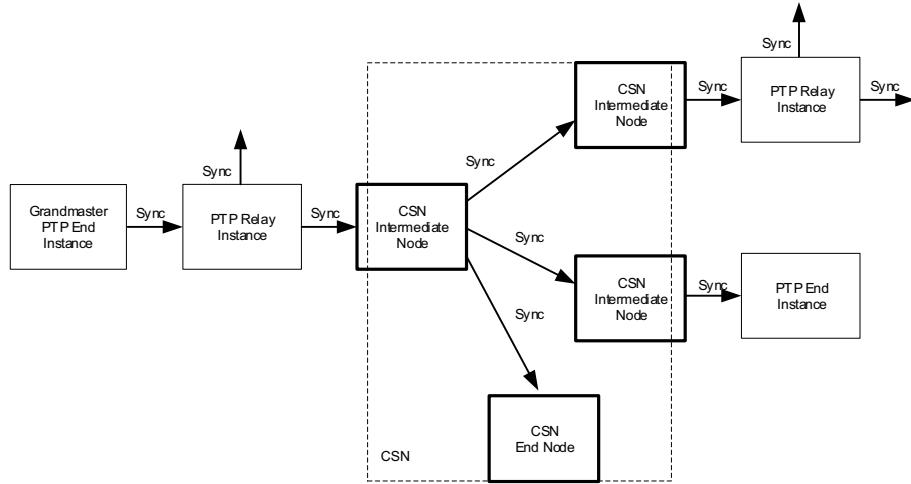


Figure 16-6—IEEE 802.1AS Sync message propagation over the CSN backbone

16 Once the path delays have been measured (a) between the upstream PTP Relay Instance or PTP End Instance and the ingress CSN node, (b) between the CSN nodes, and (c) between the egress CSN node and the downstream PTP Relay Instance or PTP End Instance, the CSN backbone can propagate the synchronization information received at its boundary nodes.

20 As with path delay measurements, various CSN technologies choose various methods for propagating time. 21 These methods are described in 16.5.2 and 16.5.3.

22 16.5.2 Synchronization message propagation on CSN without network reference clock

23 If the CSN does not feature a native mechanism that synchronizes the CSN node local clocks to each other or to a reference (i.e., to synchronize the CSN time in compliance with the requirements specified in B.1),

1 the CSN local clocks at CSN ingress and CSN egress nodes are considered independent, free-running
2 clocks.

3 In this case, synchronization over the CSN links uses the Sync and Follow_Up protocol (or only Sync if
4 optional one-step processing is used), messages, and state machines specified for full-duplex point-to-point
5 links in 10.2.8, 10.2.12, 11.2.14, and 11.2.15. Individual CSN link technologies may specify media-specific
6 encapsulation of gPTP event messages. See Table 16-2 for selection of options per link technology.

7 One PortSync and one MD entity is instantiated per logical port (i.e., per CSN link). A CSN node behaves
8 equivalently to a PTP Instance. Sync and, in the two-step case, Follow_Up messages are either transmitted
9 using unicast on each link or broadcasted. However, if Sync and Follow_Up messages are broadcasted, the
10 following apply:

- 11 — The Sync and Follow_Up messages are broadcast with the same port number used to broadcast
12 Announce messages.
- 13 — All PortSync/MD entity pairs except one set their logSyncInterval attribute (see 10.7.2.3) to 127 so
14 that they do not generate any Sync messages.
- 15 — A dynamic selection of the MD entity that broadcasts the Sync message is needed (a CSN node can
16 dynamically leave the CSN). The dynamic selection algorithm is implementation specific and out of
17 the scope of this standard.

18 **16.5.3 Synchronization message propagation on a CSN with network reference clock**

19 **16.5.3.1 General**

20 If the CSN features a native mechanism that allows the CSN node's local clocks to be fully synchronized to
21 each other in a way that complies with the requirements specified in B.1, it is possible to simplify the path
22 delay mechanism, as described below. This method is an alternative to 16.5.2. Individual CSN link
23 technologies may specify media-specific encapsulation of gPTP event messages. See Table 16-2 (in 16.6.1)
24 for selection of options per link technology.

25 Sync messages are timestamped when they are:

- 26 — Received at the ingress CSN node's PTP Port (syncEventIngressTimestamp) and
- 27 — Transmitted at the egress CSN node's PTP Port (syncEventEgressTimestamp).

28 The elapsed time between the egress and ingress timestamps is computed as the CSN residence time. In this
29 scheme, the Sync message handling is split between the MDSyncSendSM state machine in the CSN ingress
30 node and the MDSyncReceiveSM state machine in the CSN egress node as described in 16.5.3.2 and
31 16.5.3.3.

32 The reference plane for a CSN port and the path delay measurement method are specific to the type of CSN
33 technology, and are defined in Table 16-2.

34 **16.5.3.2 CSN ingress node**

35 **16.5.3.2.1 General**

36 The CSN ingress node timestamps Sync messages received from the upstream PTP Instance and compute
37 the upstreamTxTime as described in item f) of 11.2.14.2.1.

38 In addition, the setFollowUp function of the MDSyncSendSM state machine [see item a) in 11.2.15.2.3] is
39 modified as follows:

- 1 a) The quantity $rateRatio \times (syncEventEgressTimestamp - upstreamTxTime)$ is not added to the
 2 followUpCorrectionField of the Follow_Up message (or the Sync message in the one-step case) to
 3 be transmitted by the ingress to the egress CSN node
 4 b) The CSN TLV (see 16.5.3.2.2) is appended to the Follow_Up message (or to the Sync message in
 5 the one-step case) transmitted by the ingress to the egress CSN node.

6 16.5.3.2.2 CSN TLV

7 16.5.3.2.2.1 General

8 The fields of the CSN TLV are specified in Table 16-1 and 16.5.3.2.2.2 through 16.5.3.2.2.10. This TLV is a
 9 standard organization extension TLV for the Follow_Up message (or the Sync message in the one-step
 10 case), as specified in 14.3 of IEEE Std 1588-2019. This TLV is not allowed to occur before the Follow_Up
 11 information TLV (see 11.4.4.3).

Table 16-1—CSN TLV

Bits	Octets	Offset from start of TLV
7 6 5 4 3 2 1 0		
tlvType	2	0
lengthField	2	2
organizationId	3	4
organizationSubType	3	7
upstreamTXTime	12	10
neighborRateRatio	4	22
meanLinkDelay	12	26
delayAsymmetry	12	38
domainNumber	1	50

12 16.5.3.2.2.2 tlvType (Enumeration16)

13 The value of the tlvType field is 0x3.

14 NOTE—This value indicates the TLV is a vendor and standard organization extension TLV, as specified in 14.3.2.1 and
 15 Table 50 of IEEE Std 1588-2019. The tlvType is specified in that standard as ORGANIZATION_EXTENSION with a
 16 value of 0x3.

17 16.5.3.2.2.3 lengthField (UInteger16)

18 The value of the length is 46.

1 **16.5.3.2.2.4 organizationId (Octet3)**

2 The value of organizationId is 00-80-C2.

3 **16.5.3.2.2.5 organizationSubType (Enumeration24)**

4 The value of organizationSubType is 3.

5 **16.5.3.2.2.6 upstreamTxTime (UScaledNs)**

6 The computed upstreamTxTime value is described in item f) of 11.2.14.2.1.

7 **16.5.3.2.2.7 neighborRateRatio (Integer32)**

8 The neighborRateRatio value is described in 10.2.5.7.

9 **16.5.3.2.2.8 meanLinkDelay (UScaledNs)**

10 The meanLinkDelay value is described in 10.2.5.8.

11 **16.5.3.2.2.9 delayAsymmetry (ScaledNs)**

12 The delayAsymmetry value is described in 10.2.5.9.

13 **16.5.3.2.2.10 domainNumber (UInteger8)**

14 This parameter is the **domainNumber** of this gPTP domain.

15 **16.5.3.3 CSN egress node**

16 The CSN egress port sets neighborRateRatio to 1 and meanLinkDelay to 0 for its CSN port.

17 The CSN egress port modifies the function setMDSyncReceive of the MDSyncReceiveSM state machine
18 [item f) in 11.2.14.2.1] for the port attached to the CSN link entity to extract upstreamTxTime,
19 meanLinkDelay, and neighborRateRatio from the respective fields of the CSN TLV in the Follow_Up
20 message (or Sync message in the one-step case) received from the CSN ingress node.

21 The CSN egress port also modifies the ClockTimeReceiverSync state machine (see 10.2.13) to get the
22 upstreamTxTime, meanLinkDelay, neighborRateRatio, and delayAsymmetry values from the respective
23 fields of the CSN TLV in the Follow_Up message (or Sync message in the one-step case) received from the
24 CSN ingress node.

25 The CSN egress node removes the CSN TLV from the Follow_Up message (or Sync message in the one-
26 step case) received from the CSN ingress node and transmits the message to the downstream PTP Instance.

1 16.6 Specific CSN requirements

2 16.6.1 General

3 The reference plane for a CSN port is specific to the type of CSN technology and is defined in Table 16-2.

Table 16-2—Definitions and option selections per link technology

CSN link technology	Reference plane	Path Delay measurement	gPTP event message encapsulation
Multimedia over Coax Alliance (MoCA) v2.0	The first bit of an Event message crossing to and from the MoCA CTC domain.	MoCA Ranging Protocol (method of either 16.4.3.3 or 16.4.3.4)	Encapsulated in control frames as described in the MoCA MAC/PHY Specification v2.0.
ITU-T G.hn (SG15)	The first bit of an Event message crossing the A-interface (see 16.6.3).	peer-to-peer mechanism as defined in 16.4.3.2.	None

4 16.6.2 MoCA-specific behavior

5 The MoCA network is a CSN. The non-MoCA port of a CSN time-aware node behaves as a PTP Port, which might or might not use the MoCA channel time clock (CTC) for timestamping event messages. If the non-MoCA port of the CSN time-aware node uses a different clock than the MoCA CTC, then the CSN time-aware node reconciles the non-CTC timestamp with the CTC time.

9 Sync messages shall be timestamped using the CTC when they cross:

- 10 — The MoCA ingress node's timestamp reference plane (`syncEventIngressTimestamp`) and
 11 — The egress CSN node's reference plane (`syncEventEgressTimestamp`).

12 The elapsed time between the egress and ingress timestamps, $\text{syncEventIngressTimestamp} - \text{syncEventEgressTimestamp}$, is computed as the MoCA residence time.

14 The MoCA port whose PTP Port state is `TimeTransmitterPort` propagates the Sync and Follow_Up messages (or only the Sync message in the one-step case) as described in 16.5.3. The CSN TLV values of the Follow_Up message sent over the MoCA network are computed using the LocalClock, i.e., the MoCA CTC.

17 IEEE 802.1AS frames shall be transmitted over the MoCA network as MoCA control frames as described in 18 the MoCA MAC/PHY Specification v2.0.

19 NOTE—The CTC is specified in the MoCA MAC/PHY Specification v2.0.

1 16.6.3 ITU-T G.hn-specific behavior

2 A port of a PTP Instance that includes one or more ITU-T G.hn ports shall behave as defined in 16.3, 16.4,
3 and 16.5; however, for aspects where more than one behavior or option is described, the system behaves as
4 defined in Table 16-2.

5 ITU-T G.hn defines a 32-bit timestamp (which is placed in the TSMP field of a G.hn encapsulation header).

6 This timestamp, as described in Table 16-2, is captured each time an Event message is transmitted or
7 received by an ITU-T G.hn port and is used for all gPTP event messages, including SYNC and PDELAY
8 messages.

9 16.7 Grandmaster PTP Instance capability

10 Each CSN Node may be grandmaster capable to allow a CSN node to act as the Grandmaster PTP Instance
11 either for a homogeneous CSN or for a heterogeneous network.

12 The Announce messages are either sent via unicast on each link or broadcasted. However, if Announce
13 messages are broadcasted, the Announce message shall use the same port number as used by the Sync and
14 Follow_Up messages (or only the Sync message in the one-step case) by a single PortSync/MD entity pair
15 on the PTP Port. This is accomplished by setting the logAnnounceInterval attribute (see 10.7.2.2) to 127 for
16 all but one PortSync/MD entity pair so that they do not generate any Announce messages.

17 16.8 CSN clock and node performance requirements

18 The CSN clock performance complies with the requirements specified in B.1. The CSN node performance
19 complies with the requirements specified in B.2.4 and should follow the recommendations of B.2.2 and
20 B.2.3.

¹ 17. YANG data model

² <<Editor's note: The description of the YANG models refer to ASdm and ASdn. Also, figures 17-1, 17-
³ 2, 17-3 and 17-4, and Table 17-1 refer to 802.1ASdn and 802.1ASdm. Do we need to change those, as
⁴ 802.1ASdn and 802.1ASdm are now part of the 802.1AS revision? Do the ASdn and ASdm YANG
⁵ models need to be combined? Is there anything else that need to be updated? Comments to address
⁶ these points are welcome.>>

⁷ YANG (IETF RFC 7950) is a data modeling language used to model configuration data and state data for
⁸ remote network management protocols. YANG-based remote network management protocols include the
⁹ Network Configuration Protocol (NETCONF) (IETF RFC 6242 [B13a]) and RESTCONF (IETF RFC 8040
¹⁰ [B13e]). Each remote network management protocol uses a specific encoding on the wire, such as XML or
¹¹ JSON. A YANG module specifies the organization and rules for the management data, and a mapping from
¹² YANG to the specific encoding enables the data to be understood correctly by both client (e.g., network
¹³ manager) and server (e.g., PTP Instances).

¹⁴ This clause specifies the YANG data model for IEEE Std 802.1AS.

¹⁵ This clause:

- ¹⁶ a) Introduces the organization of the data models, including the relationship with other standards (17.1)
- ¹⁷ b) Provides an overview of the hierarchy of the data models using a representation similar to the
Unified Modeling Language (UML) (17.2)
- ¹⁹ c) Summarizes the structure of the YANG data model (17.3)
- ²⁰ d) Reviews security considerations (17.4)
- ²¹ e) Provides a schema tree as an overview of the YANG module (17.5)
- ²² f) Specifies the YANG module (17.5.2)

²³ 17.1 YANG framework

²⁴ The YANG framework applies hierarchy in the following areas:

- ²⁵ a) The Uniform Resource Name (URN), as specified in IEEE Std 802d-2017.
- ²⁶ b) The YANG objects form a hierarchy of configuration and operational data structures that define the
YANG model.

²⁸ Clause 14 specifies the information model for management of this standard. The data model for a specific
²⁹ management mechanism is derived from the information model. Since YANG-based protocols are an
³⁰ example of a management mechanism, the YANG data model of this clause is derived from Clause 14.

³¹ NOTE—The MIB modules specified in Clause 15 were also derived from Clause 14. Consequently, the capabilities and
³² structure of the YANG data models are aligned with those represented by the MIB. However the YANG data model has
³³ not been derived from the MIB, and there has been no attempt to include data or modeling constructs that might appear
³⁴ in the MIB but not in the information model.

³⁵ The information model in Clause 14 is organized as a hierarchy of data sets. Each data set contains one or
³⁶ more related members (items of data that can be read or written). In the context of YANG, each data set is
³⁷ represented as a YANG “container,” and each member is represented as a YANG “leaf.”

³⁸ 17.1.1 Relationship to the IEEE Std 1588 data model

³⁹ The YANG data models specified in this standard are based on, and augment, those specified in IEEE
⁴⁰ Std 1588. In particular the ieee802-dot1as-gptp.yang module imports the ieee1588-ptp-tt module as a whole,

1 augmenting that module as necessary to meet the requirements of this standard. In addition, the
 2 [ieee802-dot1as-hs.yang](#) module imports the [ieee1588-ptp-tt](#) and [ieee802-dot1as-gptp](#) modules as a whole,
 3 augmenting those modules as necessary to meet the requirements of this standard.

4 Some of the data sets in Clause 14 (e.g., defaultDS) are derived from IEEE Std 1588, and some of the data
 5 sets are unique to IEEE Std 802.1AS (i.e., not derived from IEEE Std 1588). For each data set in Clause 14
 6 that is derived from IEEE Std 1588, a portion of the members are derived from IEEE Std 1588, and the
 7 remaining members are unique to IEEE Std 802.1AS. For the members that are derived from IEEE Std 1588,
 8 the specifications in both standards are analogous (same name, data type, semantics, etc.).

9 The YANG data model for IEEE Std 1588-2019 is published as amendment IEEE Std 1588e. The YANG
 10 module of IEEE Std 1588e ([ieee1588-ptp-tt.yang](#)) contains the hierarchy (tree) of data sets and their
 11 members.

12 The YANG module of this clause ([ieee802-dot1as-gptp.yang](#) and [ieee802-dot1as-hs.yang](#)) use the YANG
 13 “import” statement to import the YANG module of IEEE Std 1588e. This effectively uses the IEEE Std 1588
 14 YANG tree as the foundation of the IEEE Std 802.1AS YANG tree. By importing the tree and its data set
 15 containers, all members from Clause 14 that are derived from IEEE Std 1588 are also imported.

16 The core of the YANG module for IEEE Std 802.1AS consists of YANG “augment” statements, used to add
 17 members to the tree that are unique for IEEE Std 802.1AS.

18 NOTE—IETF RFC 8575 [B13g] is the standard YANG data model for IEEE Std 1588-2008. The YANG data model of
 19 IEEE Std 1588e is effectively a newer version of RFC 8575. Therefore, the YANG module of RFC 8575 is not imported
 20 by the YANG module of this clause.

21 **17.2 IEEE 802.1AS YANG data model**

22 This clause uses a UML-like representation to provide an overview of the hierarchy of the IEEE
 23 Std 802.1AS YANG data model.

24 A representation of the management model is provided in Figure 17-1 through Figure 17-4. The purpose of
 25 the diagrams is to express the model design in a concise manner. The structure of the representation shows
 26 the name of the object followed by a list of properties for the object. The properties indicate their type and
 27 accessibility. The representation is meant to express simplified semantics for the properties. It is not meant
 28 to provide the specific datatype used to encode the object in either MIB or YANG. In the representation, a
 29 box with a white background represents information that comes from sources outside of this IEEE standard.
 30 A box with a gray background represents objects that are defined by this IEEE standard.

31 NOTE 1—OMG® UML 2.5²¹ [B24a] conventions together with C++ language constructs are used in this clause as a
 32 representation to convey model structure and relationships.

33 NOTE 2—This standard specifies YANG for Clause 14 of this standard. There are optional features in the YANG
 34 module of IEEE Std 1588 that are not specified in Clause 14, and are therefore not shown in the figures of this subclause.
 35 If optional IEEE Std 1588 YANG features are implemented, conformance is specified by IEEE Std 1588.

36 For all figures, Clause 14 data that is imported from the [ieee1588-ptp-tt.yang](#) module is shown in white, and
 37 Clause 14 data in augments of [ieee802-dot1as-gptp.yang](#) is shown in gray.

38 Figure 17-1 provides an overview of the IEEE Std 802.1AS YANG tree. The top level instance list provides
 39 the list of one or more PTP Instances, each with data sets. For each PTP Instance, port-ds-list provides the

²¹ OMG® is a registered trademark of the Object Management Group, Inc.

1 list of one or more PTP Ports, each with data sets. The common-services apply to all PTP Instances,
2 including the Common Mean Link Delay Service (cmlds).

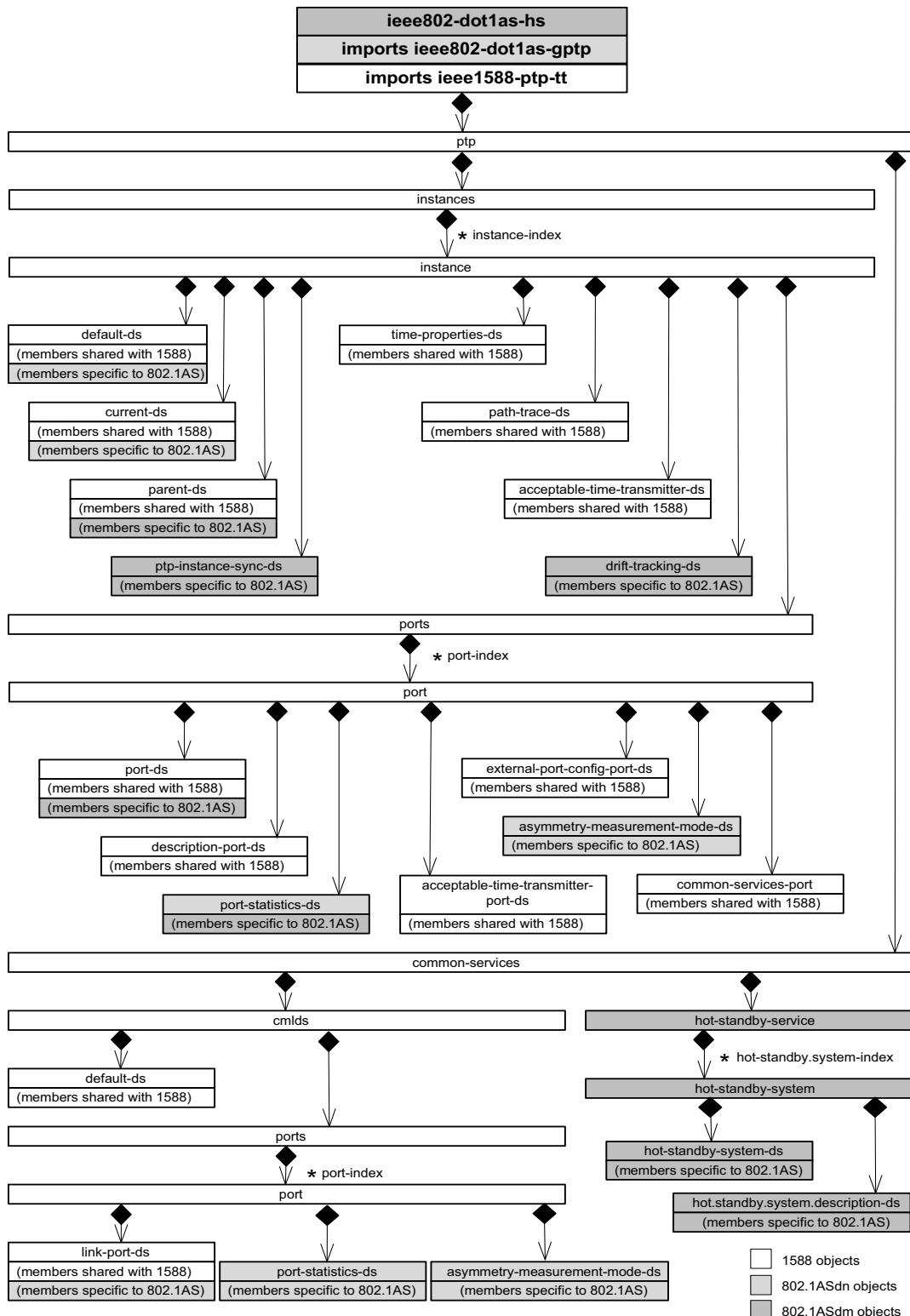


Figure 17-1—Overview of YANG tree

1 Figure 17-2 provides detail for the data sets of each PTP Instance, including each data set member.

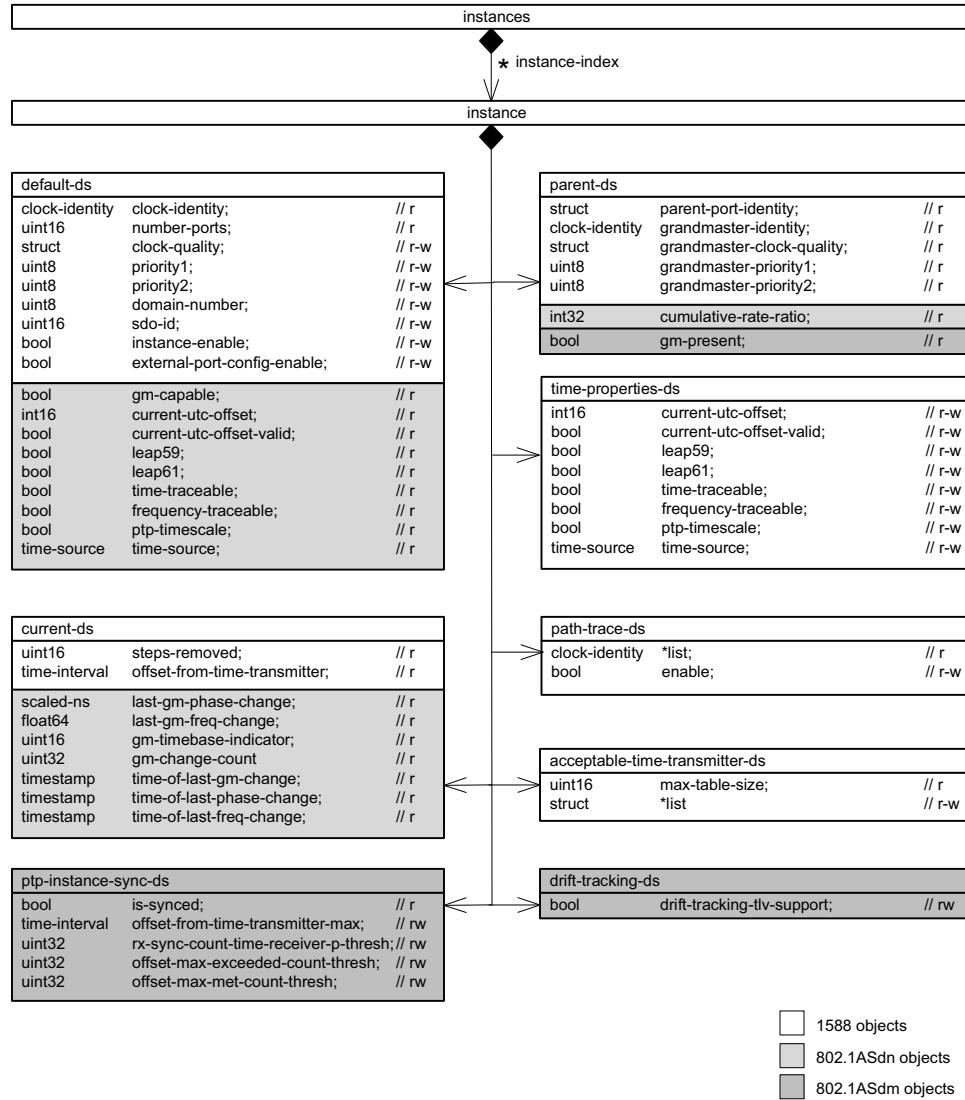


Figure 17-2—PTP Instance detail

2 Figure 17-3 provides detail for the data sets of each PTP Port, including each data set member.

3 NOTE 3—Subclause 14.8.4 specifies `ptpPortEnabled` (`ptp-port-enabled`), which is provided in YANG as the
 4 semantically equivalent node in `ieee1588-ptp-tt` named `port-enable` (in `port-ds` of Figure 17-3). Subclause 14.8.15
 5 specifies `mgtSettableLogAnnounceInterval` (`mgt-settable-log-announce-interval`), which is provided in YANG as the
 6 semantically equivalent node in `ieee1588-ptp-tt` named `log-announce-interval` (in `port-ds` of Figure 17-3). Subclause
 7 14.8.20 specifies `mgtSettableLogSyncInterval` (`mgt-settable-log-sync-interval`), which is provided in YANG as the
 8 semantically equivalent node in `ieee1588-ptp-tt` named `log-sync-interval` (in `port-ds` of Figure 17-3).

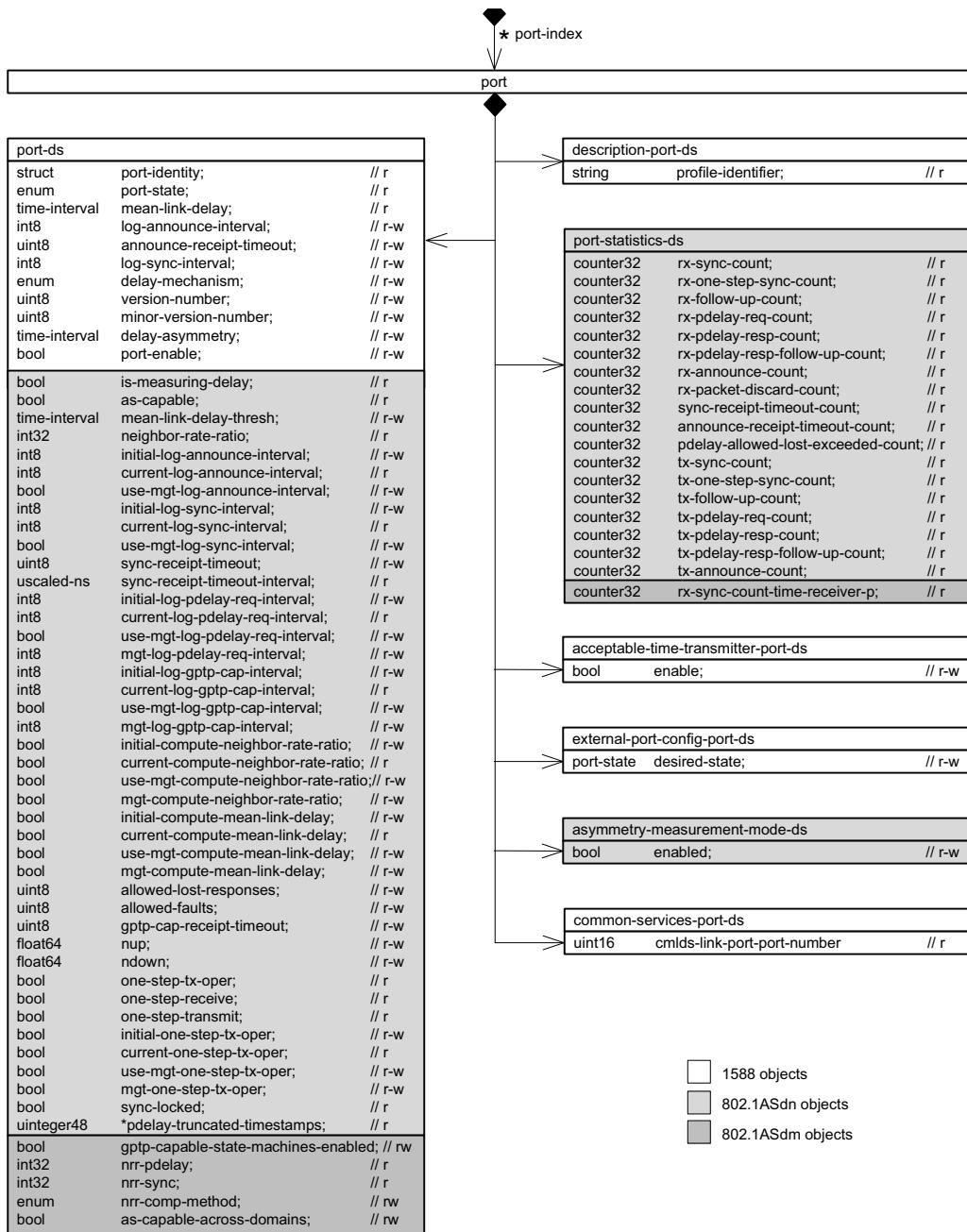


Figure 17-3—PTP Port detail

1 Figure 17-4 provides detail for the common services, including each data set member. The Common Mean
 2 Link Delay Service (cmlds) has a data set for the service itself (default-ds), and data sets for each Link Port.
 3 The Hot Standby Service has data sets for each HotStandbySystem.

4 NOTE 4—Subclause 14.16.9 specifies neighborRateRatio (neighbor-rate-ratio), which is provided in YANG as the
 5 semantically equivalent node in ieee1588-ptp-tt named scaled-neighbor-rate-ratio (in link-port-ds of Figure 17-4).

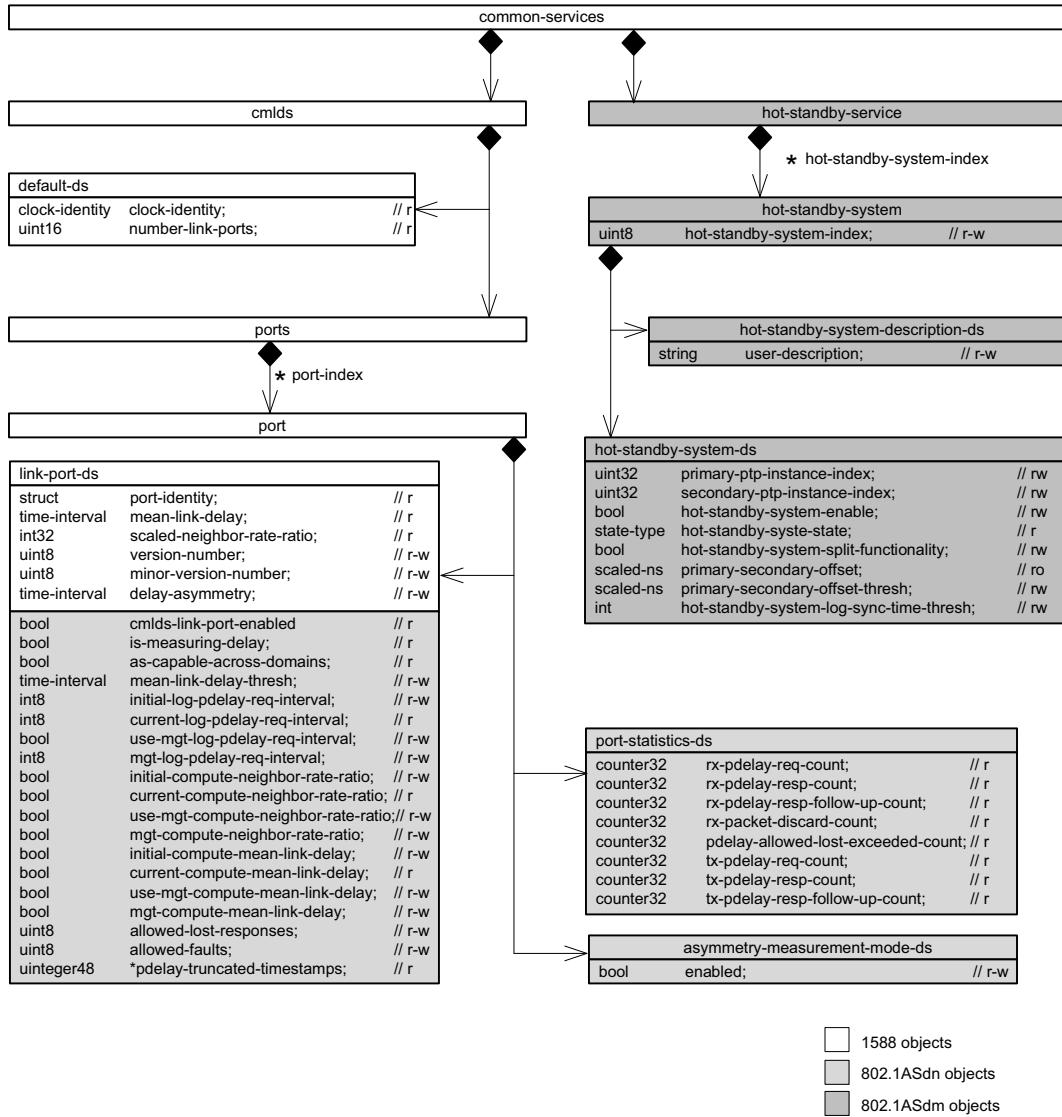


Figure 17-4—Common services detail

6 17.3 Structure of the YANG data model

7 The YANG data model specified by this standard uses the YANG modules summarized in Table 17-1.

1 In the YANG module definitions, if any discrepancy between the “description” text and the corresponding
 2 definition in any other part of this standard occurs, the definitions outside this clause (Clause 17) take
 3 precedence.

Table 17-1—Summary of the YANG modules

Module	Managed functionality	YANG specification notes
ietf-yang-types	Type definitions	IETF RFC 6991—Common YANG Data Types.
ieee1588-ptp-tt	Clause 14	IEEE Std 1588e—MIB and YANG Data Models. IEEE Std 802.1ASdn imports this YANG module as its foundational tree, including a subset of members from Clause 14.
ieee802-dot1as-gptp	Clause 14	IEEE Std 802.1ASdn—YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std 802.1AS.
ieee802-dot1as-hs	Clause 14	IEEE Std 802.1ASdm - YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std 802.1ASdm.

4 17.4 Security considerations

5 The YANG module specified in this document defines a schema for data that is designed to be accessed via
 6 network management protocols such as NETCONF (IETF RFC 6242 [B13a]) and RESTCONF (IETF
 7 RFC 8040 [B13e]). NETCONF and RESTCONF protocols provide the means to secure communication
 8 between client and server, using secure transport layers such as Secure Shell (SSH) (IETF RFC 6242
 9 [B13b]) and Transport Layer Security (TLS) (IETF RFC 7589 [B13d]).

10 It is the responsibility of a system’s implementor and administrator to ensure that the protocol entities in the
 11 system that support NETCONF, and any other remote configuration protocols that make use of these YANG
 12 modules, are properly configured to allow access only to those principals (users) that have legitimate rights
 13 to read or write data nodes. This standard does not specify how the credentials of those users are to be stored
 14 or validated.

15 The Network Configuration Access Control Model (IETF RFC 8341 [B13c]) provides the means to restrict
 16 access for particular NETCONF or RESTCONF users to a pre-configured subset of all available NETCONF
 17 or RESTCONF protocol operations and content.

18 There are a number of data sets in this YANG module that contain writable data nodes (IETF RFC 8040
 19 [B13e]), such as:

```

20   /ptp-tt/instances/instance/default-ds
21   /ptp-tt/instances/instance/path-trace-ds
22   /ptp-tt/instances/instance/acceptable-time-transmitter-ds
23   /ptp-tt/instances/instance/ports/port/port-ds
24   /ptp-tt/instances/instance/ports/port/acceptable-time-transmitter-port-ds
25   /ptp-tt/instances/instance/ports/port/external-port-config-port-ds
26   /ptp-tt/instances/instance/ports/port/asymmetry-measurement-mode-ds
27   /ptp-tt/common-services/cmlds/ports/port/link-port-ds
28   /ptp-tt/common-services/cmlds/ports/port/asymmetry-measurement-mode-ds

```

1 Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect
2 on network operations. Specifically, an inappropriate configuration of them can adversely impact a PTP
3 synchronization network. For example, loss of synchronization on a clock, accuracy degradation on a set of
4 clocks, or even breakdown of a whole synchronization network.

5 **17.5 YANG schema tree definitions**

6 The schema tree in this clause is provided as an overview of the YANG module in 17.5.2. The symbols and
7 their meaning are specified in YANG Tree Diagrams (IETF RFC 8340 [B13f]).

8 **17.5.1 Tree diagram for ieee802-dot1as-gptp.yang**

```
9 module: ieee802-dot1as-gptp

10 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:default-ds:
11    +-ro gm-capable?          boolean
12    +-ro current-utc-offset?   int16
13    +-ro current-utc-offset-valid? boolean
14    +-ro leap59?              boolean
15    +-ro leap61?              boolean
16    +-ro time-traceable?      boolean
17    +-ro frequency-traceable? boolean
18    +-ro ptp-timescale?       boolean
19    +-ro time-source?         identityref

20 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:current-ds:
21    +-ro last-gm-phase-change? scaled-ns
22    +-ro last-gm-freq-change?   float64
23    +-ro gm-timebase-indicator? uint16
24    +-ro gm-change-count?      yang:counter32
25    +-ro time-of-last-gm-change? yang:timestamp
26    +-ro time-of-last-phase-change? yang:timestamp
27    +-ro time-of-last-freq-change? yang:timestamp

28 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:parent-ds:
29    +-ro cumulative-rate-ratio? int32
```

```
1 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port/ptp-tt:port-ds:  
2    +-+ro is-measuring-delay?          boolean  
3    +-+ro as-capable?                boolean  
4    +-+rw mean-link-delay-thresh?    ptp-tt:time-interval  
5    +-+ro neighbor-rate-ratio?      int32  
6    +-+rw initial-log-announce-interval? int8  
7    +-+ro current-log-announce-interval? int8  
8    +-+rw use-mgt-log-announce-interval? boolean  
9    +-+rw initial-log-sync-interval?   int8  
10   +-+ro current-log-sync-interval?   int8  
11   +-+rw use-mgt-log-sync-interval?  boolean  
12   +-+rw sync-receipt-timeout?     uint8  
13   +-+ro sync-receipt-timeout-interval? uscaled-ns  
14   +-+rw initial-log-pdelay-req-interval? int8  
15   +-+ro current-log-pdelay-req-interval? int8  
16   +-+rw use-mgt-log-pdelay-req-interval? boolean  
17   +-+rw mgt-log-pdelay-req-interval? int8  
18   +-+rw initial-log-gptp-cap-interval? int8  
19   +-+ro current-log-gptp-cap-interval? int8  
20   +-+rw use-mgt-log-gptp-cap-interval? boolean  
21   +-+rw mgt-log-gptp-cap-interval?   int8  
22   +-+rw initial-compute-neighbor-rate-ratio? boolean  
23   +-+ro current-compute-neighbor-rate-ratio? boolean  
24   +-+rw use-mgt-compute-neighbor-rate-ratio? boolean  
25   +-+rw mgt-compute-neighbor-rate-ratio? boolean  
26   +-+rw initial-compute-mean-link-delay?  boolean  
27   +-+ro current-compute-mean-link-delay?  boolean
```

```

1  +-rw use-mgt-compute-mean-link-delay?      boolean
2  +-rw mgt-compute-mean-link-delay?          boolean
3  +-rw allowed-lost-responses?              uint8
4  +-rw allowed-faults?                     uint8
5  +-rw gtp-cap-receipt-timeout?            uint8
6  +-rw nup?                             float64
7  +-rw ndown?                           float64
8  +-ro one-step-tx-oper?                 boolean
9  +-ro one-step-receive?                boolean
10 +-ro one-step-transmit?               boolean
11 +-rw initial-one-step-tx-oper?        boolean
12 +-ro current-one-step-tx-oper?       boolean
13 +-rw use-mgt-one-step-tx-oper?        boolean
14 +-rw mgt-one-step-tx-oper?           boolean
15 +-ro sync-locked?                   boolean
16 +-ro pdelay-truncated-timestamps*    uinteger48
17 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port:
18   +-rw port-statistics-ds
19     +-ro rx-sync-count?             yang:counter32
20     +-ro rx-one-step-sync-count?  yang:counter32
21     +-ro rx-follow-up-count?     yang:counter32
22     +-ro rx-pdelay-req-count?   yang:counter32
23     +-ro rx-pdelay-resp-count?  yang:counter32
24     +-ro rx-pdelay-resp-follow-up-count? yang:counter32
25     +-ro rx-announce-count?    yang:counter32
26     +-ro rx-packet-discard-count? yang:counter32
27     +-ro sync-receipt-timeout-count? yang:counter32

```

```
1    +-+ro announce-receipt-timeout-count?      yang:counter32
2    +-+ro pdelay-allowed-lost-exceeded-count?  yang:counter32
3    +-+ro tx-sync-count?                      yang:counter32
4    +-+ro tx-one-step-sync-count?            yang:counter32
5    +-+ro tx-follow-up-count?                yang:counter32
6    +-+ro tx-pdelay-req-count?              yang:counter32
7    +-+ro tx-pdelay-resp-count?            yang:counter32
8    +-+ro tx-pdelay-resp-follow-up-count?  yang:counter32
9    +-+ro tx-announce-count?              yang:counter32
10   augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port:
11     +-+rw asymmetry-measurement-mode-ds
12     +-+rw enabled? boolean
13   augment /ptp-tt:ptp/ptp-tt:common-services/ptp-tt:cmlds/ptp-tt:ports/ptp-tt:port/ptp-tt:link-port-ds:
14     +-+ro cmlds-link-port-enabled?        boolean
15     +-+ro is-measuring-delay?          boolean
16     +-+ro as-capable-across-domains?    boolean
17     +-+rw mean-link-delay-thresh?       ptp-tt:time-interval
18     +-+rw initial-log-pdelay-req-interval? int8
19     +-+ro current-log-pdelay-req-interval? int8
20     +-+rw use-mgt-log-pdelay-req-interval? boolean
21     +-+rw mgt-log-pdelay-req-interval?   int8
22     +-+rw initial-compute-neighbor-rate-ratio? boolean
23     +-+ro current-compute-neighbor-rate-ratio? boolean
24     +-+rw use-mgt-compute-neighbor-rate-ratio? boolean
25     +-+rw mgt-compute-neighbor-rate-ratio? boolean
26     +-+rw initial-compute-mean-link-delay? boolean
27     +-+ro current-compute-mean-link-delay? boolean
```

```

1  +-rw use-mgt-compute-mean-link-delay?      boolean
2  +-rw mgt-compute-mean-link-delay?          boolean
3  +-rw allowed-lost-responses?              uint8
4  +-rw allowed-faults?                     uint8
5  +-ro pdelay-truncated-timestamps*        uinteger48
6  augment /ptp-tt:ptp/ptp-tt:common-services/ptp-tt:cmlds/ptp-tt:ports/ptp-tt:port:
7  +-rw port-statistics-ds
8  +-ro rx-pdelay-req-count?                yang:counter32
9  +-ro rx-pdelay-resp-count?               yang:counter32
10 +-ro rx-pdelay-resp-follow-up-count?     yang:counter32
11 +-ro rx-packet-discard-count?           yang:counter32
12 +-ro pdelay-allowed-lost-exceeded-count? yang:counter32
13 +-ro tx-pdelay-req-count?                yang:counter32
14 +-ro tx-pdelay-resp-count?               yang:counter32
15 +-ro tx-pdelay-resp-follow-up-count?     yang:counter32
16 augment /ptp-tt:ptp/ptp-tt:common-services/ptp-tt:cmlds/ptp-tt:ports/ptp-tt:port:
17 +-rw asymmetry-measurement-mode-ds
18 +-rw enabled?   boolean
19

```

20 17.5.2 Tree diagram for ieee802-dot1as-hs.yang

```

21 module: ieee802-dot1as-hs
22
23 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:parent-ds:
24   +-ro gm-present?   boolean
25 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance:
26   +-rw ptp-instance-sync-ds
27   |  +-ro is-synced?            boolean
28   |  +-rw offset-from-time-transmitter-max?  ptp-tt:time-interval
29   |  +-rw rx-sync-count-time-receiver-p-thresh? uint32
30   |  +-rw offset-max-exceeded-count-thresh?    uint32
31   |  +-rw offset-max-met-count-thresh?          uint32
32   +-rw drift-tracking-ds
33     +-rw drift-tracking-tlv-support?   boolean
34 augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port/
35 ptp-tt:port-ds:

```

```

1   +-rw gptp-capable-state-machines-enabled?    boolean
2   +-ro nrr-pdelay?                           int32
3   +-ro nrr-sync?                            int32
4   +-rw nrr-comp-method?                     nrr-comp-method-type
5   +-rw as-capable-across-domains?          boolean
6   augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port/
7 dot1as-gptp:port-statistics-ds:
8     +-ro rx-sync-count-time-receiver-p?  yang:counter32
9   augment /ptp-tt:ptp/ptp-tt:common-services:
10    +-rw hot-standby-service {hot-standby}?
11      +-rw hot-standby-system* [hot-standby-system-index]
12        +-rw hot-standby-system-index           uint8
13        +-rw hot-standby-system-ds
14        |  +-rw primary-ptp-instance-index?    uint32
15        |  +-rw secondary-ptp-instance-index?  uint32
16        |  +-rw hot-standby-system-enable?      boolean
17        |  +-ro hot-standby-system-state?       hot-standby-system-
18 state-type
19        |  +-rw hot-standby-system-split-functionality?  boolean
20        |  +-ro primary-secondary-offset?        dot1as-gptp:scaled-ns
21        |  +-rw primary-secondary-offset-thresh?  dot1as-gptp:scaled-
22 ns
23        |  +-rw hot-standby-system-log-sync-time-thresh?  int8
24        +-rw hot-standby-system-description-ds
25          +-rw user-description?    string
26

```

27 **17.6 YANG modules^{22, 23}**

28 The ieee1588-ptp-tt.yang YANG module²⁴ specified by IEEE Std 1588e serves as the foundation of the
29 YANG module specified in this clause.

30 **17.6.1 Module ieee802-dot1as-gptp.yang**

```

31 module ieee802-dot1as-gptp {
32   yang-version "1.1";
33   namespace urn:ieee:std:802.1AS:yang:ieee802-dot1as-gptp;
34   prefix dot1as-gptp;
35   import ietf-yang-types {
36     prefix yang;
37   }

```

²² Copyright release for YANG modules: Users of this standard may freely reproduce the YANG modules contained in this subclause so that they can be used for their intended purpose.

²³ ASCII versions of the YANG modules are attached to the PDF version of this standard, and can be obtained by Web browser from the IEEE 802.1 Website at <https://1.ieee802.org/yang-modules/>.

²⁴ An ASCII version of the IEEE Std 1588 YANG module can be obtained from <https://github.com/YangModels/yang/tree/main/standard/ieee/published/1588>.

```
1 import ieee1588-ptp-tt {  
2   prefix ptp-tt;  
3 }  
4 organization  
5 "IEEE 802.1 Working Group";  
6 contact  
7 "WG-URL: http://ieee802.org/1/  
8 WG-EMail: stds-802-1-l@ieee.org  
9  
10 Contact: IEEE 802.1 Working Group Chair  
11     Postal: C/O IEEE 802.1 Working Group  
12     IEEE Standards Association  
13     445 Hoes Lane  
14     Piscataway, NJ 08854  
15     USA  
16  
17 E-mail: stds-802-1-chairs@ieee.org";  
18 description  
19 "Management objects that control timing and synchronization for time  
20 sensitive applications, as specified in IEEE Std 802.1AS.  
21  
22 Copyright (C) IEEE (2024). This version of this YANG module is part  
23 of IEEE Std 802.1AS; see the standard itself for full legal notices.";  
24 revision 2024-09-11 {  
25   description  
26   "Published as part of IEEE Std 802.1ASdn-2024. Initial version.";  
27   reference
```

```
1 "IEEE Std 802.1AS - Timing and Synchronization for Time-Sensitive
2 Applications: IEEE Std 802.1AS-2020, IEEE Std 802.1AS-2020/Cor
3 1-2021, IEEE Std 802.1ASdr-2024, IEEE Std 802.1ASdn-2024. IEEE Std
4 1588 - IEEE Standard for a Precision Clock Synchronization Protocol
5 for Networked Measurement and Control Systems: IEEE Std 1588-2019,
6 IEEE Std 1588g-2022, IEEE Std 1588e-2024.";
7 }
8 typedef scaled-ns {
9 type string {
10 pattern "[0-9A-F]{2}(-[0-9A-F]{2}){11}";
11 }
12 description
13 "The IEEE Std 802.1AS ScaledNs type represents signed values of
14 time and time interval in units of  $2^{16}$  ns, as a signed 96-bit
15 integer. Each of the 12 octets is represented as a pair of
16 hexadecimal characters, using uppercase for a letter. Octets are
17 separated by a dash character. The most significant octet is first.";
18 reference
19 "6.4.3.1 of IEEE Std 802.1AS";
20 }
21 typedef uscaled-ns {
22 type string {
23 pattern "[0-9A-F]{2}(-[0-9A-F]{2}){11}";
24 }
25 description
26 "The IEEE Std 802.1AS UScaledNs type represents unsigned values of
27 time and time interval in units of  $2^{16}$  ns, as an unsigned 96-bit
```

```
1   integer. Each of the 12 octets is represented as a pair of
2   hexadecimal characters, using uppercase for a letter. Octets are
3   separated by a dash character. The most significant octet is first.";
4   reference
5   "6.4.3.2 of IEEE Std 802.1AS";
6 }
7 typedef float64 {
8   type string {
9     pattern "[0-9A-F]{2}(-[0-9A-F]{2}){7}";
10 }
11 description
12 "The IEEE Std 802.1AS Float64 type represents IEEE Std 754
13 binary64. Each of the 8 octets is represented as a pair of
14 hexadecimal characters, using uppercase for a letter. Octets are
15 separated by a dash character. The most significant octet is first.";
16 reference
17 "6.4.2 of IEEE Std 802.1AS";
18 }
19 typedef uinteger48 {
20   type uint64 {
21     range "0..281474976710655";
22 }
23 description
24 "48-bit unsigned integer data type.";
25 reference
26 "6.4.2 of IEEE Std 802.1AS";
27 }
```

```
1 augment
2 "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:default-ds" {
3   description
4     "Augment IEEE Std 1588 defaultDS.";
5   leaf gm-capable {
6     type boolean;
7     config false;
8     description
9       "The value is true if the time-aware system is capable of being a
10      Grandmaster, and false if the time-aware system is not capable of
11      being a Grandmaster.";
12   reference
13     "14.2.7 of IEEE Std 802.1AS";
14 }
15 leaf current-utc-offset {
16   when
17     "./current-utc-offset-valid='true'";
18   type int16;
19   config false;
20   description
21     "Offset from UTC (TAI - UTC). The offset is in units of seconds.
22     This leaf applies to the ClockTimeTransmitter entity (i.e., local
23     only, unrelated to a remote GM).";
24   reference
25     "14.2.8 of IEEE Std 802.1AS";
26 }
27 leaf current-utc-offset-valid {
```

```
1 type boolean;
2 config false;
3 description
4 "The value of current-utc-offset-valid shall be true if the value
5 of current-utc-offset is known to be correct, otherwise it shall
6 be false. This leaf applies to the ClockTimeTransmitter entity
7 (i.e., local only, unrelated to a remote GM).";
8 reference
9 "14.2.9 of IEEE Std 802.1AS";
10 }

11 leaf leap59 {
12 type boolean;
13 config false;
14 description
15 "If the timescale is PTP, a true value for leap59 shall indicate
16 that the last minute of the current UTC day contains 59 seconds.
17 If the timescale is not PTP, the value shall be false. This leaf
18 applies to the ClockTimeTransmitter entity (i.e., local only,
19 unrelated to a remote GM).";
20 reference
21 "14.2.10 of IEEE Std 802.1AS";
22 }

23 leaf leap61 {
24 type boolean;
25 config false;
26 description
27 "If the timescale is PTP, a true value for leap61 shall indicate
```

```
1      that the last minute of the current UTC day contains 61 seconds.  
2      If the timescale is not PTP, the value shall be false. This leaf  
3      applies to the ClockTimeTransmitter entity (i.e., local only,  
4      unrelated to a remote GM).";  
5      reference  
6      "14.2.11 of IEEE Std 802.1AS";  
7      }  
8      leaf time-traceable {  
9      type boolean;  
10     config false;  
11     description  
12     "The value of time-traceable shall be true if the timescale is  
13     traceable to a primary reference; otherwise, the value shall be  
14     false. This leaf applies to the ClockTimeTransmitter entity  
15     (i.e., local only, unrelated to a remote GM).";  
16     reference  
17     "14.2.12 of IEEE Std 802.1AS";  
18     }  
19     leaf frequency-traceable {  
20     type boolean;  
21     config false;  
22     description  
23     "The value of frequency-traceable shall be true if the frequency  
24     determining the timescale is traceable to a primary reference;  
25     otherwise, the value shall be false. This leaf applies to the  
26     ClockTimeTransmitter entity (i.e., local only, unrelated to a  
27     remote GM).";
```

```
1   reference
2   "14.2.13 of IEEE Std 802.1AS";
3 }
4 leaf ptptimescale {
5   type boolean;
6   config false;
7   description
8   "If ptptimescale is true, the timescale of the
9   ClockTimeTransmitter entity is PTP, which is the elapsed time
10  since the PTP epoch measured using the second defined by
11  International Atomic Time (TAI). If ptptimescale is false, the
12  timescale of the ClockTimeTransmitter entity is ARB, which is the
13  elapsed time since an arbitrary epoch. This leaf applies to the
14  ClockTimeTransmitter entity (i.e., local only, unrelated to a
15  remote GM).";
16 reference
17 "14.2.14 of IEEE Std 802.1AS";
18 }
19 leaf timesource {
20   type identityref {
21     base ptptt:timesource;
22 }
23   config false;
24   description
25   "The source of time used by the Grandmaster Clock. This leaf
26   applies to the ClockTimeTransmitter entity (i.e., local only,
27   unrelated to a remote GM).";
```

```
1 reference
2 "14.2.15 of IEEE Std 802.1AS";
3 }
4 }

5 augment
6 "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:current-ds" {
7 description
8 "Augment IEEE Std 1588 currentDS.";
9 leaf last-gm-phase-change {
10 type scaled-ns;
11 config false;
12 description
13 "Phase change that occurred on the most recent change in either
14 the Grandmaster PTP Instance or gm-timebase-indicator leaf.";
15 reference
16 "14.3.4 of IEEE Std 802.1AS";
17 }
18 leaf last-gm-freq-change {
19 type float64;
20 config false;
21 description
22 "Frequency change that occurred on the most recent change in
23 either the Grandmaster PTP Instance or gm-timebase-indicator
24 leaf.";
25 reference
26 "14.3.5 of IEEE Std 802.1AS";
27 }
```

```
1 leaf gm-timebase-indicator {  
2   type uint16;  
3   config false;  
4   description  
5     "The timeBaseIndicator of the current Grandmaster PTP Instance.";  
6   reference  
7     "14.3.6 of IEEE Std 802.1AS";  
8 }  
  
9 leaf gm-change-count {  
10  type yang:counter32;  
11  config false;  
12  description  
13    "This statistics counter tracks the number of times the  
14    Grandmaster PTP Instance has changed in a gPTP domain.";  
15  reference  
16    "14.3.7 of IEEE Std 802.1AS";  
17 }  
  
18 leaf time-of-last-gm-change {  
19  type yang:timestamp;  
20  config false;  
21  description  
22    "System time when the most recent Grandmaster Clock change  
23    occurred in a gPTP domain. This leaf's type is YANG timestamp,  
24    which is based on system time. System time is an unsigned integer  
25    in units of 10 milliseconds, using an epoch defined by the  
26    implementation (typically time of boot-up).";  
27  reference
```

```
1      "14.3.8 of IEEE Std 802.1AS";  
2  }  
3  leaf time-of-last-phase-change {  
4    type yang:timestamp;  
5    config false;  
6    description  
7      "System time when the most recent change in Grandmaster Clock  
8      phase occurred. This leaf's type is YANG timestamp, which is  
9      based on system time. System time is an unsigned integer in units  
10     of 10 milliseconds, using an epoch defined by the implementation  
11     (typically time of boot-up).";  
12    reference  
13    "14.3.9 of IEEE Std 802.1AS";  
14  }  
15  leaf time-of-last-freq-change {  
16    type yang:timestamp;  
17    config false;  
18    description  
19      "System time when the most recent change in Grandmaster Clock  
20      frequency occurred. This leaf's type is YANG timestamp, which is  
21      based on system time. System time is an unsigned integer in units  
22      of 10 milliseconds, using an epoch defined by the implementation  
23      (typically time of boot-up).";  
24    reference  
25    "14.3.10 of IEEE Std 802.1AS";  
26  }  
27 }
```

```
1 augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:parent-ds" {  
2   description  
3     "Augment IEEE Std 1588 parentDS.";  
4   leaf cumulative-rate-ratio {  
5     type int32;  
6     config false;  
7     description  
8       "Estimate of the ratio of the frequency of the Grandmaster Clock  
9         to the frequency of the LocalClock entity of this PTP Instance.  
10        cumulative-rate-ratio is expressed as the fractional frequency  
11        offset multiplied by 2^41, i.e., the quantity (rateRatio -  
12        1.0)(2^41).";  
13     reference  
14       "14.4.3 of IEEE Std 802.1AS";  
15   }  
16 }  
17 augment  
18   "/ptp-tt:ptp"+  
19   "/ptp-tt:instances"+  
20   "/ptp-tt:instance"+  
21   "/ptp-tt:ports"+  
22   "/ptp-tt:port"+  
23   "/ptp-tt:port-ds" {  
24   description  
25     "Augment IEEE Std 1588 portDS.  
26  
27     14.8.4 of IEEE Std 802.1AS specifies ptPortEnabled
```

1 (ptp-port-enabled), which is provided in YANG as the semantically
2 equivalent node in ieee1588-ptp-tt named port-enable (in port-ds).
3
4 14.8.15 of IEEE Std 802.1AS specifies
5 mgtSettableLogAnnounceInterval (mgt-log-announce-interval), which
6 is provided in YANG as the semantically equivalent node in
7 ieee1588-ptp-tt named log-announce-interval (in port-ds). In the
8 context of IEEE Std 802.1AS, log-announce-interval cannot be used
9 unless use-mgt-log-announce-interval is true.
10
11 14.8.20 of IEEE Std 802.1AS specifies mgtSettableLogSyncInterval
12 (mgt-log-sync-interval), which is provided in YANG as the
13 semantically equivalent node in ieee1588-ptp-tt named
14 log-sync-interval (in port-ds). In the context of IEEE Std 802.1AS,
15 log-sync-interval cannot be used unless use-mgt-log-sync-interval
16 is true.";
17 leaf is-measuring-delay {
18 type boolean;
19 config false;
20 description
21 "Boolean that is true if the port is measuring PTP Link
22 propagation delay.";
23 reference
24 "14.8.6 of IEEE Std 802.1AS";
25 }
26 leaf as-capable {
27 type boolean;

```
1 config false;
2 description
3 "Boolean that is true if and only if it is determined that this
4 PTP Instance and the PTP Instance at the other end of the link
5 attached to this port can interoperate with each other via the
6 IEEE Std 802.1AS protocol.";
7 reference
8 "10.2.5.1 of IEEE Std 802.1AS
9 14.8.7 of IEEE Std 802.1AS";
10 }

11 leaf mean-link-delay-thresh {
12 type ptp-tt:time-interval;
13 description
14 "Propagation time threshold for mean-link-delay, above which a
15 port is not considered capable of participating in the IEEE Std
16 802.1AS protocol.";
17 reference
18 "14.8.9 of IEEE Std 802.1AS";
19 }

20 leaf neighbor-rate-ratio {
21 type int32;
22 config false;
23 description
24 "Estimate of the ratio of the frequency of the LocalClock entity
25 of the PTP Instance at the other end of the link attached to this
26 PTP Port, to the frequency of the LocalClock entity of this PTP
27 Instance. neighbor-rate-ratio is expressed as the fractional
```

```
1   frequency offset multiplied by 2^41, i.e., the quantity
2   (rateRatio - 1.0)(2^41).";
3   reference
4   "14.8.11 of IEEE Std 802.1AS";
5 }
6 leaf initial-log-announce-interval {
7   type int8;
8   description
9   "When use-mgt-log-announce-interval is false (i.e., change with
10  Signaling message), this is the the logarithm to base 2 of the
11  announce interval used when the port is initialized.";
12  reference
13  "14.8.12 of IEEE Std 802.1AS";
14 }
15 leaf current-log-announce-interval {
16   type int8;
17   config false;
18   description
19   "Logarithm to base 2 of the current announce interval.";
20  reference
21  "14.8.13 of IEEE Std 802.1AS";
22 }
23 leaf use-mgt-log-announce-interval {
24   type boolean;
25   description
26   "Boolean that determines the source of the announce interval. If
27   the value is true, the announce interval
```

```
1   (current-log-announce-interval) is set equal to the value of
2   mgt-log-announce-interval. If the value is false, the announce
3   interval is determined by the AnnounceIntervalSetting state
4   machine (i.e., changed with Signaling message).";
5   reference
6   "14.8.14 of IEEE Std 802.1AS";
7 }
8 leaf initial-log-sync-interval {
9   type int8;
10  description
11  "When use-mgt-log-sync-interval is false (i.e., change with
12  Signaling message), this is the the logarithm to base 2 of the
13  sync interval used when the port is initialized.";
14  reference
15  "14.8.17 of IEEE Std 802.1AS";
16 }
17 leaf current-log-sync-interval {
18   type int8;
19   config false;
20   description
21   "Logarithm to base 2 of the current sync interval.";
22   reference
23   "14.8.18 of IEEE Std 802.1AS";
24 }
25 leaf use-mgt-log-sync-interval {
26   type boolean;
27   description
```

```
1 "Boolean that determines the source of the sync interval. If the
2 value is true, the sync interval (current-log-sync-interval) is
3 set equal to the value of mgt-log-sync-interval. If the value is
4 false, the sync interval is determined by the SyncIntervalSetting
5 state machine (i.e., changed with Signaling message).";
6 reference
7 "14.8.19 of IEEE Std 802.1AS";
8 }
9 leaf sync-receipt-timeout {
10 type uint8;
11 description
12 "Number of sync intervals that a timeReceiver port waits without
13 receiving synchronization information, before assuming that the
14 timeTransmitter is no longer transmitting synchronization
15 information and that the BTCA needs to be run, if appropriate.";
16 reference
17 "14.8.21 of IEEE Std 802.1AS";
18 }
19 leaf sync-receipt-timeout-interval {
20 type uscaled-ns;
21 config false;
22 description
23 "Time interval after which sync receipt timeout occurs if
24 time-synchronization information has not been received during the
25 interval.";
26 reference
27 "14.8.22 of IEEE Std 802.1AS";
```

```
1   }
2   leaf initial-log-pdelay-req-interval {
3     type int8;
4     description
5       "When use-mgt-log-pdelay-req-interval is false (i.e., change with
6       Signaling message), this is the the logarithm to base 2 of the
7       Pdelay_Req transmit interval used when the port is initialized.";
8     reference
9       "14.8.23 of IEEE Std 802.1AS";
10    }
11   leaf current-log-pdelay-req-interval {
12     type int8;
13     config false;
14     description
15       "Logarithm to base 2 of the current Pdelay_Req transmit interval.";
16     reference
17       "14.8.24 of IEEE Std 802.1AS";
18    }
19   leaf use-mgt-log-pdelay-req-interval {
20     type boolean;
21     description
22       "Boolean that determines the source of the Pdelay_Req transmit
23       interval. If the value is true, the Pdelay_Req transmit interval
24       (current-log-pdelay-req-interval) is set equal to the value of
25       mgt-log-pdelay-req-interval. If the value is false, the
26       Pdelay_Req transmit interval is determined by the
27       LinkDelayIntervalSetting state machine (i.e., changed with
```

```
1     Signaling message).";  
2     reference  
3     "14.8.25 of IEEE Std 802.1AS";  
4 }  
5 leaf mgt-log-pdelay-req-interval {  
6     type int8;  
7     description  
8     "Logarithm to base 2 of the Pdelay_Req transmit interval, used if  
9     use-mgt-log-pdelay-req-interval is true. This value is not used  
10    if use-mgt-log-pdelay-req-interval is false.";  
11    reference  
12    "14.8.26 of IEEE Std 802.1AS";  
13 }  
14 leaf initial-log-gptp-cap-interval {  
15     type int8;  
16     description  
17     "When use-mgt-log-gptp-cap-interval is false (i.e., change with  
18     Signaling message), this is the the logarithm to base 2 of the  
19     gPTP capable message interval used when the port is initialized.";  
20     reference  
21     "14.8.27 of IEEE Std 802.1AS";  
22 }  
23 leaf current-log-gptp-cap-interval {  
24     type int8;  
25     config false;  
26     description  
27     "Logarithm to base 2 of the current gPTP capable message
```

```
1     interval.";  
2     reference  
3     "14.8.28 of IEEE Std 802.1AS";  
4 }  
5 leaf use-mgt-log-gptp-cap-interval {  
6     type boolean;  
7     description  
8     "Boolean that determines the source of the gPTP capable message  
9     interval. If the value is true, the gPTP capable message interval  
10    (current-log-gptp-cap-interval) is set equal to the value of  
11    mgt-gptp-cap-req-interval. If the value is false, the gPTP  
12    capable message interval is determined by the  
13    GptpCapableMessageIntervalSetting state machine (i.e., changed  
14    with Signaling message).";  
15     reference  
16     "14.8.29 of IEEE Std 802.1AS";  
17 }  
18 leaf mgt-log-gptp-cap-interval {  
19     type int8;  
20     description  
21     "Logarithm to base 2 of the gPTP capable message interval, used  
22     if use-mgt-log-gptp-cap-interval is true. This value is not used  
23     if use-mgt-log-pdelay-req-interval is false.";  
24     reference  
25     "14.8.30 of IEEE Std 802.1AS";  
26 }  
27 leaf initial-compute-neighbor-rate-ratio {
```

```
1 type boolean;
2 description
3 "When use-mgt-compute-neighbor-rate-ratio is false (i.e., change
4 with Signaling message), this is the initial value of
5 computeNeighborRateRatio.";
6 reference
7 "14.8.31 of IEEE Std 802.1AS";
8 }
9 leaf current-compute-neighbor-rate-ratio {
10 type boolean;
11 config false;
12 description
13 "Current value of computeNeighborRateRatio.";
14 reference
15 "14.8.32 of IEEE Std 802.1AS";
16 }
17 leaf use-mgt-compute-neighbor-rate-ratio {
18 type boolean;
19 description
20 "Boolean that determines the source of computeNeighborRateRatio..
21 If the value is true, computeNeighborRateRatio is set equal to
22 the value of mgt-compute-neighbor-rate-ratio. If the value is
23 false, computeNeighborRateRatio is determined by the
24 LinkDelayIntervalSetting state machine (i.e., changed with
25 Signaling message).";
26 reference
27 "14.8.33 of IEEE Std 802.1AS";
```

```
1   }
2   leaf mgt-compute-neighbor-rate-ratio {
3     type boolean;
4     description
5       "Value of computeNeighborRateRatio, used if
6       use-mgt-compute-neighbor-rate-ratio is true. This value is not
7       used if use-mgt-compute-neighbor-rate-ratio is false.";
8     reference
9       "14.8.34 of IEEE Std 802.1AS";
10  }
11  leaf initial-compute-mean-link-delay {
12    type boolean;
13    description
14      "When use-mgt-compute-mean-link-delay is false (i.e., change with
15      Signaling message), this is the initial value of
16      computeMeanLinkDelay.";
17    reference
18      "14.8.35 of IEEE Std 802.1AS";
19  }
20  leaf current-compute-mean-link-delay {
21    type boolean;
22    config false;
23    description
24      "Current value of computeMeanLinkDelay.";
25    reference
26      "14.8.36 of IEEE Std 802.1AS";
27 }
```

```
1 leaf use-mgt-compute-mean-link-delay {  
2   type boolean;  
3   description  
4     "Boolean that determines the source of computeMeanLinkDelay. If  
5     the value is true, computeMeanLinkDelay is set equal to the value  
6     of mgt-compute-mean-link-delay. If the value is false,  
7     computeMeanLinkDelay is determined by the  
8     LinkDelayIntervalSetting state machine (i.e., changed with  
9     Signaling message).";  
10  reference  
11    "14.8.37 of IEEE Std 802.1AS";  
12  }  
13 leaf mgt-compute-mean-link-delay {  
14   type boolean;  
15   description  
16     "Value of computeMeanLinkDelay, used if  
17     use-mgt-compute-mean-link-delay is true. This value is not used  
18     if use-mgt-compute-mean-link-delay is false.";  
19   reference  
20    "14.8.38 of IEEE Std 802.1AS";  
21  }  
22 leaf allowed-lost-responses {  
23   type uint8;  
24   description  
25     "Number of Pdelay_Req messages for which a valid response is not  
26     received, above which a port is considered to not be exchanging  
27     peer delay messages with its neighbor.";
```

```
1   reference
2   "14.8.39 of IEEE Std 802.1AS";
3 }
4 leaf allowed-faults {
5   type uint8;
6   description
7   "Number of faults above which asCapable is set to false.";
8   reference
9   "14.8.40 of IEEE Std 802.1AS";
10 }
11 leaf gptp-cap-receipt-timeout {
12   type uint8;
13   description
14   "Number of transmission intervals that a port waits without
15   receiving the gPTP capable TLV, before assuming that the neighbor
16   port is no longer invoking the gPTP protocol.";
17   reference
18   "14.8.41 of IEEE Std 802.1AS";
19 }
20 leaf nup {
21   type float64;
22   description
23   "For an OLT port of an IEEE Std 802.3 EPON link, this value is
24   the effective index of refraction for the EPON upstream
25   wavelength light of the optical path.";
26   reference
27   "14.8.43 of IEEE Std 802.1AS";
```

```
1   }
2   leaf ndown {
3     type float64;
4     description
5       "For an OLT port of an IEEE 802.3 EPON link, this value is the
6       effective index of refraction for the EPON downstream wavelength
7       light of the optical path.";
8     reference
9       "14.8.44 of IEEE Std 802.1AS";
10  }
11  leaf one-step-tx-oper {
12    type boolean;
13    config false;
14    description
15      "This value is true if the port is sending one-step Sync
16      messages, and false if the port is sending two-step Sync and
17      Follow-Up messages.";
18    reference
19      "14.8.45 of IEEE Std 802.1AS";
20  }
21  leaf one-step-receive {
22    type boolean;
23    config false;
24    description
25      "This value is true if the port is capable of receiving and
26      processing one-step Sync messages.";
27    reference
```

```
1      "14.8.46 of IEEE Std 802.1AS";  
2  }  
3  leaf one-step-transmit {  
4    type boolean;  
5    config false;  
6    description  
7    "This value is true if the port is capable of transmitting  
8    one-step Sync messages.";  
9    reference  
10   "14.8.47 of IEEE Std 802.1AS";  
11  }  
12 leaf initial-one-step-tx-oper {  
13  type boolean;  
14  description  
15  "When use-mgt-one-step-tx-oper is false (i.e., change with  
16  Signaling message), this is the initial value of  
17  current-one-step-tx-oper.";  
18  reference  
19  "14.8.48 of IEEE Std 802.1AS";  
20  }  
21 leaf current-one-step-tx-oper {  
22  type boolean;  
23  config false;  
24  description  
25  "This value is true if the port is configured to transmit  
26  one-step Sync messages, either via management  
27  (mgt-one-step-tx-oper) or Signaling. If both
```

```
1   current-one-step-tx-oper and one-step-transmit are true, the port
2   transmits one-step Sync messages (i.e., one-step-tx-oper true).";
3   reference
4   "14.8.49 of IEEE Std 802.1AS";
5 }
6 leaf use-mgt-one-step-tx-oper {
7   type boolean;
8   description
9   "Boolean that determines the source of current-one-step-tx-oper.
10  If the value is true, current-one-step-tx-oper is set equal to
11  the value of mgt-one-step-tx-oper. If the value is false,
12  current-one-step-tx-oper is determined by the
13  OneStepTxOperSetting state machine (i.e., changed with Signaling
14  message).";
15  reference
16  "14.8.50 of IEEE Std 802.1AS";
17 }
18 leaf mgt-one-step-tx-oper {
19   type boolean;
20   description
21   "If use-mgt-one-step-tx-oper is true, current-one-step-tx-oper is
22   set equal to this value. This value is not used if
23   use-mgt-one-step-tx-oper is false.";
24   reference
25   "14.8.51 of IEEE Std 802.1AS";
26 }
27 leaf sync-locked {
```

```
1 type boolean;
2 config false;
3 description
4 "This value is true if the port will transmit a Sync as soon as
5 possible after the timeReceiver port receives a Sync message.";
6 reference
7 "14.8.52 of IEEE Std 802.1AS";
8 }
9 leaf-list pdelay-truncated-timestamps {
10 type uinteger48;
11 config false;
12 description
13 "For full-duplex IEEE Std 802.3 media, and CSN media that use the
14 peer-to-peer delay mechanism to measure path delay, the values of
15 the four elements of this leaf-list correspond to the timestamps
16 t1, t2, t3, and t4, listed in that order. Each timestamp is
17 expressed in units of 2^-16 ns (i.e., the value of each array
18 element is equal to the remainder obtained upon dividing the
19 respective timestamp, expressed in units of 2^-16 ns, by 2^48).
20 At any given time, the timestamp values stored in the array are
21 for the same, and most recently completed, peer delay message
22 exchange. For each timestamp, only 48-bits are valid (the upper
23 16-bits are always zero).";
24 reference
25 "14.8.53 of IEEE Std 802.1AS";
26 }
27 }
```

```
1 augment
2 "/ptp-tt:ptp"+
3 "/ptp-tt:instances"+
4 "/ptp-tt:instance"+
5 "/ptp-tt:ports"+
6 "/ptp-tt:port" {
7   description
8   "Augment to add port-statistics-ds to IEEE Std 1588 PTP Port.";
9   container port-statistics-ds {
10     description
11     "Provides counters associated with the port of the PTP Instance.";
12     reference
13     "14.10 of IEEE Std 802.1AS";
14     leaf rx-sync-count {
15       type yang:counter32;
16       config false;
17       description
18       "Counter that increments every time synchronization information
19       is received.";
20     reference
21     "14.10.2 of IEEE Std 802.1AS";
22   }
23   leaf rx-one-step-sync-count {
24     type yang:counter32;
25     config false;
26     description
27     "Counter that increments every time a one-step Sync message is
```

```
1      received.";  
2      reference  
3      "14.10.3 of IEEE Std 802.1AS";  
4  }  
5  leaf rx-follow-up-count {  
6      type yang:counter32;  
7      config false;  
8      description  
9      "Counter that increments every time a Follow_Up message is  
10     received.";  
11    reference  
12    "14.10.4 of IEEE Std 802.1AS";  
13  }  
14  leaf rx-pdelay-req-count {  
15      type yang:counter32;  
16      config false;  
17      description  
18      "Counter that increments every time a Pdelay_Req message is  
19     received.";  
20    reference  
21    "14.10.5 of IEEE Std 802.1AS";  
22  }  
23  leaf rx-pdelay-resp-count {  
24      type yang:counter32;  
25      config false;  
26      description  
27      "Counter that increments every time a Pdelay_Resp message is
```

```
1     received.";  
2     reference  
3     "14.10.6 of IEEE Std 802.1AS";  
4 }  
5 leaf rx-pdelay-resp-follow-up-count {  
6     type yang:counter32;  
7     config false;  
8     description  
9     "Counter that increments every time a Pdelay_Resp_Follow_Up  
10    message is received.";  
11    reference  
12    "14.10.7 of IEEE Std 802.1AS";  
13 }  
14 leaf rx-announce-count {  
15     type yang:counter32;  
16     config false;  
17     description  
18     "Counter that increments every time an Announce message is  
19     received.";  
20     reference  
21     "14.10.8 of IEEE Std 802.1AS";  
22 }  
23 leaf rx-packet-discard-count {  
24     type yang:counter32;  
25     config false;  
26     description  
27     "Counter that increments every time a PTP message of the
```

```
1      respective PTP Instance is discarded.";  
2      reference  
3      "14.10.9 of IEEE Std 802.1AS";  
4      }  
5      leaf sync-receipt-timeout-count {  
6          type yang:counter32;  
7          config false;  
8          description  
9              "Counter that increments every time a sync receipt timeout  
10             occurs.>";  
11         reference  
12         "14.10.10 of IEEE Std 802.1AS";  
13         }  
14         leaf announce-receipt-timeout-count {  
15             type yang:counter32;  
16             config false;  
17             description  
18             "Counter that increments every time an announce receipt timeout  
19             occurs.>";  
20         reference  
21         "14.10.11 of IEEE Std 802.1AS";  
22         }  
23         leaf pdelay-allowed-lost-exceeded-count {  
24             type yang:counter32;  
25             config false;  
26             description  
27             "Counter that increments every time the value of the variable
```

```
1      lostResponses exceeds the value of the variable
2      allowedLostResponses, in the RESET state of the MDPdelayReq
3      state machine.";
4      reference
5      "14.10.12 of IEEE Std 802.1AS";
6  }
7  leaf tx-sync-count {
8      type yang:counter32;
9      config false;
10     description
11     "Counter that increments every time synchronization information
12     is transmitted.";
13     reference
14     "14.10.13 of IEEE Std 802.1AS";
15  }
16  leaf tx-one-step-sync-count {
17      type yang:counter32;
18      config false;
19      description
20      "Counter that increments every time a one-step Sync message is
21      transmitted.";
22      reference
23      "14.10.14 of IEEE Std 802.1AS";
24  }
25  leaf tx-follow-up-count {
26      type yang:counter32;
27      config false;
```

```
1    description
2      "Counter that increments every time a Follow_Up message is
3      transmitted.";
4    reference
5      "14.10.15 of IEEE Std 802.1AS";
6  }
7  leaf tx-pdelay-req-count {
8    type yang:counter32;
9    config false;
10   description
11     "Counter that increments every time a Pdelay_Req message is
12     transmitted.";
13   reference
14     "14.10.16 of IEEE Std 802.1AS";
15 }
16 leaf tx-pdelay-resp-count {
17   type yang:counter32;
18   config false;
19   description
20     "Counter that increments every time a Pdelay_Resp message is
21     transmitted.";
22   reference
23     "14.10.17 of IEEE Std 802.1AS";
24 }
25 leaf tx-pdelay-resp-follow-up-count {
26   type yang:counter32;
27   config false;
```

```
1     description
2     "Counter that increments every time a Pdelay_Resp_Follow_Up
3     message is transmitted.";
4     reference
5     "14.10.18 of IEEE Std 802.1AS";
6   }
7   leaf tx-announce-count {
8     type yang:counter32;
9     config false;
10    description
11    "Counter that increments every time an Announce message is
12    transmitted.";
13    reference
14    "14.10.19 of IEEE Std 802.1AS";
15  }
16 }
17 }
18 augment
19 "/ptp-tt:ptp"+
20 "/ptp-tt:instances"+
21 "/ptp-tt:instance"+
22 "/ptp-tt:ports"+
23 "/ptp-tt:port" {
24   description
25   "Augment to add asymmetry-measurement-mode-ds to IEEE Std 1588 PTP
26   Port.";
27   container asymmetry-measurement-mode-ds {
```

```
1   description
2   "Represents the capability to enable/disable the Asymmetry
3   Compensation Measurement Procedure on a PTP Port. This data set
4   is used instead of the CMLDS asymmetry-measurement-mode-ds when
5   only a single PTP Instance is present (i.e., CMLDS is not used).";
6   reference
7   "14.13 of IEEE Std 802.1AS
8   Annex G of IEEE Std 802.1AS";
9   leaf enabled {
10   type boolean;
11   description
12   "For full-duplex IEEE Std 802.3 media, the value is true if an
13   asymmetry measurement is being performed for the link attached
14   to this PTP Port, and false otherwise. For all other media, the
15   value shall be false.";
16   reference
17   "14.13.2 of IEEE Std 802.1AS";
18   }
19   }
20   }
21   augment
22   "/ptp-tt:ptp"+
23   "/ptp-tt:common-services"+
24   "/ptp-tt:cmlds"+
25   "/ptp-tt:ports"+
26   "/ptp-tt:port"+
27   "/ptp-tt:link-port-ds" {
```

```
1 description
2 "Augment IEEE Std 1588 cmldsLinkPortDS.
3
4 14.16.9 of IEEE Std 802.1AS specifies neighborRateRatio
5 (neighbor-rate-ratio), which is provided in YANG as the
6 semantically equivalent node in ieee1588-ptp-tt named
7 scaled-neighbor-rate-ratio (in link-port-ds).";
8 leaf cmlds-link-port-enabled {
9   type boolean;
10  config false;
11  description
12    "Boolean that is true if both delay-mechanism is common-p2p and
13    the value of ptp-port-enabled is true, for at least one PTP Port
14    that uses the CMLDS; otherwise, the value is false.";
15  reference
16    "11.2.18.1 of IEEE Std 802.1AS
17    14.16.3 of IEEE Std 802.1AS";
18 }
19 leaf is-measuring-delay {
20   type boolean;
21   config false;
22   description
23     "This leaf is analogous to is-measuring-delay for a PTP Port, but
24     applicable to this Link Port.";
25   reference
26     "14.16.4 of IEEE Std 802.1AS";
27 }
```

```
1 leaf as-capable-across-domains {  
2   type boolean;  
3   config false;  
4   description  
5     "This leaf is true when all PTP Instances (domains) for this Link  
6     Port detect proper exchange of Pdelay messages.";  
7   reference  
8     "11.2.2 of IEEE Std 802.1AS  
9     14.16.5 of IEEE Std 802.1AS";  
10 }  
  
11 leaf mean-link-delay-thresh {  
12   type ptptt:time-interval;  
13   description  
14     "Propagation time threshold for mean-link-delay, above which a  
15     Link Port is not considered capable of participating in the IEEE  
16     Std 802.1AS protocol.";  
17   reference  
18     "14.16.7 of IEEE Std 802.1AS";  
19 }  
  
20 leaf initial-log-pdelay-req-interval {  
21   type int8;  
22   description  
23     "This leaf is analogous to initial-log-pdelay-req-interval for a  
24     PTP Port, but applicable to this Link Port.";  
25   reference  
26     "14.16.10 of IEEE Std 802.1AS";  
27 }
```

```
1 leaf current-log-pdelay-req-interval {  
2     type int8;  
3     config false;  
4     description  
5         "This leaf is analogous to current-log-pdelay-req-interval for a  
6         PTP Port, but applicable to this Link Port.";  
7     reference  
8         "14.16.11 of IEEE Std 802.1AS";  
9 }  
10 leaf use-mgt-log-pdelay-req-interval {  
11     type boolean;  
12     description  
13         "This leaf is analogous to use-mgt-log-pdelay-req-interval for a  
14         PTP Port, but applicable to this Link Port.";  
15     reference  
16         "14.16.12 of IEEE Std 802.1AS";  
17 }  
18 leaf mgt-log-pdelay-req-interval {  
19     type int8;  
20     description  
21         "This leaf is analogous to mgt-log-pdelay-req-interval for a PTP  
22         Port, but applicable to this Link Port.";  
23     reference  
24         "14.16.13 of IEEE Std 802.1AS";  
25 }  
26 leaf initial-compute-neighbor-rate-ratio {  
27     type boolean;
```

```
1   description
2   "This leaf is analogous to initial-compute-neighbor-rate-ratio
3   for a PTP Port, but applicable to this Link Port.";
4   reference
5   "14.16.14 of IEEE Std 802.1AS";
6   }
7   leaf current-compute-neighbor-rate-ratio {
8     type boolean;
9     config false;
10  description
11  "This leaf is analogous to current-compute-neighbor-rate-ratio
12  for a PTP Port, but applicable to this Link Port.";
13  reference
14  "14.16.15 of IEEE Std 802.1AS";
15  }
16  leaf use-mgt-compute-neighbor-rate-ratio {
17  type boolean;
18  description
19  "This leaf is analogous to use-mgt-compute-neighbor-rate-ratio
20  for a PTP Port, but applicable to this Link Port.";
21  reference
22  "14.16.16 of IEEE Std 802.1AS";
23  }
24  leaf mgt-compute-neighbor-rate-ratio {
25  type boolean;
26  description
27  "This leaf is analogous to mgt-compute-neighbor-rate-ratio for a
```

```
1      PTP Port, but applicable to this Link Port.";  
2      reference  
3      "14.16.17 of IEEE Std 802.1AS";  
4  }  
5  leaf initial-compute-mean-link-delay {  
6      type boolean;  
7      description  
8      "This leaf is analogous to initial-compute-mean-link-delay for a  
9      PTP Port, but applicable to this Link Port.";  
10     reference  
11    "14.16.18 of IEEE Std 802.1AS";  
12  }  
13 leaf current-compute-mean-link-delay {  
14     type boolean;  
15     config false;  
16     description  
17     "This leaf is analogous to current-compute-mean-link-delay for a  
18     PTP Port, but applicable to this Link Port.";  
19     reference  
20    "14.16.19 of IEEE Std 802.1AS";  
21  }  
22 leaf use-mgt-compute-mean-link-delay {  
23     type boolean;  
24     description  
25     "This leaf is analogous to use-mgt-compute-mean-link-delay for a  
26     PTP Port, but applicable to this Link Port.";  
27     reference
```

```
1      "14.16.20 of IEEE Std 802.1AS";  
2  }  
3  leaf mgt-compute-mean-link-delay {  
4    type boolean;  
5    description  
6      "This leaf is analogous to mgt-compute-mean-link-delay for a PTP  
7      Port, but applicable to this Link Port.>";  
8    reference  
9      "14.16.21 of IEEE Std 802.1AS";  
10 }  
11 leaf allowed-lost-responses {  
12   type uint8;  
13   description  
14   "This leaf is analogous to allowed-lost-responses for a PTP Port,  
15   but applicable to this Link Port.>";  
16   reference  
17   "14.16.22 of IEEE Std 802.1AS";  
18 }  
19 leaf allowed-faults {  
20   type uint8;  
21   description  
22   "This leaf is analogous to allowed-faults for a PTP Port, but  
23   applicable to this Link Port.>";  
24   reference  
25   "14.16.23 of IEEE Std 802.1AS";  
26 }  
27 leaf-list pdelay-truncated-timestamps {
```

```
1 type uinteger48;
2 config false;
3 description
4 "This leaf is analogous to pdelay-truncated-timestamps for a PTP
5 Port, but applicable to this Link Port.";
6 reference
7 "14.16.25 of IEEE Std 802.1AS";
8 }
9 }

10 augment
11 "/ptp-tt:ptp"+
12 "/ptp-tt:common-services"+
13 "/ptp-tt:cmlds"+
14 "/ptp-tt:ports"+
15 "/ptp-tt:port" {
16 description
17 "Augment to add port-statistics-ds to IEEE Std 1588 Link Port.";
18 container port-statistics-ds {
19 description
20 "This container is analogous to port-statistics-ds for a PTP
21 Port, but applicable to this Link Port.";
22 reference
23 "14.17 of IEEE Std 802.1AS";
24 leaf rx-pdelay-req-count {
25 type yang:counter32;
26 config false;
27 description
```

```
1      "This leaf is analogous to rx-pdelay-req-count for a PTP Port,  
2      but applicable to this Link Port.";  
3      reference  
4      "14.17.2 of IEEE Std 802.1AS";  
5      }  
6  leaf rx-pdelay-resp-count {  
7      type yang:counter32;  
8      config false;  
9      description  
10     "This leaf is analogous to rx-pdelay-resp-count for a PTP Port,  
11     but applicable to this Link Port.";  
12     reference  
13     "14.17.3 of IEEE Std 802.1AS";  
14     }  
15  leaf rx-pdelay-resp-follow-up-count {  
16      type yang:counter32;  
17      config false;  
18      description  
19      "This leaf is analogous to rx-pdelay-resp-follow-up-count for a  
20      PTP Port, but applicable to this Link Port.";  
21      reference  
22      "14.17.4 of IEEE Std 802.1AS";  
23      }  
24  leaf rx-packet-discard-count {  
25      type yang:counter32;  
26      config false;  
27      description
```

```
1      "This leaf is analogous to rx-packet-discard-count for a PTP
2      Port, but applicable to this Link Port.";
3      reference
4      "14.17.5 of IEEE Std 802.1AS";
5      }
6      leaf pdelay-allowed-lost-exceeded-count {
7          type yang:counter32;
8          config false;
9          description
10         "This leaf is analogous to pdelay-allowed-lost-exceeded-count
11        for a PTP Port, but applicable to this Link Port.";
12        reference
13        "14.17.6 of IEEE Std 802.1AS";
14        }
15        leaf tx-pdelay-req-count {
16            type yang:counter32;
17            config false;
18            description
19            "This leaf is analogous to tx-pdelay-req-count for a PTP Port,
20            but applicable to this Link Port.";
21            reference
22            "14.17.7 of IEEE Std 802.1AS";
23            }
24            leaf tx-pdelay-resp-count {
25                type yang:counter32;
26                config false;
27                description
```

```
1      "This leaf is analogous to tx-pdelay-resp-count for a PTP Port,
2      but applicable to this Link Port.";
3      reference
4      "14.17.8 of IEEE Std 802.1AS";
5      }
6      leaf tx-pdelay-resp-follow-up-count {
7          type yang:counter32;
8          config false;
9          description
10         "This leaf is analogous to tx-pdelay-resp-follow-up-count for a
11        PTP Port, but applicable to this Link Port.";
12        reference
13        "14.17.9 of IEEE Std 802.1AS";
14        }
15    }
16  }
17 augment
18   "/ptp-tt:ptp"+
19   "/ptp-tt:common-services"+
20   "/ptp-tt:cmlds"+
21   "/ptp-tt:ports"+
22   "/ptp-tt:port" {
23     description
24     "Augment to add asymmetry-measurement-mode-ds to IEEE Std 1588 Link
25     Port.";
26     container asymmetry-measurement-mode-ds {
27       description
```

```
1 "This container is analogous to asymmetry-measurement-mode-ds for
2 a PTP Port, but applicable to this Link Port.";
3 reference
4 "14.18 of IEEE Std 802.1AS";
5 leaf enabled {
6 type boolean;
7 description
8 "This leaf is analogous to
9 asymmetry-measurement-mode-ds.enabled for a PTP Port, but
10 applicable to this Link Port.";
11 reference
12 "14.18.2 of IEEE Std 802.1AS";
13 }
14 }
15 }
16 }
17
```

18 17.6.2 Module ieee802-dot1as-hs.yang

```
19 module ieee802-dot1as-hs {
20   yang-version 1.1;
21   namespace "urn:ieee:std:802.1AS:yang:ieee802-dot1as-hs";
22   prefix dot1as-hs;
23
24   import ietf-yang-types {
25     prefix yang;
26   }
27   import ieee1588-ptp-tt {
28     prefix ptp-tt;
29   }
30   import ieee802-dot1as-gptp {
31     prefix dot1as-gptp;
32   }
33
34   organization
35     "IEEE 802.1 Working Group";
36   contact
37     "WG-URL: http://ieee802.org/1/
```

```

1   WG-EMail: stds-802-1-1@ieee.org
2
3   Contact: IEEE 802.1 Working Group Chair
4       Postal: C/O IEEE 802.1 Working Group
5       IEEE Standards Association
6       445 Hoes Lane
7       Piscataway, NJ 08854
8       USA
9
10  E-mail: stds-802-1-chairs@ieee.org";
11  description
12      "Management objects that control hot standby systems as
13      specified in IEEE Std 802.1ASdm-2024.
14
15      References in this YANG module to IEEE Std 802.1AS are to
16      IEEE Std 802.1AS-2020 as modified by
17      IEEE Std 802.1AS-2020/Cor-1-2021, and amended by
18      IEEE Std 802.1ASdr-2024, IEEE Std 802.1ASdn-2024, and
19      IEEE Std 802.1ASdm-2024.
20
21  Copyright (C) IEEE (2024).
22  This version of this YANG module is part of IEEE Std 802.1AS;
23  see the standard itself for full legal notices.";
24
25 revision 2024-09-20 {
26     description
27         "Published as part of IEEE Std 802.1ASdm-2024.
28             Initial version.";
29     reference
30         "IEEE Std 802.1AS - Timing and Synchronization for Time-Sensitive
31         Applications: IEEE Std 802.1AS-2020, IEEE Std 802.1AS-2020/Cor
32         1-2021, IEEE Std 802.1ASdr-2024, IEEE Std 802.1ASdn-2024,
33         IEEE Std 802.1ASdm-2024.";
34 }
35
36 feature hot-standby {
37     description
38         "This feature indicates that the device supports the
39             hot-standby functionality.";
40 }
41
42 typedef hot-standby-system-state-type {
43     type enumeration {
44         enum init {
45             value 0;
46             description
47                 "Initialization after the HotStandbySystem powers on and
48                 is enabled. In this state, the system is waiting for
49                 both PTP Instances to synchronize.";
50         }
51         enum redundant {
52             value 1;
53             description
54                 "Both PTP Instances are synchronized according to the
55                 requirements of the respective application or profile
56                 standard. Time synchronization is redundant.";
57         }
58         enum not-redundant {
59             value 2;

```

```

1      description
2          "One PTP Instance is synchronized, and the other
3              PTP Instance is faulted (not synchronized). Time
4                  synchronization continues to meet the requirements of
5                      the respective application or profile standard. Time
6                          synchronization is not redundant.";
7      }
8      enum out-of-sync {
9          value 3;
10         description
11             "The HotStandbySystem is adjusting phase/frequency of
12                 its local time using the data stored while the system
13                     was in the REDUNDANT or NOT_REDUNDANT state, but the
14                         local time will eventually drift relative to other
15                             time-aware systems. During OUT_OF_SYNC state, time
16                                 synchronization might not meet the requirements of the
17                                     respective application or profile standard.";
18     }
19   }
20   description
21     "Possible states of the hot-standby system.";
22 }
23
24 typedef nrr-comp-method-type {
25   type enumeration {
26     enum sync {
27         value 0;
28         description
29             "If the value is Sync and driftTrackingTlvSupport is
30                 TRUE, neighborRateRatio is populated with the value of
31                     nrrSync whenever a new value of nrrSync is computed.";
32     }
33     enum pdelay {
34         value 1;
35         description
36             "If the value is Pdelay or if driftTrackingTlvSupport
37                 is FALSE, neighborRateRatio is populated with the value
38                     of nrrPdelay whenever a new value of nrrPdelay is
39                         computed.";
40   }
41 }
42   description
43     "typedef for nrrCompMethod.";
44 }
45
46 augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
47     + "/ptp-tt:parent-ds" {
48   description
49     "Augment IEEE Std 1588 parentDS.";
50   leaf gm-present {
51     type boolean;
52     config false;
53     description
54       "The value of gmPresent is set equal to the value of the
55           global variable gmPresent. This parameter indicates to the
56             ClockTarget whether a Grandmaster PTP Instance is
57               present.";
58   reference
59     "14.4.7a of IEEE Std 802.1AS";

```

```

1      }
2  }
3
4 augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance" {
5   description
6     "Augment IEEE Std 1588 instanceList.";
7   container ptp-instance-sync-ds {
8     description
9       "The ptpInstanceSyncDS describes the synchronization status
10      of the PTP Instance.";
11     reference
12       "14.7a of IEEE Std 802.1AS";
13     leaf is-synced {
14       type boolean;
15       config false;
16       description
17         "The value of the global variable isSynced.";
18       reference
19         "14.7a.2 of IEEE Std 802.1AS";
20     }
21     leaf offset-from-time-transmitter-max {
22       type ptp-tt:time-interval;
23       description
24         "The value is the threshold for
25          offsetFromTimeTransmitter, below which the PTP Instance
26          is considered to be synchronized.";
27       reference
28         "14.7a.3 of IEEE Std 802.1AS";
29     }
30     leaf rx-sync-count-time-receiver-p-thresh {
31       type uint32;
32       description
33         "The value of rxSyncCountTimeReceiverPThresh is the
34           threshold for rxSyncCountTimeReceiverP, above which
35           the PTP Instance is considered to be synchronized.";
36       reference
37         "14.7a.4 of IEEE Std 802.1AS";
38     }
39     leaf offset-max-exceeded-count-thresh {
40       type uint32;
41       description
42         "The value of offsetMaxExceededCountThresh is the
43           threshold for the number of consecutive exceedances of
44             offsetFromTimeTransmitterMax by
45               offsetFromTimeTransmitter, at which isSynced is no
46                 longer TRUE.";
47       reference
48         "14.7a.5 of IEEE Std 802.1AS";
49     }
50     leaf offset-max-met-count-thresh {
51       type uint32;
52       description
53         "The value of offsetMaxMetCountThresh is the threshold
54           for the number of consecutive occurrences of
55             offsetFromTimeTransmitter being within
56               offsetFromTimeTransmitterMax, at which isSynced is
57                 changed to TRUE if it currently is FALSE.";
58       reference
59         "14.7a.6 of IEEE Std 802.1AS";

```

```

1      }
2  }
3  container drift-tracking-ds {
4    description
5      "The driftTrackingDS contains a managed object that is used
6      to enable or disable the optional Drift_Tracking TLV.";
7    reference
8      "14.7b of IEEE Std 802.1AS";
9    leaf drift-tracking-tlv-support {
10      type boolean;
11      description
12        "The value of driftTrackingTlvSupport indicates whether
13        the Drift_Tracking TLV is enabled or disabled.";
14      reference
15        "14.7b.2 of IEEE Std 802.1AS";
16    }
17  }
18 }
19
20 augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
21   + "/ptp-tt:ports/ptp-tt:port/ptp-tt:port-ds" {
22   description
23     "Augment IEEE Std 1588 commonServices.";
24   leaf gptp-capable-state-machines-enabled {
25     type boolean;
26     description
27       "A Boolean that is used to enable or disable the
28       GptpCapableTransmit, GptpCapableReceive, and
29       GptpCapableIntervalSetting state machines.";
30     reference
31       "14.8.54a of IEEE Std 802.1AS";
32   }
33   leaf nrr-pdelay {
34     type int32;
35     config false;
36     description
37       "The value is an estimate of the ratio of the frequency of
38       the LocalClock entity of the time-aware system at the
39       other end of the link attached to this Link Port, to the
40       frequency of the LocalClock entity of this time-aware
41       system. nrrPdelay is expressed as the fractional frequency
42       offset stored in the global variable nrrPdelay multiplied
43       by 2^41, i.e., the quantity (nrrPdelay - 1.0)(2^41).";
44     reference
45       "14.8.54b of IEEE Std 802.1AS";
46   }
47   leaf nrr-sync {
48     type int32;
49     config false;
50     description
51       "The value is an estimate of the ratio of the frequency of
52       the LocalClock entity of the time-aware system at the
53       other end of the link attached to this Link Port, to the
54       frequency of the LocalClock entity of this time-aware
55       system. nrrSync is expressed as the fractional frequency
56       offset stored in the global variable nrrSync multiplied by
57       2^41, i.e., the quantity (nrrSync - 1.0)(2^41).";
58     reference
59       "14.8.54c of IEEE Std 802.1AS";

```

```

1      }
2  leaf nrr-comp-method {
3    type nrr-comp-method-type;
4    description
5      "An Enumeration that takes on the values sync and pdelay to
6       indicate the source of the value of neighborRateRatio.";
7    reference
8      "14.8.54d of IEEE Std 802.1AS";
9  }
10 leaf as-capable-across-domains {
11   type boolean;
12   description
13     "This leaf is true when this PTP port detects proper
14      exchange of Pdelay messages.";
15   reference
16     "14.8.54e of IEEE Std 802.1AS";
17 }
18 }
19
20 augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
21   + "/ptp-tt:ports/ptp-tt:port"
22   + "/dotlas-gptp:port-statistics-ds" {
23   description
24     "Augment IEEE Std 802.1AS PortStatisticsDS.";
25   leaf rx-sync-count-time-receiver-p {
26     type yang:counter32;
27     config false;
28     description
29       "This counter increments whenever time synchronization
30          information is received on a PTP Port when its port
31          state is TimeReceiverPort.";
32     reference
33       "14.10.19a of IEEE Std 802.1AS";
34   }
35 }
36
37 augment "/ptp-tt:ptp/ptp-tt:common-services" {
38   description
39     "Augment IEEE Std 1588 commonServices.
40
41     IEEE Std 802.1ASdm-2024 specifies nrrPdelay
42     (nrr-pdelay), which is provided in YANG as the
43     semantically equivalent node in ieee1588-ptp-tt named
44     scaled-neighbor-rate-ratio (in link-port-ds).";
45   container hot-standby-service {
46     if-feature "hot-standby";
47     description
48       "The hotStandbyService structure contains the
49          hotStandbySystemList, which is a list of instances of the
50          Hot Standby Service.";
51     reference
52       "14.19 of IEEE Std 802.1AS";
53     list hot-standby-system {
54       key "hot-standby-system-index";
55       description
56         "List of instances of the Hot Standby Service";
57       leaf hot-standby-system-index {
58         type uint8;
59         description

```

```

1      "Index for the hot-standby system.";
2  }
3  container hot-standby-system-ds {
4    description
5      "The hotStandbySystemDS describes the attributes of the
6      respective instance of the Hot Standby Service.";
7    reference
8      "14.19 of IEEE Std 802.1AS";
9    leaf primary-ptp-instance-index {
10      type uint32;
11      description
12        "The value of primaryPtpInstanceIndex is the index of
13        the primary PTP Instance associated with this
14        hotStandbySystem instance.";
15      reference
16        "14.19.2 of IEEE Std 802.1AS";
17    }
18    leaf secondary-ptp-instance-index {
19      type uint32;
20      description
21        "The value of secondaryPtpInstanceIndex is the index
22        of the secondaryPTP Instance associated with this
23        hotStandbySystem instance.";
24      reference
25        "14.19.3 of IEEE Std 802.1AS";
26    }
27    leaf hot-standby-system-enable {
28      type boolean;
29      description
30        "The value is the hotStandbySystemEnable attribute of
31        the HotStandbySystem entity.";
32      reference
33        "14.19.4 of IEEE Std 802.1AS";
34    }
35    leaf hot-standby-system-state {
36      type hot-standby-system-state-type;
37      config false;
38      description
39        "The value of hotStandbySystemState is the state of
40        the hotStandbySystem, i.e., the value of the global
41        variable hotStandbySystemState.";
42      reference
43        "14.19.5 of IEEE Std 802.1AS";
44    }
45    leaf hot-standby-system-split-functionality {
46      type boolean;
47      description
48        "If the value is TRUE, the optional split
49        functionality is used. If the value is FALSE, the
50        optional split functionality is not used.";
51      reference
52        "14.19.6 of IEEE Std 802.1AS";
53    }
54    leaf primary-secondary-offset {
55      type dotlas-gptp:scaled-ns;
56      config false;
57      description
58        "The absolute value of the difference between the
59        clockTimeTransmitterTimes of the primary and

```

```

1      secondary PTP Instances." ;
2      reference
3          "14.19.7 of IEEE Std 802.1AS";
4  }
5  leaf primary-secondary-offset-thresh {
6      type dotlas-gptp:scaled-ns;
7      description
8          "The threshold for
9              hotStandbySystemDS.primarySecondaryOffset, above
10             which the hotStandbySystemState transitions from
11             REDUNDANT to NOT_REDUNDANT, or does not transition
12             from NOT_REDUNDANT or OUT_OF_SYNC to REDUNDANT even
13             if other conditions for these transitions are
14             satisfied.";
15      reference
16          "14.19.8 of IEEE Std 802.1AS";
17  }
18  leaf hot-standby-system-log-sync-time-thresh {
19      type int8;
20      description
21          "The value of hotStandbySystemLogSyncTimeThresh is
22              the logarithm to base 2 of the time interval, in
23              seconds, after which the hotStandbySystem
24              transitions from the OUT_OF_SYNC state to either the
25              NOT_REDUNDANT or REDUNDANT state, or from the
26              NOT_REDUNDANT to the REDUNDANT state, if all other
27              conditions for the respective transition are met.
28              The value -128 means that the transition time is
29              zero, i.e., the transition occurs immediately.";
30      reference
31          "14.19.9 of IEEE Std 802.1AS";
32  }
33  }
34  container hot-standby-system-description-ds {
35      description
36          "The hotStandbySystemDescriptionDS contains descriptive
37              information for the respective instance of the Hot
38              Standby Service.";
39      reference
40          "14.20 of IEEE Std 802.1AS";
41      leaf user-description {
42          type string {
43              length "0..128";
44          }
45          description
46              "Configurable description of the hot standby system.";
47          reference
48              "14.20.3 of IEEE Std 802.1AS";
49      }
50  }
51  }
52  }
53  }
54 }
```

55

1 18. Hot standby

2 18.1 General

3 Hot standby includes:

- 4 — A function that transforms the synchronized times of two generalized Precision Time Protocol (gPTP) domains into one synchronized time for use by applications;
- 5 — A function that directs the synchronized time of one gPTP domain into a different gPTP domain; and
- 6 — Mechanisms that determine whether a gPTP domain has sufficient quality to be used for hot standby.

9 For time synchronization using hot standby, two distinct domains are statically configured in the network and the best timeTransmitter clock algorithm (BTCA) is disabled for these domains. When hot standby is used for redundancy, a time-aware system operates two PTP Instances simultaneously, each in its own domain. One of the domains is considered the primary domain, and the other domain is considered the secondary domain. A time-aware system that has a primary domain and a secondary domain available uses the primary domain via the associated PTP instance. If the primary domain becomes unavailable (e.g., due to temporary or permanent failure of a physical link) and the secondary domain is still available to the time-aware system, the time-aware system begins using the secondary domain.

17 Examples of hot standby are shown in 7.2.4 of this standard.

18 18.2 Overview

19 Figure 18-1 provides a model of hot standby for time synchronization.

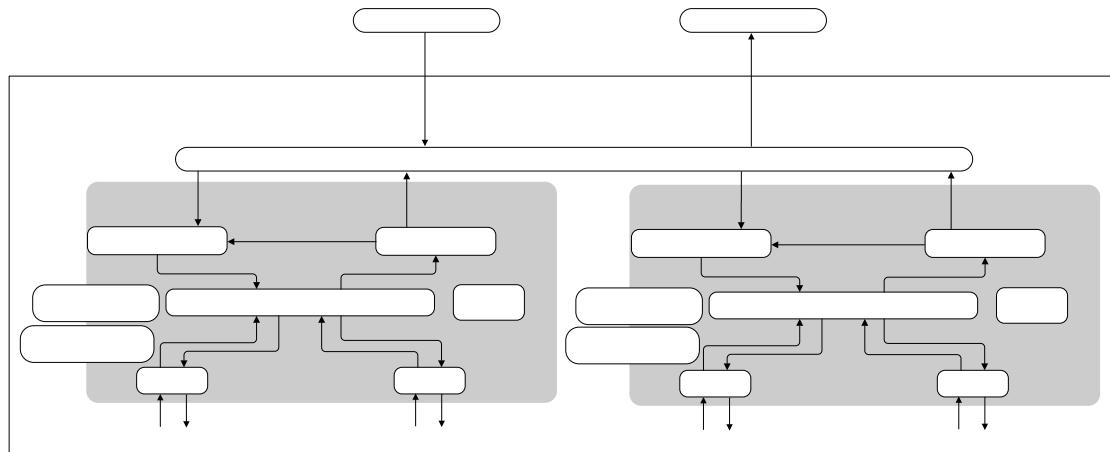


Figure 18-1—Model for hot standby entity in a time-aware system, and its interfaces to higher-layer applications

20 Each PTP Instance has a corresponding PtpInstanceSyncStatus state machine (see 18.4). The PtpInstanceSyncStatus state machine monitors its PTP Instance to determine whether it is synchronized according to the requirements of the respective application or profile standard.

1 There is one HotStandbySystem entity, which interacts with the primary and secondary PTP Instances via
2 the HotStandbySystem state machine (see 18.6) in order to provide a single value of time to the application
3 (ClockTarget). This single value of time is based on the redundant values of time provided by either the
4 primary PTP Instance, or the secondary PTP Instance, or both. There is one HotStandbySystem for both PTP
5 Instances that use hot standby. The primary and secondary PTP Instances either both use the PTP timescale
6 or both use the ARB timescale.

7 The ClockTarget entity represents the application that uses the synchronized time. The ClockTarget and its
8 application interfaces are analogous to the ClockTarget specified in Clause 9. In the case of PTP Relay
9 Instances, the ClockTarget is present if it is needed by the application.

10 The ClockSource entity is used to transfer the source of time when at least one of the PTP Instances is
11 grandmaster-capable. The ClockSource and its application interfaces are analogous to the ClockSource
12 specified in Clause 9. The ClockSource might be present if the PTP Instance is capable of becoming a
13 Grandmaster PTP Instance.

14 There is one LocalClock entity for each PTP Instance. Therefore, Figure 18-1 shows two LocalClock
15 entities.

16 NOTE—The LocalClock entities can be traceable to the same oscillator or to different oscillators.

17 **18.3 PTP Instance configuration**

18 PTP Instances that support hot standby are configured as follows:

- 19 a) The two PTP Instances are enabled using domainNumbers as specified by the respective application
20 or profile standard;
- 21 b) For both PTP Instances, externalPortConfigurationEnabled is set to TRUE;
- 22 c) If a PTP Instance (primary or secondary) is grandmaster, externalPortConfigurationPortDS.desiredState is configured to TimeTransmitterPort or PassivePort
23 for all PTP Ports (portNumber 1 and higher); otherwise, externalPortConfiguration.desiredState is
24 configured to TimeReceiverPort for one PTP Port, and is configured to TimeTransmitterPort or
25 PassivePort for other PTP Ports; and
- 26 d) Each PTP Instance shall have a corresponding PtpInstanceSyncStatus state machine (see 18.4).

28 **18.4 PtpInstanceSyncStatus state machine**

29 This state machine operates within the PTP Instance, to determine whether it is synchronized according to
30 the requirements of the respective application or profile standard (see 3.20a).

31 **18.4.1 PtpInstanceSyncStatus state machine global variables**

32 The following variables are used in the state diagram in Figure 18-2 (in 18.4.4):

33 **18.4.1.1 isSynced:** The synchronization status of the PTP Instance. The variable is a Boolean, and is TRUE
34 if synchronization is sufficient according to the operation of the PtpInstanceSyncStatus state machine and
35 FALSE otherwise. An application or profile standard can set the offsetFromTimeTransmitterMax
36 (see 18.4.1.3), rxSyncCountTimeReceiverPThresh (see 18.4.1.5), offsetMaxExceededCountThresh
37 (see 18.4.1.7), and offsetMaxMetCountThresh (see 18.4.1.9) according to its requirements.

38 **18.4.1.2 offsetFromTimeTransmitter:** The value of the managed object
39 currentDS.offsetFromTimeTransmitter (see 14.3.3).

1 **18.4.1.3 offsetFromTimeTransmitterMax:** The value of the managed object
 2 ptInstanceSyncDS.offsetFromTimeTransmitterMax (see 14.7a.3)

3 **18.4.1.4 rxSyncCountTimeReceiverP:** The value of the managed object
 4 portStatisticsDS.rxSyncCountTimeReceiverP (see 14.10.19a).

5 **18.4.1.5 rxSyncCountTimeReceiverPThresh:** The value of the managed object
 6 ptInstanceSyncDS.rxSyncCountTimeReceiverPThresh (see 14.7a.4)

7 **18.4.1.6 offsetMaxExceededCount:** The current number of consecutive exceedances of
 8 offsetFromTimeTransmitterMax (see 18.4.1.3) by offsetFromTimeTransmitter (see 18.4.1.2). The data type
 9 of offsetMaxExceededCount is UInteger32.

10 **18.4.1.7 offsetMaxExceededCountThresh:** The threshold (see 14.7a.5) for the number of consecutive
 11 exceedances of offsetFromTimeTransmitterMax (see 18.4.1.3) by offsetFromTimeTransmitter (see
 12 18.4.1.2), at which isSynced for the PTP Instance is no longer TRUE (see 18.4.3.3).

13 **18.4.1.8 offsetMaxMetCount:** The current number of consecutive occurrences of
 14 offsetFromTimeTransmitter (see 18.4.1.2) being within offsetFromTimeTransmitterMax (see 18.4.1.3). The
 15 data type of offsetMaxMetCount is UInteger32.

16 **18.4.1.9 offsetMaxMetCountThresh:** The threshold (see 14.7a.6) for the number of consecutive
 17 occurrences of offsetFromTimeTransmitter (see 18.4.1.2) being within offsetFromTimeTransmitterMax
 18 (see 18.4.1.3), at which isSynced for the PTP Instance is changed from FALSE to TRUE (see 18.4.3.3).

19 **18.4.2 PtInstanceSyncStatus state machine local variables**

20 **18.4.2.1 lastRxSyncCountTimeReceiverP:** Holds the last value of rxSyncCountTimeReceiverP
 21 (see 18.4.1.4) that the state machine read for the TimeReceiverPort, as part of monitoring for a sync event
 22 message timeout.

23 **18.4.3 PtInstanceSyncStatus state machine functions**

24 **18.4.3.1 isGptpCapable():** This function returns a Boolean value that is TRUE when ptpPortEnabled
 25 (see 10.2.5.13) and asCapable (see 10.2.5.1) are TRUE, as determined by the function portIsCapable, for at
 26 least one PTP Port (i.e., port for which portNumber is not zero).

```
27 isGptpCapable()
28 {
29     if (!instanceEnable)
30         return (FALSE);
31     for (int i = 1; i <= numberPorts; i++)
32         // see 8.6.2.8 for numberPorts
33     {
34         if (portIsCapable (i))
35             return (TRUE);
36     }
37     return (FALSE);
38 }
```

39 **18.4.3.2 isGm():** This function determines if the PTP Instance is a Grandmaster PTP Instance (TRUE) or
 40 not (FALSE) by searching the selectedState array (see 10.2.4.20) for absence or presence of at least one
 41 value equal to TimeReceiverPort (see Table 10-2).

42 isGm()

```

1 {
2     for (int i = 1; i <= numberPorts; i++)
3         // see 8.6.2.8 for numberPorts
4     {
5         if (selectedState[i] == TimeReceiverPort)
6             return (FALSE);
7     }
8     return (TRUE);
9 }
```

10 **18.4.3.3 portIsCapable(j):** This function returns a Boolean value that is TRUE if the PTP Port whose
11 portList index is j is enabled and asCapable is TRUE, and FALSE otherwise.

```

12 portIsCapable (j)
13 {
14     if (ptpPortEnabled && asCapable)
15         return (TRUE);
16     else
17         return (FALSE);
18 }
```

19 **18.4.4 State diagram**

20 The PtpInstanceSyncStatus state machine shall implement the function specified by the state diagram in
21 Figure 18-2, the variables specified in 18.4.1 and 18.4.2, and the functions specified in 18.4.3.

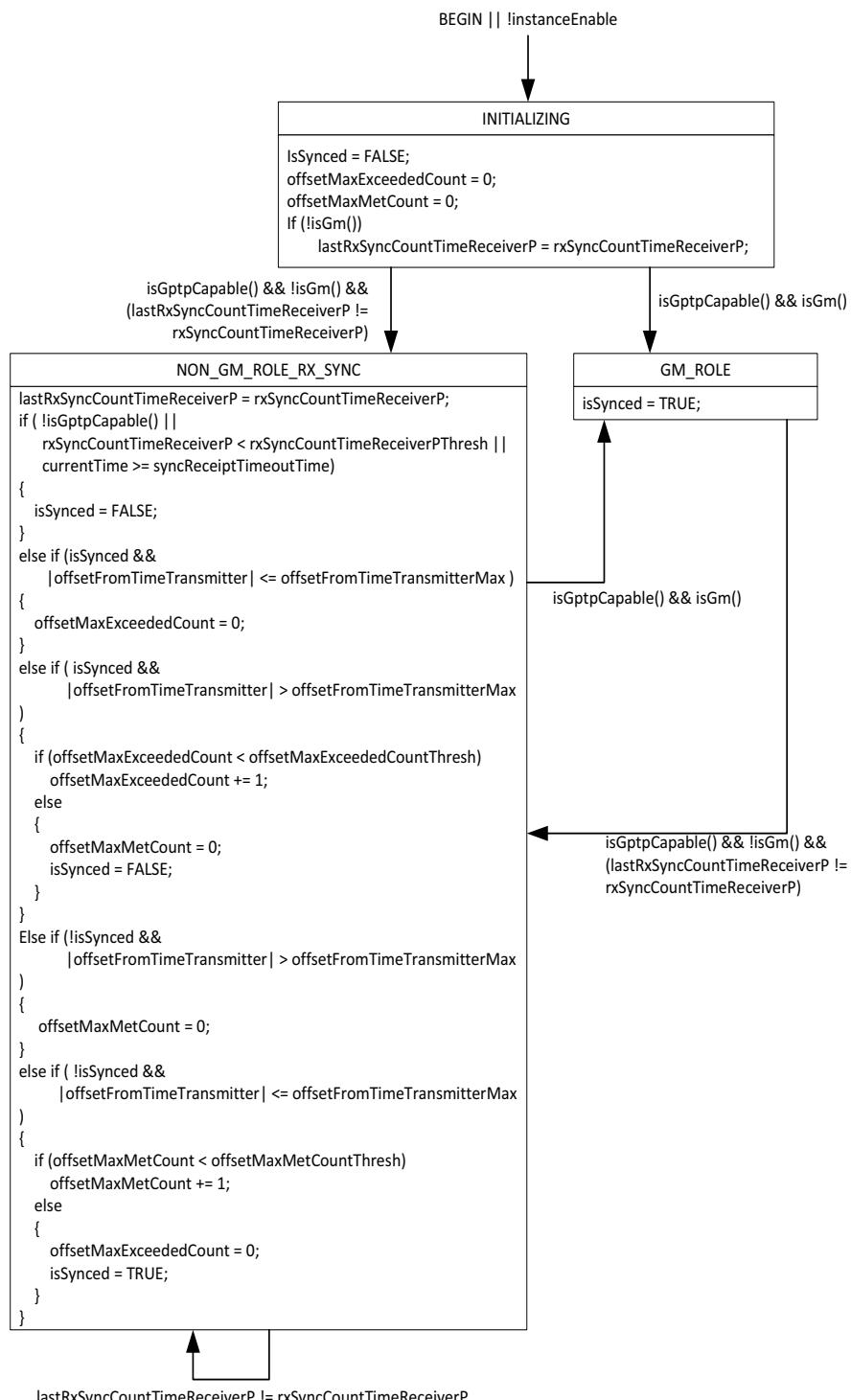


Figure 18-2—PtpInstanceSyncStatus state machine

1 18.5 HotStandbySystem state machine

2 This state machine interacts with the primary and secondary PTP Instances in order to provide a single
 3 redundant time to the application.

4 18.5.1 HotStandbySystem state machine global variables

5 **18.5.1.1 hotStandbySystemState:** State of the HotStandbySystem. The variable is an enumeration that
 6 takes one of the following values:

- 7 a) INIT: Initialization after the HotStandbySystem powers on and is enabled. In this state, the system is
 8 waiting for both PTP Instances to synchronize.
- 9 b) REDUNDANT: Both PTP Instances are synchronized according to the requirements of the
 10 respective application or profile standard (see 3.20a). Time synchronization is redundant.

11 NOTE—A PTP Instance is implicitly considered to be synchronized if it is a Grandmaster PTP Instance (see 18.4.3).

- 12 c) NOT_REDUNDANT: One PTP Instance is synchronized, and the other PTP Instance is faulted (not
 13 synchronized). Time synchronization continues to meet the requirements of the respective
 14 application or profile standard (see 3.20a). Time synchronization is not redundant.
- 15 d) OUT_OF_SYNC: The HotStandbySystem can adjust phase/frequency of its local time using the
 16 data stored while the system was in the REDUNDANT or NOT_REDUNDANT state, but the local
 17 time will eventually drift relative to other time-aware systems. During OUT_OF_SYNC state, time
 18 synchronization might not meet the requirements of the respective application or profile standard
 19 (see 3.20a).

20 **18.5.1.2 hotStandbySystemEnable:** A Boolean variable used to enable/disable the HotStandbySystem.

21 **18.5.1.3 hotStandbySystemLogSyncTimeThresh:** The value of the managed object
 22 hotStandbySystemLogSyncTimeThreshold (see 14.19.9) of the hotStandbySystemDS. The data type for
 23 hotStandbySystemLogSyncTimeThreshold is Integer8.

24 **18.5.1.4 primarySecondaryOffset:** The value of the managed object
 25 hotStandbySystemDS.primarySecondaryOffset (see 14.19.7), which is the absolute value of the difference
 26 between the clockTimeReceiverTime variables (see 10.2.4.3) of the primary and secondary PTP Instances.

27 18.5.2 HotStandbySystem state machine local variables

28 **18.5.2.1 primary:** Reference to the data sets of the PTP Instance configured to use the primary
 29 domainNumber as specified by the respective application or profile standard (see 3.20a). This references an
 30 element of the instanceList specified in 14.1.

31 **18.5.2.2 secondary:** Reference to the data sets of the PTP Instance configured to use the secondary
 32 domainNumber as specified by the respective application or profile standard (see 3.20a). This references an
 33 element of the instanceList specified in 14.1.

34 **18.5.2.3 TEMP:** A temporary variable used to reduce clutter in the state diagram (see Figure 18-3). The data
 35 type for TEMP is Integer16.

36 **18.5.2.4 transitionTime:** The value of currentTime (see 10.2.4.12) after which the hotStandbySystem
 37 transitions from the OUT_OF_SYNC state to either the NOT_REDUNDANT or REDUNDANT state, or
 38 from the NOT_REDUNDANT to the REDUNDANT state, if all other conditions for the respective
 39 transition are met. The data type for transitionTime is UScaledNs.

1 **18.5.2.5 primarySecondaryOffsetThresh:** The value of the managed object
 2 hotStandbySystemDS.primarySecondaryOffsetThresh, which is the threshold for primarySecondaryOffset
 3 (see 18.5.1.4), above which the hotStandbySystemState transitions from REDUNDANT to
 4 NOT_REDUNDANT, or stays in NOT_REDUNDANT or OUT_OF_SYNC even if other conditions for
 5 transitioning to REDUNDANT are satisfied.

6 **18.5.3 HotStandbySystem requirements**

7 **18.5.3.1 Primary grandmaster**

8 When the primary PTP Instance is a Grandmaster PTP Instance (i.e., no PTP Port in TimeReceiverPort
 9 state), the HotStandbySystem shall transfer phase, frequency, clockSourceTimeBaseIndicator (see 10.2.4.8),
 10 clockSourceLastGmPhaseChange (see 10.2.4.10), and clockSourceLastGmFreqChange (see 10.2.4.11) from
 11 the ClockSource to the ClockTimeTransmitter of the primary PTP Instance (see Figure 18-1). If no external
 12 source of time is implemented, the ClockSource is equivalent to the LocalClock of this PTP Instance.

13 When the primary PTP Instance is grandmaster, there is no requirement for the HotStandbySystem to
 14 receive time from the ClockTimeReceiver of the secondary PTP Instance.

15 **18.5.3.2 Secondary grandmaster**

16 **18.5.3.2.1 HotStandbySystem in REDUNDANT state**

17 When the secondary PTP Instance is a Grandmaster PTP Instance (i.e., no PTP Port in TimeReceiverPort
 18 state), and the hotStandbySystemState is REDUNDANT, the HotStandbySystem shall transfer phase,
 19 frequency, clockSourceTimeBaseIndicator (see 10.2.4.8), clockSourceLastGmPhaseChange (see 10.2.4.10),
 20 and clockSourceLastGmFreqChange (see 10.2.4.11) from the ClockTimeReceiver of the primary PTP
 21 Instance to the ClockTimeTransmitter of the secondary PTP Instance (see Figure 18-1). By using phase from
 22 the primary PTP Instance, the secondary grandmaster can maintain continuity in the event of a fault in the
 23 primary domain.

24 **18.5.3.2.2 HotStandbySystem in NOT_REDUNDANT state**

25 When the secondary PTP Instance is a Grandmaster PTP Instance (i.e., no PTP Port in TimeReceiverPort
 26 state), and the hotStandbySystemState is NOT_REDUNDANT, which means that isSynced for the primary
 27 PTP Instance is FALSE and isSynced for the secondary PTP Instance is TRUE, the HotStandbySystem shall
 28 transfer phase, frequency, clockSourceTimeBaseIndicator (see 10.2.4.8), clockSourceLastGmPhaseChange
 29 (see 10.2.4.10), and clockSourceLastGmFreqChange (see 10.2.4.11) from the ClockSource to the
 30 ClockTimeTransmitter of the secondary PTP Instance (see Figure 18-1). If no external source of time is
 31 implemented, the ClockSource is equivalent to the LocalClock of this PTP Instance.

32 The secondary PTP Instance is considered synchronized because it is a Grandmaster PTP Instance
 33 (see 18.4.4 and Figure 18-2). Since the hotStandbySystemState is NOT_REDUNDANT, this means that
 34 either the primary PTP Instance is not synchronized or primarySecondaryOffset exceeds
 35 primarySecondaryOffsetThresh. However, in the former case primarySecondaryOffset can exceed
 36 primarySecondaryOffsetThresh because in this case the primary and secondary PTP Instances receive
 37 timing from different sources. The result is that the HotStandbySystem remains in the NOT_REDUNDANT
 38 state, even if the primary PTP Instance subsequently becomes synchronized again by the primary
 39 Grandmaster PTP Instance, because primarySecondaryOffset continues to exceed
 40 primarySecondaryOffsetThresh. This situation does not occur if it is the primary PTP Instance that is
 41 synchronized and the secondary PTP Instance that is not synchronized due to some fault condition. When
 42 the fault condition is cleared, the secondary Grandmaster PTP Instance ClockTimeTransmitter would
 43 receive timing from the primary PTP Instance ClockTimeReceiver (see 18.5.3.2.1) and, as a result, it would
 44 become synchronized again and the hotStandbySystemState would change to REDUNDANT.

1 The situation described in the previous paragraph is avoided if the optional split functionality (see 18.5.3.4)
 2 is used. In this case, timing is transferred directly from the secondary SiteSync entity to the primary
 3 SiteSync entity. This will synchronize the primary PTP Instance to the secondary PTP Instance; hence, the
 4 primarySecondaryOffset changes to within primarySecondaryOffsetThresh. The HotStandbySystem state
 5 changes to REDUNDANT and the secondary Grandmaster PTP Instance begins receiving timing from the
 6 primary PTP Instance. However, even if the optional split functionality is not used, an application can
 7 choose to re-initialize the secondary PTP Instance after the primary PTP Instance becomes synchronized. If
 8 this is done, the secondary Grandmaster PTP Instance receives timing from the primary PTP Instance
 9 ClockTimeReceiver after initialization and hotStandbySystemState is REDUNDANT.

10 NOTE—If the secondary PTP Instance is re-initialized after the primary PTP Instance becomes synchronized, this can
 11 cause a jump in time when the secondary PTP Instance synchronizes to the primary PTP Instance.

12 On initialization, if the primary PTP Instance is synchronized, the secondary PTP Instance shall receive
 13 timing from the primary PTP Instance. When the conditions as specified the HotStandbySystem state
 14 machine (see Figure 18-3) are satisfied, the hotStandbySystemState machine transitions to the
 15 REDUNDANT state.

16 **18.5.3.2.3 Transition of hotStandbySystemState from REDUNDANT to NOT_REDUNDANT**

17 The secondary grandmaster shall conform to the time synchronization requirements of the respective
 18 application or profile standard (see 3.20a) during a transition of HotStandbySystemState from
 19 REDUNDANT to NOT_REDUNDANT due to a fault in its primary PTP Instance (i.e., primary.isSynced
 20 == FALSE).

21 NOTE 1—The secondary grandmaster is responsible for maintaining continuity (no “jump” in time) when a fault occurs
 22 in the primary grandmaster.

23 NOTE 2—The primary domain global variables lastGmPhaseChange and lastGmFrequencyChange are propagated
 24 throughout the secondary domain, and can be used to compensate for any phase or frequency jump that occurs when the
 25 HotStandbySystem state becomes NOT_REDUNDANT.

26 **18.5.3.3 TimeReceiver HotStandbySystem**

27 **18.5.3.3.1 General**

28 A HotStandbySystem is said to be a TimeReceiver HotStandbySystem if and only if neither PTP Instance is
 29 a Grandmaster PTP Instance.

30 **18.5.3.3.2 HotStandbySystem in REDUNDANT state**

31 When the HotStandbySystem is a timeReceiver (i.e., neither PTP Instance is a grandmaster), and
 32 hotStandbySystemState is REDUNDANT, the HotStandbySystem shall transfer phase, frequency,
 33 gmTimeBaseIndicator (see 10.2.4.15), lastGmPhaseChange (see 10.2.4.16), and lastGmFrequencyChange
 34 (see 10.2.4.17) from the ClockTimeReceiver of the primary PTP Instance to the ClockTarget (application) if
 35 a ClockTarget is present.

36 **18.5.3.3.3 HotStandbySystem in NOT_REDUNDANT state**

37 **18.5.3.3.3.1 Transfer of synchronized time to the ClockTarget**

38 When the HotStandbySystem is a timeReceiver, and the HotStandbySystem is in the NOT_REDUNDANT
 39 state, the HotStandbySystem shall transfer phase, frequency, gmTimeBaseIndicator (see 10.2.4.15),
 40 lastGmPhaseChange (see 10.2.4.16), and lastGmFrequencyChange (see 10.2.4.17) from the

1 ClockTimeReceiver of the synchronized PTP Instance (i.e., isSynced == TRUE) to the ClockTarget if a
 2 ClockTarget is present.

3 18.5.3.4 Split functionality

4 The HotStandbySystem may provide an optional split functionality. If the managed object
 5 hotStandbySystemDS.hotstandbySystemSplitFunctionality is set to TRUE, the split functionality is invoked.
 6 If the managed object hotStandbySystemDS.hotstandbySystemSplitFunctionality is set to FALSE, the split
 7 functionality is not invoked. The split functionality shall provide an interworking function (IWF) that
 8 transfers time synchronization information from the primary PTP Instance to the secondary PTP Instance
 9 when isSynced is FALSE (e.g., the timeReceiver port of the PTP Instance is in failure condition,
 10 see 18.4.1.1) for the secondary PTP Instance and isSynced is TRUE (see 18.4.1.1) for the primary PTP
 11 Instance, or from the secondary PTP Instance to the primary PTP Instance when isSynced is FALSE for the
 12 primary PTP Instance and isSynced is TRUE for the secondary PTP Instance. The IWF provides the most
 13 recently received PortSyncSync structure at the timeReceiver port of the synchronized PTP Instance
 14 (isSynced == TRUE, see 18.4.1.1) to the unsynchronized PTP Instance (isSynced == FALSE) SiteSyncSync
 15 entity, in order to generate a PortSyncSync structure that is used by the timeTransmitter ports of the
 16 unsynchronized PTP Instance, as follows:

- 17 a) The domainNumber of the received PortSyncSync structure is changed by the IWF from the
 18 synchronized PTP Instance domainNumber to the unsynchronized PTP Instance domainNumber;
- 19 b) The localPortNumber of the received PortSyncSync structure is changed by the IWF to the
 20 portNumber of the unsynchronized PTP Instance timeReceiver port; and
- 21 c) All other members (i.e., syncReceiptTimeoutTime, followUpCorrectionField, sourcePortIdentity,
 22 logMessageInterval, preciseOriginTimestamp, upstreamTxTime, rateRatio, gmTimeBaseIndicator,
 23 lastGmPhaseChange, and lastGmFreqChange) of the received PortSyncSync structure of the
 24 synchronized PTP Instance are provided to the unsynchronized PTP Instance SiteSync entity
 25 unchanged.
- 26 d) The flags leap61, leap59, currentUtcOffsetValid, ptpTimescale, timeTraceable, and
 27 frequencyTraceable, and the fields currentUtcOffset and timeSource of received Announce
 28 messages on the timeReceiver port of the unsynchronized PTP Instance are ignored. The IWF
 29 provides the values of these corresponding members of the timePropertiesDS of the synchronized
 30 PTP Instance to the unsynchronized PTP Instance. These values are then copied to:
 - 31 i) The respective members of the timePropertiesDS of the unsynchronized PTP Instance;
 - 32 ii) The unsynchronized PTP Instance global variables leap61, leap59, currentUtcOffsetValid,
 33 ptpTimescale, timeTraceable, and frequencyTraceable, and the fields currentUtcOffset
 34 and timeSource, respectively; and
 - 35 iii) The unsynchronized PTP Instance global variables annLeap61, annLeap59,
 36 annCurrentUtcOffsetValid, annPtpTimescale, annTimeTraceable, and
 37 annFrequencyTraceable, and the fields annCurrentUtcOffset and annTimeSource,
 38 respectively.
- 39 e) The messagePriorityVector (see 10.3.4 and 10.3.5) information of received Announce messages on
 40 the timeReceiver port of the unsynchronized PTP Instance is ignored. The IWF provides the values of
 41 the corresponding members of the synchronized PTP Instance parentDS, i.e., grandmasterIdentity,
 42 grandmasterClockQuality.clockClass, grandmasterClockQuality.clockAccuracy,
 43 grandmasterClockQuality.offsetScaledLogVariance, grandmasterPriority1, grandmasterPriority2,
 44 which are copied to:
 - 45 i) The corresponding members of the parentDS of the unsynchronized PTP Instance; and
 - 46 ii) The gmPriorityVector of the unsynchronized PTP Instance.

1 The IWF also provides the value of the stepsRemoved member of the synchronized PTP Instance
2 currentDS, which is copied to:

- 3 iii) The stepsRemoved member of the currentDS of the unsynchronized PTP Instance; and
4 iv) The gmPriorityVector of the unsynchronized PTP Instance.

5 NOTE 1—With the above, the secondary/primary PTP Instance state machines operate as though the time
6 synchronization information had been received from the secondary/primary PTP Instance timeReceiver port. The
7 SiteSync entity of the secondary/primary PTP Instance transfers the timing information to the PortSync entity of each
8 timeTransmitter port of the secondary/primary PTP Instance. Each PortSyncSync state machine computes rateRatio,
9 which now is relative to the primary/secondary PTP Instance GM. Each MDSyncSend state machine computes the fields
10 of transmitted Sync and, in the two-step case, Follow_Up messages. The copied syncReceiptTimeout time is less than
11 currentTime because sync receipt timeout has not occurred at the primary/secondary PTP Instance.

12 NOTE 2—The split functionality is used to transfer time synchronization information from the PTP Instance for which
13 isSynced is TRUE to the PTP Instance for which isSynced is FALSE. It is not meant to cover the case where isSynced is
14 FALSE for both the primary and secondary PTP Instances.

15 **18.5.3.5 Both PTP Instances have isSynced FALSE**

16 When isSynced is FALSE for both the primary and secondary PTP Instances (i.e., both are not
17 synchronized), the HotStandbySystemState is OUT_OF_SYNC and the HotStandbySystem shall transfer
18 phase, frequency, gmTimeBaseIndicator (see 10.2.4.15), lastGmPhaseChange (see 10.2.4.16), and
19 lastGmFrequencyChange (see 10.2.4.17) from the ClockTimeReceiver of the PTP Instance for which
20 isSynced was TRUE, before the HotStandbySystem transitioned to the OUT_OF_SYNC state, to the
21 ClockTarget if a ClockTarget is present.

22 When the HotStandbySystem state is OUT_OF_SYNC, time synchronization performance is not required to
23 meet the respective application or profile standard (see 3.20a) requirements. Nevertheless, in order to
24 mitigate LocalClock frequency drift, the PTP Instance should adjust the phase/frequency of its LocalClock
25 using the data stored while isSynced for that PTP Instance was TRUE and the HotStandbySystem was in the
26 REDUNDANT or NOT_REDUNDANT state.

27 **18.5.4 State diagram**

28 The HotStandbySystem state machine shall implement the function specified by the state diagram in
29 Figure 18-3, the variables specified in 18.5.1 and 18.5.2, and the state requirements specified in 18.5.3.

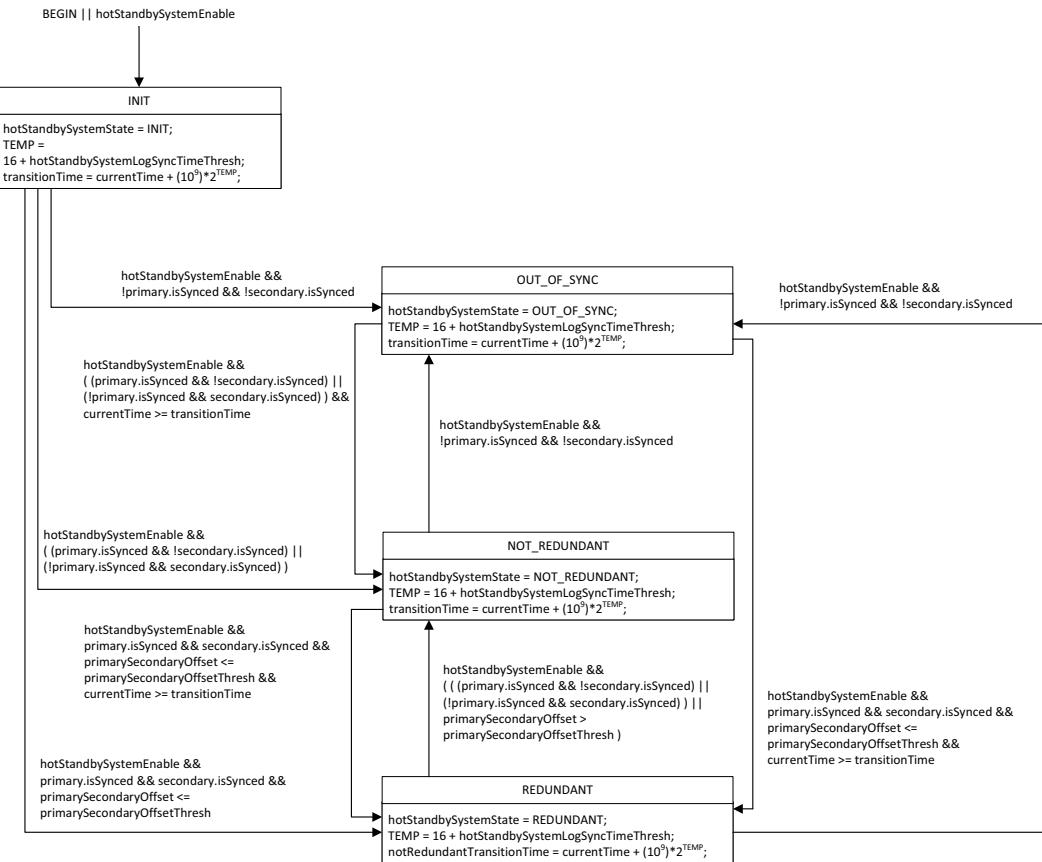


Figure 18-3—HotStandbySystem state machine

1 18.6 PrimarySecondaryOffset state machine

- 2 This state machine updates the value of the global variable `primarySecondaryOffset` (see 18.5.1.4) whenever
 3 a new value of `clockTimeReceiverTime` (see 10.2.4.3) is computed for either the primary or secondary PTP
 4 Instance. A new value of `clockTimeReceiverTime` is computed when the global variable `rcvdPSSyncCSS`
 5 (see 10.2.4.29) or the global variable `rcvdLocalClockTickCSS` (see 10.2.4.30) is set to TRUE.

6 18.6.1 State diagram

- 7 The PrimarySecondaryOffset state machine shall implement the function specified by the state diagram in
 8 Figure 18-4.

9

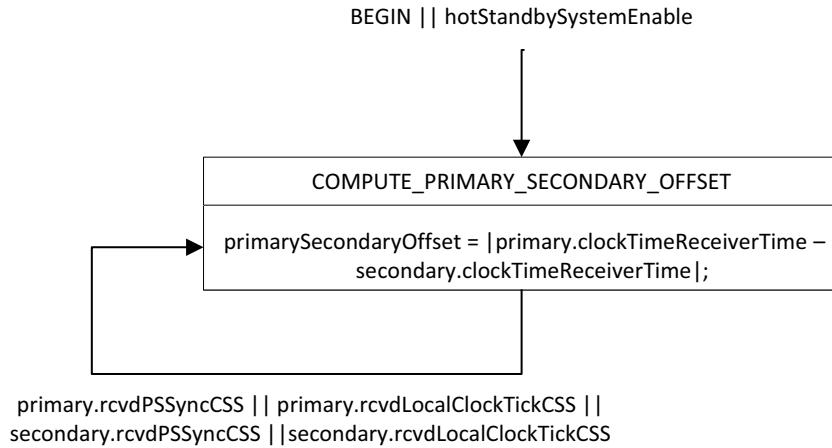


Figure 18-4—PrimarySecondaryOffset state machine

¹ Annex A

² (normative)

³ Protocol Implementation Conformance Statement (PICS) ⁴ proforma²⁵

⁵ A.1 Introduction

⁶ The supplier of a protocol implementation that is claimed to conform to this standard shall complete the
⁷ following PICS proforma.

⁸ A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of
⁹ which capabilities and options of the protocol have been implemented. The PICS can have a number of uses,
¹⁰ including the following:

- ¹¹ a) By the protocol implementer, as a checklist to reduce the risk of failure to conform to the standard
¹² through oversight;
- ¹³ b) By the supplier and acquirer—or potential acquirer—of the implementation, as a detailed indication
¹⁴ of the capabilities of the implementation, stated relative to the common basis for understanding
¹⁵ provided by the standard PICS proforma;
- ¹⁶ c) By the user—or potential user—of the implementation, as a basis for initially checking the
¹⁷ possibility of interworking with another implementation (note that, while interworking can never be
¹⁸ guaranteed, failure to interwork can often be predicted from incompatible PICS);
- ¹⁹ d) By a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for
²⁰ conformance of the implementation.

²¹ A.2 Abbreviations and special symbols

²² A.2.1 Status symbols

²³ M	mandatory
²⁴ O	optional
²⁵ O.n	optional, but support of at least one of the group of options labeled by the same numeral n ²⁶ is required
²⁷ X	prohibited
²⁸ pred:	conditional-item symbol, including predicate identification (see A.3.4)
²⁹ \neg	logical negation, applied to a conditional item's predicate

³⁰ A.2.2 General abbreviations

³¹ N/A	not applicable
³² PICS	Protocol Implementation Conformance Statement

²⁵ Copyright release for PICS proformas: Users of this standard may freely reproduce the PICS proforma in this annex so that it can be used for its intended purpose and may further publish the completed PICS.

1 A.3 Instructions for completing the PICS proforma

2 A.3.1 General structure of the PICS proforma

3 The first part of the PICS proforma, implementation identification and protocol summary, is to be completed
4 as indicated with the information necessary to identify fully both the supplier and the implementation.

5 The main part of the PICS proforma is a fixed-format questionnaire, divided into several subclauses, each
6 containing a number of individual items. Answers to the questionnaire items are to be provided in the
7 rightmost column, either by simply marking an answer to indicate a restricted choice (usually Yes or No), or
8 by entering a value or a set or range of values. (Note that there are some items where two or more choices
9 from a set of possible answers can apply; all relevant choices are to be marked.)

10 Each item is identified by an item reference in the first column. The second column contains the question to
11 be answered; the third column records the status of the item—whether support is mandatory, optional, or
12 conditional (see also A.3.4). The fourth column contains the reference or references to the material that
13 specifies the item in the main body of this standard, and the fifth column provides the space for the answers.

14 A supplier might also provide (or be required to provide) further information, categorized as either
15 Additional Information or Exception Information. When present, each kind of further information is to be
16 provided in a further subclause of items labeled A_i or X_i , respectively, for cross-referencing purposes, where
17 i is any unambiguous identification for the item (e.g., simply a numeral). There are no other restrictions on
18 its format and presentation.

19 A completed PICS proforma, including any Additional Information and Exception Information, is the
20 Protocol Implementation Conformance Statement for the implementation in question.

21 NOTE—Where an implementation is capable of being configured in more than one way, a single PICS might be able to
22 describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering
23 some subset of the implementation's configuration capabilities, in case that makes for easier and clearer presentation of
24 the information.

25 A.3.2 Additional Information

26 Items of Additional Information allow a supplier to provide further information intended to assist the
27 interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS
28 can be considered complete without any such information. Examples might be an outline of the ways in
29 which a (single) implementation can be set up to operate in a variety of environments and configurations, or
30 information about aspects of the implementation that are outside the scope of this standard but that have a
31 bearing upon the answers to some items.

32 References to items of Additional Information may be entered next to any answer in the questionnaire and
33 may be included in items of Exception Information.

34 A.3.3 Exception Information

35 It occasionally happens that a supplier will wish to answer an item with mandatory status (after any
36 conditions have been applied) in a way that conflicts with the indicated requirement. No preprinted answer
37 will be found in the Support column for this; instead, the supplier shall write the missing answer into the
38 Support column, together with an X_i reference to an item of Exception Information, and shall provide the
39 appropriate rationale in the Exception item itself.

- 1 An implementation for which an Exception item is required in this way does not conform to this standard.
- 2 NOTE—A possible reason for the situation described above is that a defect in this standard has been reported, a correction for which is expected to change the requirement not met by the implementation.

4 A.3.4 Conditional status

5 A.3.4.1 Conditional items

6 The PICS proforma contains a number of conditional items. These are items for which both the applicability
7 of the item itself, and its status if it does apply—mandatory or optional—are dependent upon whether or not
8 certain other items are supported.

9 Where a group of items is subject to the same condition for applicability, a separate preliminary question
10 about the condition appears at the head of the group, with an instruction to skip to a later point in the
11 questionnaire if the Not Applicable (N/A) answer is selected. Otherwise, individual conditional items are
12 indicated by a conditional symbol in the Status column.

13 A conditional symbol is of the form “**pred:** S” where **pred** is a predicate as described in A.3.4.2, and S is a
14 status symbol, M or O.

15 If the value of the predicate is true (see A.3.4.2), the conditional item is applicable, and its status is indicated
16 by the status symbol following the predicate; the answer column is to be marked in the usual way. If the
17 value of the predicate is false, the N/A answer is to be marked.

18 A.3.4.2 Predicates

19 A predicate is one of the following:

- 20 a) An item-reference for an item in the PICS proforma; the value of the predicate is true if the item is
21 marked as supported, and is false otherwise;
- 22 b) A predicate-name, for a predicate defined as a Boolean expression constructed by combining item-
23 references using the Boolean operator OR; the value of the predicate is true if one or more of the
24 items is marked as supported;
- 25 c) The logical negation symbol “ \neg ” prefixed to an item-reference or predicate-name; the value of the
26 predicate is true if the value of the predicate formed by omitting the “ \neg ” symbol is false, and vice
27 versa.

28 Each item whose reference is used in a predicate or predicate definition, or in a preliminary question for
29 grouped conditional items, is indicated by an asterisk in the Item column.

¹ A.4 PICS proforma for IEEE Std 802.1AS-2020

A.4.1 Implementation identification

Supplier	
Contact point for queries about the PICS	
Implementation Name(s) and Version(s)	
Other information necessary for full identification—e.g., name(s) and version(s) of machines and/or operating system names	

² NOTE 1—Only the first three items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification.

⁴ NOTE 2—The terms Name and Version should be interpreted appropriately to correspond with a supplier’s terminology (e.g., Type, Series, Model).

6

A.4.2 Protocol summary, IEEE Std 802.1AS

Identification of protocol specification	IEEE Std 802.1AS-2020, IEEE Standard for Local and metropolitan area networks—Timing and Synchronization for Time-Sensitive Applications	
Identification of amendments and corrigenda to the PICS proforma which have been completed as part of the PICS	Amd. : Corr. :	Amd. : Corr. :
Have any Exception items been required? (See A.3.3: the answer Yes means that the implementation does not conform to IEEE Std 802.1AS-2020)	No []	Yes []

Date of Statement	
-------------------	--

7

A.5 Major capabilities

Item	Feature	Status	References	Support
DOMADD	Does the time-aware system support one or more PTP Instances with domainNumber in the range 1 to 127?	O	item e) of 5.4.2, 8.1	Yes [] No []
MINTA	Does the PTP Instance support at least one PTP Port with minimal requirements?	M	10.2.13, item c) of 5.4, A.7	Yes []
BTC	Does the PTP Instance implement the best timeTransmitter clock algorithm?	M	10.2.13, item f) of 5.4, 10.3, A.9	Yes []
SIG	Does the PTP Instance transmit Signaling messages?	M	item h) of 5.4.1, 10.4, 10.6.4, A.8	Yes []
GMCAP	Is the PTP Instance capable of acting as a Grandmaster PTP Instance?	O	10.2.13, item c) of 5.4.2, 10.1.3, A.10	Yes [] No []
BRDG	Does the PTP Instance act as a PTP Relay Instance on two or more PTP Ports?	O	item d) of 5.4.2, 5.4.3	Yes [] No []
MITT	Does the PTP Instance support media-independent timeTransmitter functionality on at least one PTP Port?	GMCAP or BRDG:M	item b) of 5.4.2, A.11	Yes [] N/A []
MIPERF	Does the PTP Instance support the performance requirements?	M	10.2.13, item j) of 5.4, A.12	Yes []
EXT	Does the PTP Instance support external port configuration?	O	item f) of 5.4.2, A.21	Yes [] No []
MDFDPP	Does the PTP Instance support media-dependent full-duplex point-to-point functionality on one or more PTP Ports?	O.1	5.5, Clause 11, A.6, A.13	Yes [] No []
MDDOT11	Does the PTP Instance support media-dependent IEEE 802.11 link functionality on one or more PTP Ports?	O.1	5.6, Clause 12, A.6, A.14	Yes [] No []
MDEPON	Does the PTP Instance support IEEE 802.3 Passive Optical Networking (EPON)?	O.1	5.7, Clause 13, A.6, A.15	Yes [] No []
MDGHN	Does the PTP Instance support media-dependent ITU-T G.hn functionality on one or more PTP Ports?	O.1	item b) of 5.8, 16.6.3, A.18	Yes [] No []
MDMOCA	Does the PTP Instance support media-dependent MoCA functionality on one or more PTP Ports?	O.1	item b) of 5.8, 16.6.2, A.17	Yes [] No []
MDCSN	Does the PTP Instance support media-dependent CSN functionality on one or more PTP Ports?	MDGHN or MDMOCA:M	5.8, Clause 16, A.6, A.16	Yes [] No []
MGT	Is management of the PTP Instance supported?	O	item j) of 5.4.2, Clause 14	Yes [] No []
RMGT	Is a remote management protocol supported?	MGT: O	item k) of 5.4.2, A.19	Yes [] No []

A.5 Major capabilities (*continued*)

Item	Feature	Status	References	Support
APPL	Does the PTP Instance support one or more of the application interfaces?	O	item i) of 5.4.2, Clause 9, A.20	Yes [] No []
HOTSTDBY	Does the time-aware system support hot standby as specified in Clause 17?	O	Clause 18, 14.7a, item m) of 5.4.2, 9.3.3.4, 9.4.3.4, 9.5.3.4, 9.6.2.6	Yes [] No []
DRFTRK	Does the PTP Instance support the Drift_Tracking TLV as specified in 10.2.4.26, 10.2.4.27, 10.2.4.28, 11.2.14, 11.2.15, and 11.4.4?	O	item n) of 5.4.2, 10.2.4.26, 10.2.4.27, 10.2.4.28, 11.2.14, 11.2.15, and 11.4.4	Yes [] No []

1

A.6 Media access control methods

Item	Feature	Status	References	Support
MAC-IEEE-802.3 MAC-IEEE-802.11	Which MAC methods are implemented in conformance with the relevant MAC standards?	O:2	11.1	Yes [] No []
		O:2	12.1	Yes [] No []
MAC-1	Has a PICS been completed for each of the MAC methods implemented as required by the relevant MAC Standards?	M		Yes []
MAC-2	Do all the MAC methods implemented support the MAC Timing aware Service as specified?	M	Clause 11 Clause 12 Clause 13	Yes []

2

A.7 Minimal time-aware system

Item	Feature	Status	References	Support
MINTA-1	Do all PTP Instances of the device implement the functionality specified by the SiteSyncSync state machine in Figure 10-3 in compliance with the requirements of 10.2.7?	M	item f) of 5.4, 10.2.7	Yes []
MINTA-2	Do all PTP Instances of the device implement the functionality specified by the PortSyncSyncReceive state machine in Figure 10-4 on each PTP Port in compliance with the requirements of 10.2.8?	M	item d) of 5.4	Yes []

A.7 Minimal time-aware system (*continued*)

Item	Feature	Status	References	Support
MINTA-3	Do all PTP Instances of the device implement the functionality specified by the ClockTimeReceiverSync state machine in Figure 10-9 in compliance with the requirements of 10.2.13?	M	10.2.13, item e) of 5.4	Yes []
MINTA-4	For all PTP Instances of the device, does the PTP Port sending a Signaling message that contains a message interval request TLV adjust its syncReceiptTimeoutTimeInterval of this PTP Instance in compliance with the requirements of 10.6.4.3.7 and Table 10-16?	SIG:M	10.6.4.3.7	Yes [] N/A []
MINTA-5	Is the clockIdentity constructed in compliance with the requirements of 8.5.2.2?	M	8.5.2.2	Yes []
MINTA-6	Is the domainNumber for all transmitted messages in the range 0 through 127, in compliance with the requirements of 8.1?	M	8.1	Yes []
MINTA-7	Is the majorSdId 0x1 and the minorSdId 0x0 for all transmitted gPTP domain messages?	M	8.1	Yes []
MINTA-8	Is the domainNumber for at least one of the gPTP domains supported by the time-aware system, in compliance with the requirements of 8.1?	M	8.1	Yes []
MINTA-9	Is the IEEE 802.1AS time of domain 0 measured relative to the PTP epoch in compliance with the requirements of 8.2.2?	M	8.2.2	Yes []
MINTA-10	If path delay asymmetry is modeled by this PTP Instance does it comply with the requirements of 8.3?	O	8.3	Yes [] No [] N/A []
MINTA-11	Do all derived data types that are transmitted in IEEE 802.1AS messages and headers comply with 6.4.4?	M	6.4.4	Yes []
MINTA-12	Is the granularity of the local clock 40 ns or better in compliance with the requirements of B.1.2?	M	B.1.2	Yes []
MINTA-13	Is the frequency of the local clock relative to TAI \pm 100 ppm in compliance with the requirements of B.1.1?	M	B.1.1	Yes []
MINTA-14	Does the PTP Instance ignore non-propagating TLVs of Announce and Signaling messages that it cannot parse, and attempt to parse the next TLV, in compliance with the requirements of 10.6.1?	M	10.6.1	Yes []
MINTA-15	Does the PTP Instance support the state machines related to signaling gPTP capability?	M	item g) of 5.4.1, 10.4.2	Yes []

A.7 Minimal time-aware system (*continued*)

Item	Feature	Status	References	Support
MINTA-16	For receive of all messages and for transmit of all messages except Announce, does the PTP Instance support the message requirements?	M	item h) of 5.4. 1, 10.5, 10.6, 10.7	Yes []
MINTA-17	Does the PTP Instance support the gPTP requirements specified in Clause 8, including the PTP Instance attributes?	M	item a) of 5.4, Clause 8, 8.6.2	Yes []
MINTA-18	Does the PTP Instance support the requirements for time-synchronization state machines?	M	item b) of 5.4	Yes []
MINTA-19	Does the PTP Instance implement the path trace TLV (i.e., process this TLV when received in an Announce message, and attach this TLV to a transmitted Announce message unless the TLV would cause the maximum frame size to be exceeded)?	M	10.3.11, 10.3.13, 10.3.14, 10.3.16	Yes []
MINTA-20	Does the PTP Instance forward TLVs as required?	M	10.6.1	Yes []
MINTA-21	Do the PTP Ports of this PTP Instance implement the functionality of the GptpCapableIntervalSetting state machine in compliance with the requirements of 10.4.3 and Figure 10-23?	MINTA-15:O	10.4.3, item h) of 5.4.2, item h) of 5.4.1	Yes [] No []
MINTA-22	Does the PTP Instance implement the PtpInstanceSyncStatus state machine of 18.4 and the PtpInstanceStateDS of 14.7a?	¬HOTSTDBY: O	18.4, 14.7a	Yes [] No []

1

A.8 Signaling

Item	Feature	Status	References	Support
SIG-1	Do the sequence numbers of Signaling messages comply with the requirements of 10.5.7?	SIG:M	10.5.7	Yes []
SIG-2	Does the Signaling message body comply with the requirements of 10.6.4.1 and Table 10-13?	SIG:M	10.6.4.1	Yes []
SIG-3	Does the Signaling message header comply with the requirements of 10.6.2?	SIG:M	10.6.2	Yes []
SIG-4	Are all Signaling message reserved fields equal to 0 in compliance with the requirements of 10.6.1?	SIG:M	10.6.1	Yes []

A.8 Signaling

Item	Feature	Status	References	Support
SIG-5	Is the destination MAC address for all Signaling messages equal to 01:80:C2:00:00:0E in compliance with the requirements of 10.5.3?	SIG:M	10.5.3	Yes []
SIG-6	Is the EtherType for all Signaling messages equal to 88-F7 in compliance with the requirements of 10.5.4?	SIG:M	10.5.4	Yes []
SIG-7	Does the message interval request TLV for Signaling messages comply with the requirements in 10.6.4.3?	BTC-23:M, MIMSTR-16:M, MDFDPP-6:M	10.6.4.3	Yes []

A.9 Best time Transmitter clock

Item	Feature	Status	References	Support
BTC-1	Does the PTP Instance implement the functionality specified by the PortAnnounceReceive state machine in Figure 10-13 on each PTP Port in compliance with the requirements of 10.3.11?	M	10.3.11	Yes []
BTC-2	Does the PTP Instance implement the functionality specified by the PortAnnounceInformation state machine in Figure 10-14 on each PTP Port in compliance with the requirements of 10.3.12?	M	10.3.12	Yes []
BTC-3	Does the PTP Instance implement the functionality specified by the PortStateSelection state machine in Figure 10-15 on each PTP Port in compliance with the requirements of 10.3.13? NOTE—There is one instance of the PortStateSelection state machine for the PTP Instance, for each gPTP domain. Some of the PortStateSelection state machine computations are performed for each PTP Port, and some of the computations are performed for the PTP Instance as a whole (and all the computations are performed for each gPTP domain).	M	10.3.13	Yes []
BTC-4	If the value of clockA's SystemIdentity is less than that of clockB, is clockA selected as Grandmaster PTP Instance in compliance with the requirements of 10.3.2?	M	10.3.2	Yes []
BTC-5	Does the value of priority1 comply with the requirements of 8.6.2.1?	M	8.6.2.1	Yes []

A.9 Best timeTransmitter clock (*continued*)

Item	Feature	Status	References	Support
BTC-6	Does the value of clockClass comply with the requirements of 8.6.2.2?	M	8.6.2.2	Yes []
BTC-7	Does the value of priority2 comply with the requirements of 8.6.2.5?	M	8.6.2.5	Yes []
BTC-8	Does the value of clockAccuracy comply with requirements of 8.6.2.3?	M	8.6.2.3	Yes []
BTC-9	Does the value of offsetScaledVariance comply with the requirements of 8.6.2.4?	M	8.6.2.4	Yes []
BTC-10	Does the value of timeSource comply with requirements of 8.6.2.7 and Table 8-2?	M	8.6.2.7	Yes []
BTC-11	Is the PTP Port number equal to 1 in compliance with the requirements of 8.5.2.3?	BRDG:M	8.5.2.3	Yes [] N/A []
BTC-12	Are the PTP Ports numbered 1 through N for each of N PTP Ports in compliance with the requirements of 8.5.2.3?	M	8.5.2.3	Yes []
BTC-13	Does the clockIdentity field comply with the requirements of 8.5.2.2?	M	8.5.2.2	Yes []
BTC-14	When no grandmaster-capable PTP Instance is available does the behavior of the PTP Instance comply with the requirements of 10.2.13.2, i.e., the clockTimeReceiverTime should be provided by the local clock?	M	10.2.13.2	Yes []
BTC-15	Does the value of announceReceiptTimeout comply with the requirements of 10.7.3.2?	M	10.7.3.2	Yes []
BTC-16	Does the TimeReceiverPort remove the PTP Port from the BTC selection after announceReceiptTimeout expires in compliance with the requirements of 10.7.3.2?	M	10.7.3.2	Yes []
BTC-17	Does the value of syncReceiptTimeout comply with the requirements of 10.7.3.1?	M	10.7.3.1	Yes []
BTC-18	Does the TimeReceiverPort remove the PTP Port from the BTC selection after syncReceiptTimeout expires in compliance with 10.7.3.1?	M	10.7.3.1	Yes []
BTC-19	Does the PTP Port sending a message interval request Signaling message adjust its announceReceiptTimeoutTimeInterval in compliance with the requirements of 10.6.4.3.8 and Table 10-17?	SIG:M	10.6.4.3.8	Yes []
BTC-20	If the PTP Instance implements the ClockSourceTime interface, does the value of lastGmPhaseChange comply with the requirements of 9.2.2 and 6.4.3.3?	O	9.2.2	Yes [] No []

A.9 Best timeTransmitter clock (*continued*)

Item	Feature	Status	References	Support
BTC-21	Does the transmitted timing information comply with the requirements of 10.3.1, including specifications for externalPortConfigurationEnabled value of false?	GMCAP:M	10.3.1	Yes [] N/A []
BTC-22	Does the PTP Instance implement BTCA requirements that are not listed in the preceding BTC rows?	M	10.3.2, 10.3.3 10.3.4, 10.3.5, 10.3.6, 10.3.8, 10.3.10	Yes []
BTC-23	Do the TimeTransmitterPorts of this PTP Instance implement the functionality of the AnnounceIntervalSetting state machine in compliance with the requirements of 10.3.17 and Figure 10-19?	BTCA:O	10.3.17, item g) of 5.4.2, item h) of 5.4.1	Yes [] No []

1

A.10 Grandmaster-capable PTP Instance

Item	Feature	Status	References	Support
	If GMCAP not supported, mark N/A.			N/A []
GMCAP-1	Does the PTP Instance implement the functionality specified by the ClockTimeTransmitterSyncSend state machine in compliance with the requirements of 10.2.9 and Figure 10-5?	GMCAP:M	10.2.9	Yes []
GMCAP-2	Does the PTP Instance implement the functionality specified by the ClockSyncOffset state machine in compliance with the requirements of 10.2.10 and Figure 10-6?	GMCAP:M	10.2.10	Yes []
GMCAP-3	Does the device implement the functionality specified by the ClockTimeTransmitterSyncReceive state machine in compliance with the requirements of 10.2.11 and Figure 10-7?	GMCAP:M	10.2.11	Yes []

A.11 Media-independent timeTransmitter

Item	Feature	Status	References	Support
	If MITT not supported, mark N/A.			N/A []
MITT-1	Does the PTP Instance implement the functionality of the AnnounceIntervalSetting state machine in compliance with the requirements of 10.3.17 and Figure 10-19 on each PTP Port?	MITT:M	10.3.17	Yes []
MITT-2	Does the PTP Instance implement the functionality of the PortSyncSyncSend state machine in compliance with the requirements of 10.2.9 and Figure 10-8 on each PTP Port?	MITT:M	10.2.9	Yes []
MITT-3	Does the PTP Instance implement the functionality of the PortAnnounceTransmit state machine in compliance with the requirements of 10.3.16 and Figure 10-18 on each PTP Port?	MITT:M	10.3.16	Yes []
MITT-4	Does the destination MAC address of all Announce messages equal 01:80:C2:00:00:0E?	MITT:M	10.5.3	Yes []
MITT-5	Does the EtherType of all Announce messages equal 88-F7?	MITT:M	10.5.4	Yes []
MITT-6	Do the sequence numbers of Announce messages comply with the requirements of 10.5.7?	MITT:M	10.5.7	Yes []
MITT-7	Does the Announce message header comply with 10.6.2?	MITT:M	10.6.2	Yes []
MITT-8	Does the Announce message body comply with the requirements in 10.6.3.1 and Table 10-11?	MITT:M	10.6.3.1	Yes []
MITT-9	Are all Announce message reserved fields equal to 0?	MITT:M	10.6.1	Yes []
MITT-10	If it is not otherwise specified, is the logAnnounceInterval equal to zero or within the allowed range?	MITT:M	10.7.2.1	Yes []
MITT-11	Does the value of currentUtcOffset comply with the requirements of 8.2.3?	MITT:M	8.2.3	Yes []
MITT-12	Do the values of the leap59, leap61, and currentUtcOffsetValid flags comply with the requirements of 10.3.8?	MITT:M	10.3.8	Yes []
MITT-13	Does this PTP Instance ensure that messages that traverse it or originate from it are not transmitted with VLAN tags in compliance with the requirements of 11.3.3?	MITT:M	11.3.3	Yes []

A.11 Media-independent timeTransmitter (continued)

Item	Feature	Status	References	Support
MITT-14	Is the computation of cumulative rateRatio in accordance with 10.2.8.3?	MITT:M	10.2.8.3	Yes [] N/A []
MITT-15	For transmit of the Announce message, does the PTP Instance support the message requirements?	MITT:M	10.5, 10.6, 10.7	Yes []
MITT-16	Do the TimeTransmitterPorts of this PTP Instance implement the functionality of the SyncIntervalSetting state machine in compliance with the requirements of 10.3.18 and Figure 10-20?	MITT:O	10.3.18, item b) 3) of 5.4.2, item h) of 5.4.1	Yes [] No []

1

A.12 Media-independent performance requirements

Item	Feature	Status	References	Support
MIPERF-1	Does the PTP Instance comply with the performance requirements of B.1?	M	B.1	Yes []
MIPERF-2	Does the PTP Instance comply with the performance requirements of B.2.4?	M	B.2.4	Yes []
MIPERF-3	Does the PTP Instance comply with the performance recommendations of B.2.2?	O	B.2.2	Yes [] No []
MIPERF-4	Does the PTP Instance comply with the performance recommendations of B.2.3?	O	B.2.3	Yes [] No []

2

A.13 Media-dependent, full-duplex point-to-point link

Item	Feature	Status	References	Support
MDFDPP-1	Does this PTP Port implement the functionality of the MDSyncReceiveSM state machine in compliance with the requirements of 11.2.14 and Figure 11-6?	MDFDPP:M	11.2.14	Yes []
MDFDPP-2	Does this PTP Port implement the functionality of the MDSyncSendSM state machine in compliance with the requirements of 11.2.15 and Figure 11-7?	MITT and MDFDPP:M	11.2.15	Yes []
MDFDPP-3	Does this port implement the functionality of the MDPdelayRequest state machine in compliance with the requirements of 11.2.19 and Figure 11-9?	MDFDPP:M	11.2.19	Yes []
MDFDPP-4	Does this port implement the functionality of the MDPdelayResponse state machine in compliance with the requirements of 11.2.20 and Figure 11-10?	MDFDPP:M	11.2.20	Yes []

A.13 Media-dependent, full-duplex point-to-point link (*continued*)

Item	Feature	Status	References	Support
MDFDPP-5	Does this port implement the functionality of the LinkDelayIntervalSetting state machine in compliance with the requirements of 11.2.21 and Figure 11-11?	MDFDPP:O	11.2.21, item l) of 5.5, item h) of 5.4.1	Yes []
MDFDPP-6	Does this PTP Port timestamp Sync messages on ingress with respect to the LocalClock in compliance with 11.3.2.1 and 11.3.9?	MDFDPP:M	11.3.2.1	Yes []
MDFDPP-7	Does this PTP Port timestamp Sync messages on egress with respect to the LocalClock in compliance with the requirements of 11.3.2.1 and 11.3.9?	MITT and MDFDPP:M	11.3.2.1	Yes []
MDFDPP-8	Does this port timestamp Pdelay_Req messages on ingress and egress with respect to the LocalClock in compliance with the requirements of 11.3.2.1 and 11.3.9?	MDFDPP:M	11.3.2.1	Yes []
MDFDPP-9	Does this port timestamp Pdelay_Resp messages on ingress and egress with respect to the LocalClock in compliance with the requirements of 11.3.2.1 and 11.3.9?	MDFDPP:M	11.3.2.1	Yes []
MDFDPP-10	Are all IEEE 802.1AS messages on this port sent without a Q-tag in compliance with the requirements of 11.3.3?	MDFDPP:M	11.3.3	Yes []
MDFDPP-11	Do all media-dependent messages transmitted on this port use a destination MAC address taken from Table 11-3 in compliance with the requirements of 11.3.4 [01-80-C2-00-00-0E]?	MDFDPP:M	11.3.4	Yes []
MDFDPP-12	Do all media-dependent messages transmitted on this port use a source MAC address that is assigned to that port in compliance with the requirements of 11.3.4?	MDFDPP:M	11.3.4	Yes []
MDFDPP-13	Do all media-dependent message tranmited on this port us an EtherType specified in Table 11-4 [88-F7]?	MDFDPP:M	11.3.5	Yes []
MDFDPP-14	Does the header of all the media-dependent messages on this port comply with the requirements of 11.4.2 and Table 10-7?	MDFDPP:M	11.4.2	Yes [] N/A []
MDFDPP-15	Does the body of Sync messages sent on this PTP Port comply with the requirements of 11.4.3, Table 11-8, and Table 11-9?	MDFDPP:M	11.4.3	Yes []
MDFDPP-16	Does the body of Follow_Up messages sent on this PTP Port comply with the requirements of 11.4.4, 6.4.3.3 (lastGmPhaseChange), and Table 11-10?	MDFDPP:M	11.4.4, 6.4.3.3	Yes []
MDFDPP-17	Does the body of Pdelay_Req messages sent on this port comply with the requirements of 11.4.5 and Table 11-12?	MDFDPP:M	11.4.5	Yes []

A.13 Media-dependent, full-duplex point-to-point link (*continued*)

Item	Feature	Status	References	Support
MDFDPP-18	Does the body of Pdelay_Resp messages sent on this port comply with the requirements of 11.4.6 and Table 11-13?	MDFDPP:M	11.4.6	Yes []
MDFDPP-19	Does the body of Pdelay_Resp_Follow_Up messages sent on this port comply with the requirements of 11.4.7 and Table 11-14?	MDFDPP:M	11.4.7	Yes []
MDFDPP-20	Are all reserved fields in media-dependent messages sent on this port set to 0 in compliance with the requirements of 11.4.1?	MDFDPP:M	11.4.1	Yes []
MDFDPP-21	Do the Sync message sequence numbers comply with the requirements of 11.3.8?	MITT and MDFDPP:M	11.3.8	Yes [] N/A []
MDFDPP-22	Do the Pdelay_Req message sequence numbers comply with the requirements of 11.3.8?	MDFDPP:M	11.3.8	Yes []
MDFDPP-23	Does the Pdelay mean request transmission interval comply with the requirements of 11.5.2.2?	MDFDPP:M	11.5.2.2	Yes []
MDFDPP-24	Does the Sync mean transmission interval comply with the requirements of 11.5.2.3?	MDFDPP:M	11.5.2.3	Yes []
MDFDPP-25	Does the full-duplex point-to-point media-dependent layer set the asCapable global variable in the media-independent PortSync entity in compliance with the requirements of 11.2.2?	MDFDPP:M	11.2.2	Yes []
MDFDPP-26	Does the device's use of flow control comply with the requirements of 11.2.3 and 11.2.4?	MDFDPP:M	11.2.3, 11.2.4	Yes []
MDFDPP-27	Does the PTP Instance or CMLDS consider the PTP Port or Link Port, respectively, to not be exchanging Pdelay messages when a valid response is not received in compliance with the requirements of 11.5.3?	MDFDPP:M	11.5.3	Yes []
MDFDPP-28	Does the PTP Instance ignore TLVs, of PTP messages, that it cannot parse and attempt to parse the next TLV, in compliance with the requirements of 11.4.1?	MDFDPP:M	11.4.1	Yes []
MDFDPP-29	Does the time-aware system initialize meanLinkDelayThresh as specified in 11.2.2?	MDFDPP:M	11.2.2	Yes []
MDFDPP-30	Does this port of the time-aware system support asymmetry measurement mode (see Annex G for informative description)?	MDFDPP:O	14.13, 14.18, 10.2.5, 10.2.8, 10.3.12, 10.3.13, 10.3.15, 10.3.16, 11.2.14, 11.2.15, 11.2.20	Yes [] No []

A.13 Media-dependent, full-duplex point-to-point link (*continued*)

Item	Feature	Status	References	Support
MDFDPP-31	Does this PTP Port support one-step receive?	MDFDPP:O	11.2.14	Yes [] No []
MDFDPP-32	Does this PTP Port support one-step transmit?	MDFDPP:O	11.2.15	Yes [] No []
MDFDPP-33	Does this PTP Port implement the functionality of the OneStepTxOperSetting state machine in compliance with the requirements of 11.4, 11.2.16, and Figure 11-8?	MDFDPP:O	11.4, 11.2.16	Yes [] No []
MDFDPP-34	Does this port support propagation delay averaging?	MDFDPP:O	11.2.19.3.4	Yes [] No []
MDFDPP-35	If the time-aware system implements more than one domain, does the time-aware system provide CMLDS?	MDFDPP:M	11.2.17.1	Yes []
MDFDPP-36	If the time-aware system implements only one domain, does the time-aware system provide CMLDS?	MDFDPP:O	11.2.17.2	Yes [] No []
MDFDPP-37	Does this port support two-step capability on receive?	MDFDPP:M	11.2.14, item c) of 5.5	Yes [] No []
MDFDPP-38	Does this port support two-step capability on transmit?	MDFDPP:M	11.2.15, item d) of 5.5	Yes [] No []

1

2

3

4

A.14 Media-dependent IEEE 802.11 link

Item	Feature	Status	References	Support
MDDOT11-1	Does the IEEE 802.11 MAC implement the timeTransmitter port functionality in compliance with the requirements of 12.5.1?	MDDOT11 and MITT:M	item d) of 5.6, 12.5.1	Yes []
MDDOT11-2	Does the IEEE 802.11 MAC implement the timeReceiver port functionality in compliance with the requirements of 12.5.2?	MDDOT11:M	item a), item b), and item d) of 5.6, 12.5.2	Yes []
MDDOT11-3	Does the IEEE 802.11 MAC determine the value of asCapable in compliance with the requirements of 12.4?	MDDOT11:M	12.4	Yes []

A.14 Media-dependent IEEE 802.11 link

Item	Feature	Status	References	Support
MDDOT11-4	Does the IEEE 802.11 MAC determine the value of mean time interval between synchronization messages in compliance with the requirements of 12.8?	MDDOT11 and MITT:M	12.8	Yes []
MDDOT11-5	Does the IEEE 802.11 MAC support the use of the VendorSpecific information element of 12.7 to carry end-to-end link-independent timing information?	MDDOT11:M	12.7	Yes []
MDDOT11-6	Does the IEEE 802.11 MAC implement Fine Timing Measurement as a timeTransmitter port?	MDDOT11-1:O	item c) and item e) of 5.6, 12.5.1	Yes [] No []
MDDOT11-7	Does the IEEE 802.11 MAC implement Fine Timing Measurement as a timeReceiver port?	MDDOT11-2:O	item c) and item e) of 5.6, 12.5.2	Yes [] No []

1

A.15 Media-dependent IEEE 802.3 EPON link

Item	Feature	Status	References	Support
MDEPON-1	Does the TIMESYNC message format comply with the requirements of 13.3 and Table 13-1?	MDEPON:M	13.3	Yes []
MDEPON-2	Does the PTP Instance implement the functionality specified by the requester state machine in compliance with the requirements of 13.8.1 and Figure 13-3?	MDEPON and MITT:M	13.8.1.4	Yes []
MDEPON-3	Does the PTP Instance implement the functionality specified by the responder state machine in compliance with the requirements of 13.8.2 and Figure 13-4?	MDEPON:M	13.8.2.4	Yes []
MDEPON-4	Does the TIMESYNC message transmission interval comply with the requirements of 13.9.1 and 13.9.2?	MDEPON:M	13.9.1, 13.9.2	Yes []
MDEPON-5	Does the implementation of best timeTransmitter selection comply with the requirements of 13.1.3?	MDEPON:M	13.1.3	Yes []
MDEPON-6	Does the determination of the value of asCapable comply with the requirements of 13.4?	MDEPON:M	13.4	Yes []

1

A.16 Media-dependent CSN link

Item	Feature	Status	References	Support
MDCSN-1	Does the PTP Instance implement the functionality of the MDSyncSendSM state machine in compliance with 11.2.15?	MDCSN and MITT:M	11.2.15	Yes []
MDCSN-2	Does the PTP Instance implement the functionality of the MDSyncReceiveSM state machine in compliance with 11.2.14?	MDCSN:M	11.2.14	Yes []
MDCSN-3	Does the PTP Instance calculate path delay in compliance with the requirement of 16.4?	MDCSN:M	16.4.1, 16.4.2, 16.4.3	Yes []
MDCSN-4	Does the PTP Instance propagate synchronized time in compliance with the requirements of 16.5?	MDCSN:M	16.5.2, 16.5.3	Yes []
MDCSN-5	Does the PTP Instance act as Grandmaster PTP Instance in compliance with the requirements of 16.7?	GMCAP and MDCSN:M	16.7	Yes []
MDCSN-6	Does the PTP Instance comply with the performance requirements of 16.8?	GMCAP and MDCSN:M	16.8	Yes []

2

A.17 Media-dependent MoCA link

Item	Feature	Status	References	Support
MDMOCA-1	Does the MoCA MD entity propagate Sync messages in compliance with the requirements of 16.6.2?	MDMOCA:M	16.6.2	Yes []

3

A.18 Media-dependent ITU-T G.hn link

Item	Feature	Status	References	Support
MDGHN-1	Does the GHN MD entity propagate Sync messages in compliance with the requirements of 16.6.3?	MDGHN:M	16.6.3	Yes []

1

A.19 Remote management

Item	Feature	Status	References	Support
	If item RMGT is not supported, mark N/A.			N/A[]
RMGT-1	What management protocol standard(s) or specification(s) are supported?	RMGT:M	item k) 1) of 5.4.2	
RMGT-2	What standard(s) or specification(s) for managed object definitions and encodings are supported for use by the remote management protocol?	RMGT:M	item k) 2) of 5.4.2	
RMGT-3	If the Simple Network Management Protocol (SNMP) is listed in RMGT-2, is the IEEE 8021-AS-MIB module fully supported (per its MODULE-COMPLIANCE)?	RMGT:O	item k) 3) of 5.4.2, Clause 15	Yes [] No [] N/A[]
RMGT-4	If a remote management protocol that supports YANG is listed in RMGT-2, is the YANG data model <code>ieee802-dot1as-gptp</code> of Clause 17 supported?	RMGT:O	item k) 4) of 5.4.2, Clause 17	Yes [] No []
RMGT-5	If a remote management protocol that supports YANG is listed in RMGT-2, and if hot standby is supported, is the YANG data model <code>ieee802-dot1as-hs</code> of Clause 17 supported?	RMGT:O	item k) 5) of 5.4.2, Clause 17, Clause 18	Yes [] No [] N/A[]

2

A.20 Application interfaces

Item	Feature	Status	References	Support
	If item APPL is not supported, mark N/A.			N/A[]
APPL-1	What application interfaces(s) are supported?	APPL:M	item i) of 5.4.2	

3

A.21 External port configuration

Item	Feature	Status	References	Support
	If item EXT is not supported, mark N/A.			NA[]
EXT-1	Does the PTP Instance support the specifications for externalPortConfigurationEnabled value of true?	EXT:M	10.3.1	Yes []
EXT-2	Does the PTP Instance support the PortAnnounceInformationExt state machine?	EXT:M	10.3.14	Yes []
EXT-3	Does the PTP Instance support the PortStateSettingExt state machine?	EXT:M	10.3.15	Yes []

1 Annex B

2 (normative)

3 Performance requirements

4 B.1 LocalClock requirements

5 B.1.1 Frequency accuracy

6 The fractional frequency offset of the LocalClock relative to the TAI frequency (see ISO 80000-3:2006 and
7 Annex C) shall be within ± 100 ppm.

8 B.1.2 Time measurement granularity

9 The granularity with which the LocalClock measures time shall be less than or equal to $40/(1-0.0001)$ ns.

10 B.1.3 Noise generation

11 B.1.3.1 Jitter generation

12 The jitter generation of the free-running LocalClock shall not exceed 2 ns peak-to-peak, when measured
13 over a 60 s measurement interval using a band-pass filter that consists of the following low-pass and high-
14 pass filters:

- 15 a) High-pass filter: first-order characteristic (i.e., 0 dB gain peaking), 20 dB/decade roll-off, and 3 dB
16 bandwidth (i.e., corner frequency) of 10 Hz
- 17 b) Low-pass filter: maximally-flat (i.e., Butterworth) characteristic, 60 dB/decade roll-off, and 3 dB
18 bandwidth equal to the Nyquist rate of the LocalClock entity (i.e., one-half the nominal frequency of
19 the LocalClock entity)

20 B.1.3.2 Wander generation

21 Wander generation is specified using the Time Deviation (TDEV) parameter. The corresponding values of
22 the Allan Deviation (ADEV) and PTP Deviation (PTPDEV) are given for information; the former is also
23 useful in describing the wander generation of clocks and oscillators, and the latter is related to the
24 offsetScaledLogVariance attribute (see 8.6.2.4). Information on ADEV and TDEV is contained in
25 ITU-T G.810 [B21] and IEEE Std 1139™-1999 [B9]. Information on Allan Deviation and PTP Variance
26 (PTP Deviation is the square root of PTP Variance) is contained in 7.6.3 of IEEE Std 1588-2019.

1 TDEV, denoted $\sigma_x(\tau)$, is estimated from a set of measurements, as shown in Equation (B-1).

$$2 \quad \sigma_x(\tau) = \sqrt{\frac{1}{6n^2(N-3n+1)} \sum_{j=1}^{N-3n+1} \left[\sum_{i=j}^{n+j-1} (x_{i+2n} - 2x_{i+n} + x_i) \right]^2}, n = 1, 2, \dots \left\lfloor \frac{N}{3} \right\rfloor \quad (\text{B-1})$$

3 where

- 4 τ is $n\tau_0$ = observation interval
- 5 τ_0 is the sampling interval
- 6 N is the total number of samples [$(N-1)\tau_0$ = measurement interval]
- 7 $\lfloor y \rfloor$ denotes the floor function, i.e., the greatest integer less than or equal to y
- 8 x_i is the measured phase (time) error at the i^{th} sampling time [the units of x_i and $\sigma_x(\tau)$ are the same]

9 ADEV, denoted $\sigma_y(\tau)$, is estimated from a set of measurements, as shown in Equation (B-2).

$$10 \quad \sigma_y(\tau) = \sqrt{\frac{1}{2n^2\tau_0^2(N-2n)} \sum_{i=1}^{N-2n} (x_{i+2n} - 2x_{i+n} + x_i)^2}, n = 1, 2, \dots \left\lfloor \frac{N-1}{2} \right\rfloor \quad (\text{B-2})$$

11 where the notation is the same as defined above for TDEV.

12 PTPDEV, denoted $\sigma_{PTP}(\tau)$, is estimated from a set of measurements, as shown in Equation (B-3).

$$13 \quad \sigma_{PTP}(\tau) = \sqrt{\frac{1}{6(N-2n)} \sum_{i=1}^{N-2n} (x_{i+2n} - 2x_{i+n} + x_i)^2}, n = 1, 2, \dots \left\lfloor \frac{N-1}{2} \right\rfloor \quad (\text{B-3})$$

14 where the notation is the same as defined above for TDEV.

15 TDEV, ADEV, and PTPDEV are second-order statistics on the phase error. All three statistics are functions
 16 of second differences of the phase error. As a result, these statistics are not affected by a constant frequency
 17 offset. This behavior is desired, because these statistics are used here to constrain noise generation.

1 TDEV for the LocalClock entity shall not exceed the mask of Table B-1 and Figure B-1, when measured
 2 using:

- 3 a) A measurement interval that is at least 120 s (i.e., at least 12 times the longest observation interval),
- 4 b) A low-pass filter with 3 dB bandwidth of 10 Hz, first-order characteristic, and 20 dB/decade roll-off,
 5 and
- 6 c) A sampling interval τ_0 that does not exceed 1/30 s.

Table B-1—Wander generation TDEV requirement for LocalClock entity

TDEV limit	Observation interval τ
No requirement	$\tau < 0.05$ s
5.0τ ns	$0.05 \leq \tau \leq 10$ s
No requirement	$\tau > 10$ s

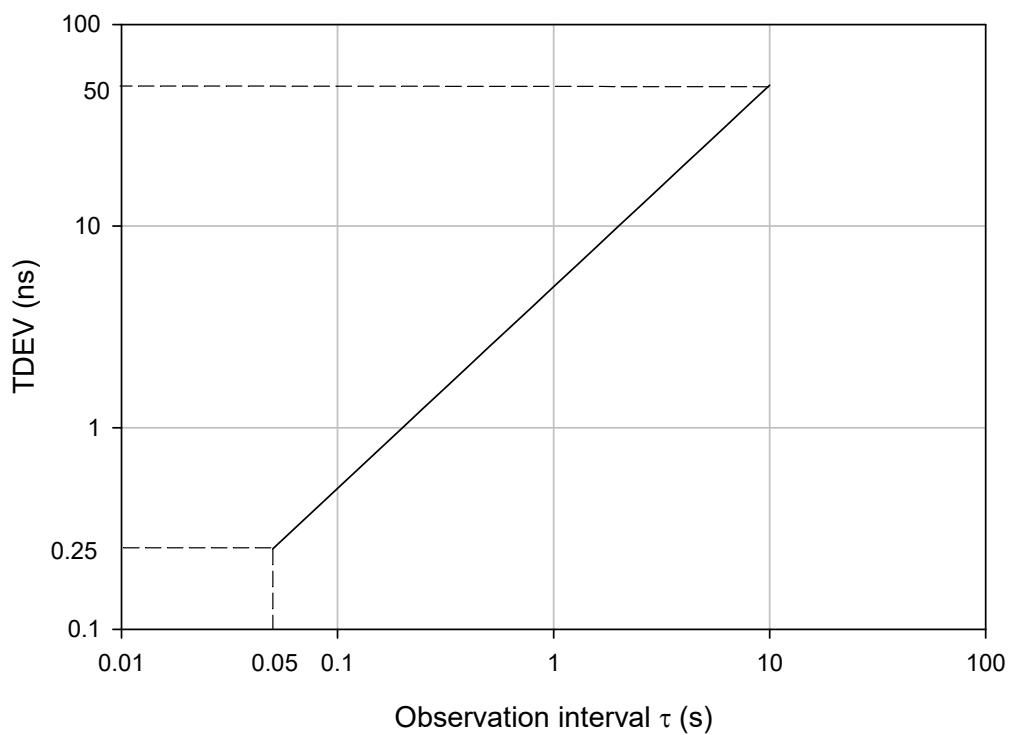


Figure B-1—Wander generation (TDEV) requirement for LocalClock entity

1 The ADEV limit that corresponds to the TDEV requirement of Table B-1 and Figure B-1 is shown in
2 Table B-2 and Figure B-2, respectively.

Table B-2—ADEV limit corresponding to wander generation requirement of Table B-1

ADEV limit	Observation interval τ
No requirement	$\tau < 0.05$ s
1.054×10^{-8}	$0.05 \leq \tau \leq 10$ s
No requirement	$\tau > 10$ s

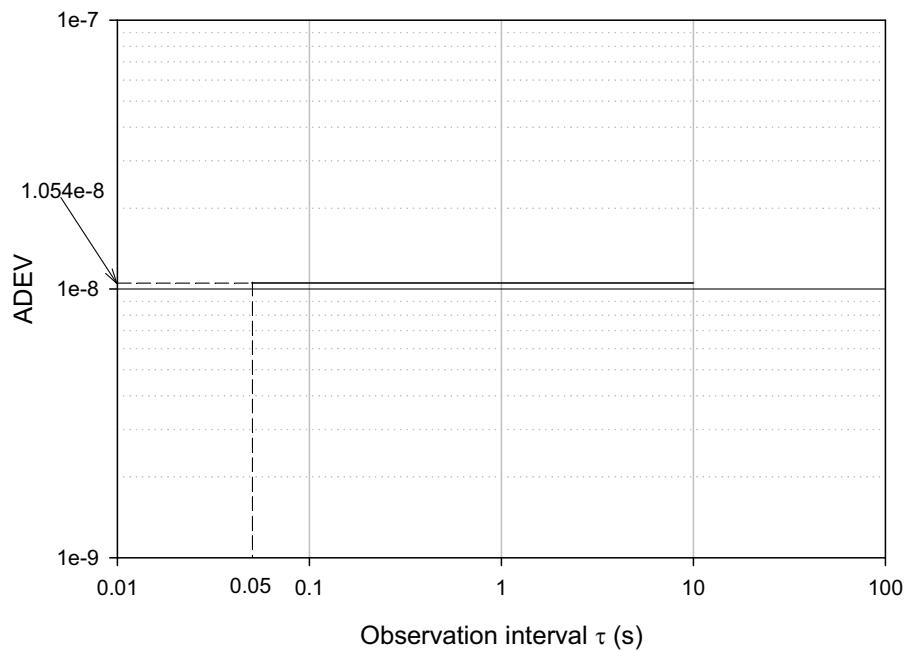


Figure B-2—ADEV limit corresponding to wander generation requirement of Figure B-1

1 The PTPDEV limit that corresponds to the TDEV requirement of Table B-1 and Figure B-1 is shown in
2 Table B-3 and Figure B-3, respectively.

Table B-3—PTPDEV limit corresponding to wander generation requirement of Table B-1

PTPDEV limit	Observation interval τ
No requirement	$\tau < 0.05 \text{ s}$
$6.08\tau \text{ ns}$	$0.05 \leq \tau \leq 10 \text{ s}$
No requirement	$\tau > 10 \text{ s}$

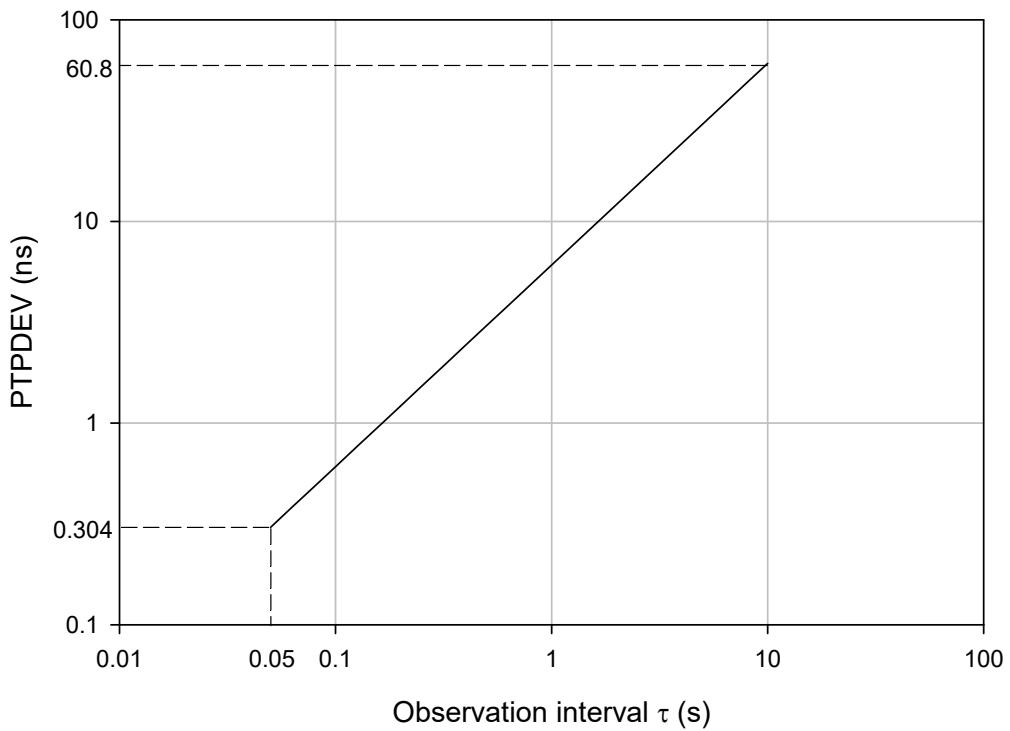


Figure B-3—PTPDEV limit corresponding to wander generation requirement of Figure B-1

1 **B.2 PTP Instance requirements**

2 **B.2.1 General**

3 In order to achieve the accuracy goals, certain constraints are placed on the responsiveness and accuracy of
4 PTP Instances.

5 **B.2.2 Residence time**

6 The residence time (see 3.27) of a PTP Instance, measured relative to the TAI second (see 8.2), should be
7 less than or equal to 10 ms.

8 Any error in the measured frequency offset relative to the Grandmaster Clock (i.e., error in accumulated
9 rateRatio) results in an error in the transported synchronized time that is equal to the frequency offset error
10 multiplied by the residence time (see 7.3.3 and 11.1.3).

11 **B.2.3 Pdelay turnaround time**

12 The pdelay turnaround time is the duration of the interval between the receipt of a Pdelay_Req message by a
13 port of a time-aware system, and the sending of the corresponding Pdelay_Resp message.

14 The pdelay turnaround time of a time-aware system, measured relative to the TAI second (see 8.2), should
15 be less than or equal to 10 ms.

16 A nonzero pdelay turnaround time and any error in the measured frequency offset between the peer delay
17 initiator and peer delay responder (i.e., error in *nrrPdelay*) results in an error in the measured mean
18 propagation delay. This in turn results in an error in the transported synchronized time (see 11.1.2 for more
19 details).

20 NOTE—While a larger value of pdelay turnaround time can result in worse time-synchronization performance, the peer
21 delay protocol will still operate as long as the peer delay initiator receives Pdelay_Resp and Pdelay_Resp_Follow_Up
22 within a time interval since sending Pdelay_Req that is less than the current Pdelay_Req message transmission interval
23 (see 11.2.19.4 and 11.5.2.2).

24 **B.2.4 Measurement of rate ratio**

25 This standard requires the measurement of rate ratio or, equivalently, frequency offset, in several subclauses
26 (see 10.2.10, 10.2.11, 11.2.19, 12.5.2, 16.4.2, and 16.4.3.2). The error inherent in any scheme used to
27 measure rate ratio shall not exceed ± 0.1 ppm.

28 NOTE—This requirement is consistent with a rate ratio measurement made by measuring the default Pdelay_Req
29 message transmission interval (the nominal interval duration is 1 s; see 11.5.2.2) relative to the clocks whose rate ratio is
30 desired, assuming the clocks meet the time measurement granularity requirement of B.1.2 (i.e., no worse than 40 ns).

1 B.3 End-to-end time-synchronization performance

2 Assuming that the requirements of this standard and of standards referenced for each medium are met, any
 3 two PTP Instances separated by six or fewer PTP Instances (i.e., seven or fewer hops) will be synchronized
 4 to within 1 microsecond peak-to-peak of each other during steady-state operation (i.e., each PTP Instance
 5 receives time-synchronization information every sync interval).

6 B.4 End-to-end jitter and wander performance

7 The requirements of this standard and standards referenced by this standard ensure that the synchronized
 8 time at a PTP Instance that is separated from the Grandmaster PTP Instance by six or fewer PTP Instances
 9 (i.e., seven or fewer hops) will, when filtered by a reference endpoint filter with rolloff of 20 dB/decade,
 10 gain peaking that does not exceed 0.1 dB, and bandwidth that does not exceed the value given in each entry
 11 of Table B-4, have maximum time interval error (MTIE) (see ITU-T G.810 [B21]) that does not exceed the
 12 MTIE for that entry of Table B-4, and jitter that does not exceed the peak-to-peak jitter of Table B-4 when
 13 measured through the corresponding high-pass jitter measurement filter given in Table B-4.

14 NOTE—For example, the endpoint filter can be of the following form:

$$15 \quad y_k = a_1y_{k-1} + a_2y_{k-2} + \dots + a_ny_{k-n} + b_0x_k + b_1x_{k-1} + \dots + b_nx_{k-n}$$

16 where the x_k are the unfiltered synchronized time values, the y_k are the filtered synchronized time values, and the a_k and
 17 b_k are filter coefficients. The a_k and b_k are chosen such that the filter has desired bandwidth and gain peaking that does
 18 not exceed 0.1 dB. The preceding equation is a general infinite impulse response (IIR) digital filter. Simplified forms,
 19 e.g., a second order IIR filter obtained by setting $n = 2$, or a finite impulse response (FIR) filter obtained by setting the a_k
 20 to zero are possible.

**Table B-4—Maximum endpoint filter bandwidths needed to meet respective
MTIE masks and peak-to-peak jitter limits**

Endpoint filter maximum bandwidth (Hz)	Corresponding MTIE mask of Figure B-4 that is not exceeded	Corresponding jitter high-pass measurement filter (Hz)	Corresponding peak-to-peak jitter that is not exceeded (ns)
10	Mask 2 (Figure B-4, Table B-6)	8000	10.2
1	Mask 1 (Figure B-4, Table B-5)	200	11.1

21 Mask 1 of Table B-4 and Figure B-4 corresponds to consumer digital audio applications. Mask 2 of
 22 Table B-4 and Figure B-4 corresponds to professional digital audio applications. Mask 1 is derived from the
 23 requirements given in IEC 60958-3 [B6]. Mask 2 is derived from the requirements given in
 24 IEC 60958-4 [B7], AES3-2009 [B1], and AES11-2009 [B2]. Garner describes the methodology for
 25 deriving MTIE from the various jitter and synchronization requirements and presents the detailed
 26 derivations for masks 1 and 2.

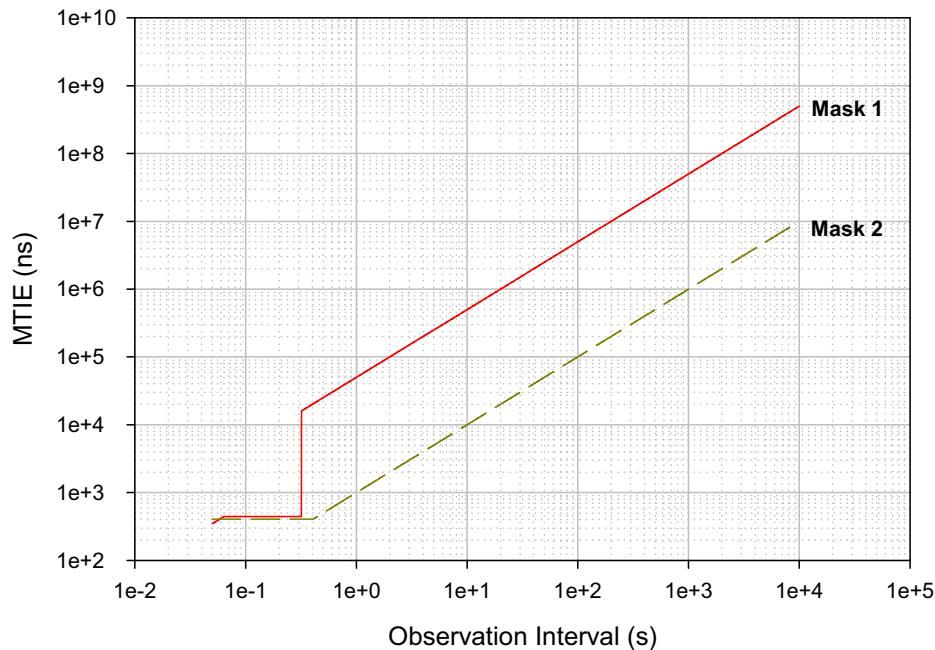


Figure B-4—MTIE masks met for maximum endpoint filter bandwidths of Table B-4

Table B-5—Breakpoints for Mask 1

Observation interval S (s)	MTIE (ns)
$0.05 \leq S < 0.0637$	6954.8S
$0.0637 \leq S < 0.3183$	443
$0.3183 \leq S \leq 10000$	50000S

Table B-6—Breakpoints for Mask 2

Observation interval S (s)	MTIE (ns)
$0.05 \leq S < 0.4069$	407
$0.4069 \leq S < 10000$	1000S

¹ Annex C

² (informative)

³ Timescales and epochs

⁴ C.1 Overview

⁵ A more detailed discussion of many of the topics in this annex can be found in Allan et al. [B3].

⁶ For historical reasons, time is specified in a variety of ways as listed in Table C-1. GPS, PTP, and TAI times
⁷ are based on values yielded by atomic clocks and advance on each second. NTP and UTC times are similar,
⁸ but are occasionally adjusted by one leap second, to account for differences between the atomic clocks and
⁹ the rotation time of the earth.

Table C-1—Timescale parameters

Parameter	Timescale				
	GPS	PTP	TAI	NTP	UTC
Approximate epoch ^a	1980-01-06 1999-08-22 2019-04-07	1970-01-01	No epoch defined	1900-01-01	No epoch defined
Representation	weeks.seconds	seconds	YYYY-MM-DD hh:mm:ss	seconds	YYYY-MM-DD hh:mm:ss
Rollover (years)	19.7	≈8 900 000	10 000	≈136	10 000
NOTE 1—After 1972-01-01 00:00:00 TAI, TAI and UTC differ by only integer seconds.					
NOTE 2—The following represent the same instant in time: 1958-01-01 00:00:00 TAI and 1958-01-01 00:00:00 UTC					

^a Each approximate epoch occurs at 00:00:00 on the respective date.

GPS global positioning satellite

NTP network time protocol

PTP IEEE 1588 precision time protocol

TAI International Atomic Time (from the French term *Temps Atomique International*)

UTC Coordinated Universal Time [The abbreviation is a compromise between the English phrase *coordinated universal time* (CUT) and the French phrase *temps universel coordonné* (TUC).]

¹⁰ C.2 TAI and UTC

¹¹ TAI and UTC are international standards for time based on the SI second (see Allan et al. [B3], IAU [B27],
¹² and Petit and Luzum [B26]). The SI second is the duration of 9 192 631 770 periods of the
¹³ radiation corresponding to the transition between the two hyperfine levels of the ground state of the
¹⁴ cesium 133 atom.²⁶ TAI is implemented by a suite of atomic clocks and forms the timekeeping basis for
¹⁵ other timescales in common use. The rate at which UTC time advances is normally identical to the rate of

²⁶ See ITU-R TF.460-6 [B20], Jekeli [B22], Service de la Rotation Terrestre [B28], SI [B14], IAU [B27], and Petit and Luzum [B26] for more details on UTC, TAI, and the SI second.

1 TAI. An exception is an occasion when UTC is modified by adding or subtracting exactly one whole leap
 2 second. The TAI frequency is the frequency of a signal whose period is the TAI second.

3 The TAI and UTC timescales were introduced as of 1958-01-01, and the times 1958-01-01 00:00:00 TAI
 4 and 1958-01-01 00:00:00 UTC represent the same instant in time. However, this instant in time is not an
 5 epoch for TAI and UTC. An epoch, in the sense of the origin of a timescale (see 8.2.2), is not defined for
 6 either TAI or UTC. TAI and UTC are expressed in the form YYYY-MM-DD hh:mm:ss, rather than as
 7 elapsed time since an epoch; here, YYYY-MM-DD denotes the date, and hh:mm:ss denotes the time in
 8 each day.

9 Prior to 1972-01-01, corrections to the offset between UTC and TAI were made by applying fractional-
 10 second corrections to UTC and corrections to the rate at which UTC advanced relative to the rate at which
 11 TAI advanced. After 1972-01-01, leap-second corrections are applied to UTC by inserting or deleting
 12 second(s) at the end of the last minute of preferably the last day of June or December. Also after 1972-01-01,
 13 UTC and TAI advance at the same rate. As of 2006-01-01, TAI and UTC times differed by +33 s (i.e.,
 14 TAI time minus UTC time equals +33 s for 2006-01-01).

15 In computer networks, the common POSIX-based time conversion algorithms are typically used to produce
 16 the correct ISO 8601:2004 [B15] printed representations for both TAI and UTC.

17 The PTP epoch is set such that a direct application of the POSIX algorithm to a PTP timescale timestamp
 18 converts the PTP timestamp to the ISO 8601:2004 [B15] printed representation of TAI. PTP also distributes
 19 the current offset between TAI and UTC in the currentUtcOffset field of Announce messages. Except during
 20 leap seconds, subtracting currentUtcOffset from a PTP timestamp and then applying the POSIX algorithm
 21 result in the ISO 8601:2004 printed representation of UTC. Conversely, except during leap seconds,
 22 applying the inverse POSIX algorithm and adding currentUtcOffset convert from the ISO 8601:2004 printed
 23 form of UTC to the form required to generate a PTP timestamp.

24 For example, at 0h 2 January 1972 TAI, the value of PTP Instance Time was 63 158 400. At this time
 25 currentUtcOffset was 10. The POSIX algorithm applied to the value (63 158 400 – 10) gives a value of
 26 1972-01-01 23:59:50 (10 s before 0h 2 January 1972 UTC). The value of PTP Instance Time on
 27 0h 2 January 1972 TAI is computed by observing that PTP Instance Time = 0 on 0h 1 January 1970 TAI, i.e.,
 28 Modified Julian Day (MJD) 40587 (see Petit and Luzum [B26]). On 0h 2 January 1972 TAI, MJD = 41 318.
 29 Thus, PTP Instance Time on 0h 2 January 1972 TAI is $0 + 86\ 400 \times (41\ 318 - 40\ 587)$. Note that if this
 30 calculation were done for a day in which a leap second occurred, a more complex algorithm would be
 31 required to ensure that, for the duration of the leap second, the ISO 8601:2004 [B15] print form seconds
 32 value was 60 for a positive leap second.

33 International standards specify that if a correction to UTC relative to TAI is required, the leap second occurs
 34 at the last second of the UTC day, preferably at the end of June 30 or December 31. For a negative leap
 35 second, the last minute of the designated day has only 59 seconds. Negative leap seconds have never
 36 occurred and are unlikely to occur in the future. For a positive leap second, the last minute of the designated
 37 day has 61 seconds.

38 Although a negative leap second is unlikely to occur, if such a correction becomes necessary, the
 39 ISO 8601:2004 [B15] printed representation would appear as follows for a hypothetical negative leap
 40 second on 30 June 1972 UTC:

41 1972-06-30 23:59:57, 1972-06-30 23:59:58, 1972-07-01 00:00:00

42 For the positive leap second that actually occurred on 30 June 1972 UTC, the ISO 8601:2004 [B15] printed
 43 representation appeared as follows:

44 1972-06-30 23:59:59, 1972-06-30 23:59:60, 1972-07-01 00:00:00

- 1 Note the 23:59:60 notation to indicate the added second.
- 2 The update semantics for leap second updates in PTP are discussed in B.2.2 of IEEE Std 1588-2019.

3 C.3 NTP and GPS

4 Two standard time sources of particular interest in implementing PTP Instances: NTP and GPS. Both NTP
5 and GPS systems are expected to provide time references for calibration of the grandmaster-supplied PTP
6 time.

7 NTP represents seconds as a 32-bit unsigned integer that rolls-over every 2^{32} s \approx 136 year, with the first such
8 rollover occurring in the year 2036. The precision of NTP systems is usually in the millisecond range.

9 NTP is a widely used protocol for synchronizing computer systems. NTP is based on sets of servers, to
10 which NTP clients synchronize. These servers themselves are synchronized to time servers that are traceable
11 to international standards.

12 NTP version 4 provides the current UTC time and warning flags indicating that a leap second will be
13 inserted at the end of the current UTC day. The NTP clock effectively stops for one second when the leap
14 second is inserted.

15 GPS time comes from a global positioning satellite system, GPS, maintained by the U.S. Department of
16 Defense. The precision of GPS system is usually in the 10 ns to 100 ns range. GPS system transmissions
17 represent the time as {weeks, secondsInWeek}, the number of weeks since the GPS epoch and the number of
18 seconds since the beginning of the current week.

19 GPS provides a leap seconds offset and warning flags marking the introduction of a leap second correction
20 (see IS-GPS-200J [B19]). UTC and TAI times can be computed solely based the information contained in
21 the GPS transmissions.

22 GPS timing receivers generally manage the epoch transitions (1024-week rollovers), providing the correct
23 time (YYYY-MM-DD hh:mm:ss) in TAI and/or UTC timescales, and often also local time; in addition to
24 providing the raw GPS week, second of week, and leap-second information.

25 C.4 Timescale conversions

26 Previously discussed representations of time can be readily converted to/from PTP *time* based on a constant
27 offset and the distributed *utcOffset* value, as specified in Table C-2. Within Table C-2, all variables represent
28 integers; “/” and “%” represent a integer divide and remainder operation, respectively.

29 NOTE—Some of the conversions in Table C-2 will not give the correct result when the current second is a leap second.

Table C-2—Timescale conversions

ta		PTP value tb
Name	Format	
GPS	weeks:seconds	$tb = ta.seconds + 315\,964\,819 + (gpsRollovers * 1024 + ta.weeks) * (7 * DAYSECS);$
		$ta.weeks = (tb - 315\,964\,819) / (7 * DAYSECS) - gpsRollovers * 1024;$ $ta.seconds = (tb - 315\,964\,819) \% (7 * DAYSECS);$
TAI	date{YYYY,MM,DD}:time{hh,mm,ss}	$tb = \text{DateToDays}("1970-01-01", ta.date) * DAYSECS + ((ta.time.hh * 24) + ta.time.mm) * 60 + ta.time.ss;$
		$secs = tb \% DAYSECS;$ $ta.date = \text{DaysToDate}("1970-01-01", tb / DAYSECS);$ $ta.time.hh = secs / 3600;$ $ta.time.mm = (secs \% 3600) / 60;$ $ta.time.ss = (secs \% 60);$
NTP	seconds	$tb = (ta + utcOffset) - 2208988800;$
		$ta = (tb - utcOffset) + 2208988800;$
UTC	date{YYYY,MM,DD}:time{hh,mm,ss}	$tb = \text{DateToDays}("1970-01-01", ta.date) * DAYSECS + ((ta.time.hh * 24) + ta.time.mm) * 60 + ta.time.ss + utcOffset;$
		$tc = tb - utcOffset;$ $secs = tc \% DAYSECS;$ $ta.date = \text{DaysToDate}("1970-01-01", tc / DAYSECS);$ $ta.time.hh = secs / 3600;$ $ta.time.mm = (secs \% 3600) / 60;$ $ta.time.ss = (secs \% 60);$

gpsRollovers Currently equals 1; changed from 0 to 1 between 1999-08-15 and 1999-08-22

DAYSECS The number of seconds within a day: $(60 \times 60 \times 24)$

utcOffset The difference TAI – UTC, in seconds [i.e., currentUtcOffset (see 8.2.3)]

DateToDays For arguments DateToDays(*past, present*), returns days between *past* and *present* dates

DaysToDate For arguments DaysToDate(*past, days*), returns the current date, *days* after the *past* date

1 C.5 Time zones and GMT

2 The term *Greenwich Mean Time* (GMT) once referred to mean solar time at the Royal Observatory in
 3 Greenwich, England. GMT now commonly refers to the timescale UTC; or the UK winter time zone
 4 (Western European Time, WET). Such GMT references are, strictly speaking, incorrect but nevertheless
 5 quite common. The following representations correspond to the same instant of time:

6	18:07:00 (GMT), commonplace usage	13:07:00 (Eastern Standard Time, EST)
7	18:07:00 (UTC)	01:07 PM (Eastern Standard Time, EST)
8	18:07:00 (Western European Time, WET)	10:07:00 (Pacific Standard Time, PST)
9	06:07 PM (Western European Time, WET)	10:07 AM (Pacific Standard Time, PST)

¹ **Annex D**

²

³ **Reserved for future use**

⁴

¹ Annex E

²

³ Reserved for future use

⁴ (For information on media-dependent layer specification for CSN, see Clause 16.)

¹ Annex F

² (informative)

³ F.1 PTP profile included in this standardGeneral

⁴ The specification in this standard of synchronized time transport over a full-duplex point-to-point link
⁵ includes a PTP profile. The information contained in a PTP profile is described in 20.3 of IEEE Std 1588-
⁶ 2019. This annex summarizes the PTP profile for transport of timing over full-duplex point-to-point links.
⁷ This PTP profile is also used in the transport of timing over CSN when a CSN clock reference is not present
⁸ (see Clause 16). This PTP profile is not used in the transport of timing over IEEE 802.11 links and IEEE
⁹ 802.3 EPON links; both these transports use native timing mechanisms to assist in the synchronized time
¹⁰ transport.

¹¹ F.2 Identification

¹² The identification values for this PTP profile (see 20.3.3 of IEEE Std 1588-2019) are as follows:

¹³ PTP Profile:
¹⁴ IEEE 802.1AS PTP profile for transport of timing
¹⁵ Profile Name: IEEE 802.1AS PTP profile
¹⁶ profileNumber: 0
¹⁷ primaryVersion: 2
¹⁸ revisionNumber: 0
¹⁹ profileIdentifier: 00-80-C2-00-02-00

²⁰ NOTE—In the above profileIdentifier (see 20.3.3 of IEEE Std 1588-2019):

- ²¹ a) 00-80-C2 (first three octets) is the OUI owned by the IEEE 802.1 Working Group (it identifies the organization
²² that specifies the PTP profile);
²³ b) 00 (fourth octet) is a number that identifies the PTP profile, among all profiles specified by the organization that
²⁴ owns the OUI (or, in general, OUI or CID) of item a);
²⁵ c) 02 (fifth octet) is the primary version of this PTP profile; and
²⁶ d) 00 (sixth octet) is the revisionNumber of this PTP profile.

²⁷ This profile is specified by the IEEE 802.1 Working Group of the IEEE 802 LAN/MAN Standards
²⁸ Committee.

²⁹ A copy can be obtained by ordering IEEE Std 802.1AS-2020 from the IEEE Standards Organization.²⁷

³⁰ This PTP profile is a revision of the PTP profile included in IEEE Std 802.1AS-2011, i.e., major changes
³¹ have been made to the profile relative to Version 1.0. Therefore, the primaryVersion is changed to 2, with
³² revisionNumber 0.

²⁷ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org>).

1 F.3 PTP attribute values

2 The ranges and default values for time-aware system and PTP Instance attributes covered by this profile are
 3 as follows:

- 4 a) A domain whose **domainNumber** is 0 is present. A domain whose **domainNumber** is in the range 1
 5 through 127 can be present (see 8.1).
- 6 b) The default logAnnounceInterval (see 10.7.2.2) is 0. The value 127 is supported.
- 7 c) The default logSyncInterval (see 11.5.2.3) is -3. The value 127 is supported.
- 8 d) The default logPdelayReqInterval (see 11.5.2.2) is 0. The value 127 is supported.
- 9 e) The default announceReceiptTimeout (see 10.7.3.2) is 3.
- 10 f) The default values of priority1, for different media, are specified in 8.6.2.1, Table 8-1. The value of
 11 priority1 for a PTP Instance that is not grandmaster-capable is 255.
- 12 g) The default value of priority2 is 248 (see 8.6.2.5).

13 The default observation interval for offsetScaledLogVariance is equal to the default sync interval, i.e., 0.125
 14 s (see 8.6.2.4).

15 F.4 PTP options

- 16 a) The BTCA of this standard is the default BTCA according to the specifications of 9.3 of
 17 IEEE Std 1588-2019.
- 18 b) The following options of 17.7 of IEEE Std 1588-2019 are invoked:
 - 19 1) The FAULTY state is not used.
 - 20 2) The UNCALIBRATED state is not used.
 - 21 3) The LISTENING state is not used.
 - 22 4) The PRE_TIME_TRANSMITTER state, and PRE_TIME_TRANSMITTER qualification are
 23 not used.
 - 24 5) The foreign timeTransmitter feature is not used.
- 25 c) The management mechanism is the mechanism specified in Clause 14, Clause 15, and Clause 17.
- 26 d) The path delay mechanism is the peer-to-peer delay mechanism (see Clause 11).
- 27 e) The transport mechanism is full-duplex and point-to-point, and it uses attribute values described in
 28 Annex E of IEEE Std 1588-2019 for IEEE 802.3 Ethernet. Specifically, the address, EtherType, and
 29 subtype are specified in 11.3.4, 11.3.5, and 11.3.6.
- 30 f) A PTP Instance that contains one PortSync and one MD entity is an Ordinary Clock. A PTP Instance
 31 that contains more than one PortSync and more than one MD entity is a Boundary Clock.
- 32 g) Each port of a time-aware system measures the frequency offset of its neighbor, at the other end of
 33 the attached link, relative to itself (see Clause 11). The frequency offset, relative to the Grandmaster
 34 Clock, is accumulated in a standard organization TLV that is attached to the Follow_Ups message if
 35 the PTP Port is two-step and the Sync message if the PTP Port is one-step (see 11.4.4.3). The
 36 standard organization TLV also carries information on Grandmaster Clock traceability and phase
 37 and frequency change due to the most recent Grandmaster PTP Instance change. The physical
 38 adjustment of the frequency of the LocalClock entity (i.e., physical syntonization) is allowed but not
 39 required.

40 NOTE 1—This feature is similar to the cumulative frequency transfer method specified in 16.10 of IEEE Std 1588-2019;
 41 however, it is not the same feature and uses a different TLV from the one used in the IEEE 1588 feature. This feature
 42 existed in the 2011 edition of the present standard and is retained for backward compatibility and also because the TLV
 43 of this standard carries additional information that is not carried in the IEEE 1588 feature [see item g) in this subclause].

- 1 h) A standard organization TLV is defined to allow a PTP Port to signal to its neighbor PTP Port that it
2 is capable of invoking gPTP (see 10.4 and 10.6.4.4).
- 3 i) The path trace feature of 16.2 of IEEE Std 1588-2019 is used (see 10.6.3.2.8).
- 4 j) A standard organization TLV is defined that allows a port of a time-aware system to request that its
5 neighbor slow down or speed up the rate at which it sends Sync/Follow_U, peer delay, and/or
6 Announce messages (see 10.6.4.3).
- 7 k) The acceptable timeTransmitter table feature of IEEE Std 1588-2019 is used with IEEE 802.3 EPON
8 links to ensure that the OLT is timeTransmitter and ONUs are timeReceivers.

9 NOTE 2—This feature is used with EPON links and therefore could be considered to be outside the PTP profile
10 (because EPON links are not part of the PTP profile). It is included here because it is one of the optional features
11 described in IEEE Std 1588-2019.

- 12 l) The profile isolation feature of IEEE Std 1588-2019 is not explicitly used; however, the PTP profile
13 specified in this standard is isolated from other PTP profiles because it uses sdoId 0x100 (see 8.1).
- 14 m) A time-aware system can have more than one gPTP domain (see 8.1).
- 15 n) The Common Mean Link Delay Service specified in 16.6 of IEEE Std 1588-2019 is required if more
16 than one domain is implemented and is optional if one domain (with domainNumber 0) is
17 implemented (see 11.2.17).
- 18 o) The security mechanism of 16.14 of IEEE Std 1588-2019 and security annex (Annex P) of
19 IEEE Std 1588-2019 are not used.
- 20 p) The external port configuration feature of 17.6 of IEEE Std 1588-2019 is optional in the present
21 standard.
- 22 q) Except for items g) through p) in this subclause, the optional features of Clause 16 and Clause 17
23 and the optional annexes of IEEE Std 1588-2019 are not used.

24 F.5 LocalClock and PTP Instance performance requirements

25 The LocalClock performance requirements are as specified in B.1. The PTP Instance performance
26 requirements are as specified in B.2.

27

¹ Annex G

² (informative)

³ The asymmetry compensation measurement procedure based on ⁴ line-swapping

⁵ G.1 Introduction

⁶ This annex describes the asymmetry compensation measurement procedure based on the line-swapping
⁷ method. The entire procedure is controlled by the Network Management System (NMS), except that the
⁸ line-swapping is manually operated. This annex is intended to address the delay asymmetry due to length
⁹ differences between transmit and receive fibers for long fiber runs.

¹⁰ NOTE—When a port is put into asymmetry measurement mode, it is put into this mode on all domains. The per-port
¹¹ global variable asymmetryMeasurement mode is common to, and accessible by, all domains (see 10.2.5.2).

¹² G.2 Pre-conditions for measurement

¹³ The following pre-conditions should be met to both improve the accuracy of the measurement and make the
¹⁴ measurement procedure more convenient:

- ¹⁵ a) The measurement environment, including the testing nodes (i.e., the time-aware systems at the
¹⁶ endpoint of the link whose asymmetry is being compensated) and related nodes (i.e., nodes in the
¹⁷ paths between the testing nodes and the Grandmaster PTP Instance), should enable gPTP and the
¹⁸ BTCA so that the test can be made for each link without changing the configuration.
- ¹⁹ b) The testing nodes should have redundant paths for synchronization so that they remain synchronized
²⁰ to the Grandmaster Clock during the asymmetry compensation measurement.

²¹ G.3 Measurement procedure

²² The assumed measurement environment is shown in Figure G-1. Before the measurement starts, every node
²³ has enabled syntonization [i.e., by measuring `nrrPdelay` (see 11.2.13.13) for each port of each time-aware
²⁴ system and accumulating the respective values over the synchronization spanning tree paths to obtain
²⁵ `rateRatio` (see 10.2.8.1.4) relative to the Grandmaster Clock^{28]} and time synchronization. The testing link is
²⁶ between Port2 on node ACC1 and Port1 on node ACC2.

²⁷

²⁸

²⁹

³⁰

²⁸ This standard neither requires nor prohibits syntonization (see 3.31) at the physical layer, e.g., using Synchronous Ethernet as the physical layer. If frequency is syntonized at the physical layer, the respective `nrrPdelay` and `rateRatio` values are expected to be close to 1.

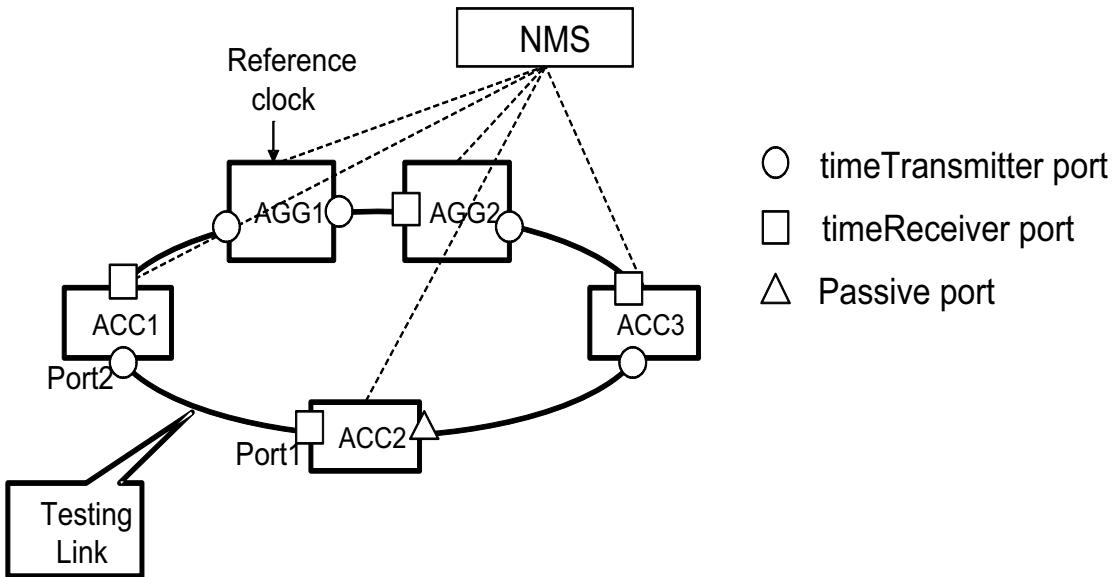


Figure G-1—Asymmetry compensation measurement procedure

1 The criteria of 11.2.17 for determining whether the peer-to-peer delay mechanism is the **transport-specific**
 2 **peer-to-peer delay mechanism** or CMLDS apply here.

3 The measurement procedure is as follows:

- 4 a) The NMS puts Port1 of ACC2 and Port2 of ACC1 into asymmetry measurement mode through the
 5 MIB (i.e., by setting the managed object asymmetryMeasurementMode for each port to TRUE).
 6 These two ports will not affect the PTP calculations of either node when the ports are in asymmetry
 7 measurement mode. If synchronization flowed over the link connecting these ports prior to their
 8 being put into asymmetry measurement mode, the BTCA will result in a reconfiguration of the
 9 synchronization spanning tree so that both ACC1 and ACC2 remain synchronized.
- 10 b) Port1 of ACC2 will send Pdelay_Req messages periodically using the peer delay measurement
 11 mechanism and receive Pdelay_Resp and Pdelay_Resp_Follow_Up messages. For each set of
 12 messages, ACC2 should save t3 [the pdelayRespEventEgressTimestamp (see 11.3.2.1), carried in
 13 the Pdelay_Resp_Follow_Up message] and t4 (the pdelayRespEventIngressTimestamp, taken when
 14 the Pdelay_Resp message timestamp point crosses the reference plane at Port1 of ACC2 on
 15 reception).
- 16 c) The NMS reads and saves multiple sets of (t3, t4) from ACC2 through the MIB. It is necessary that
 17 each set of (t3, t4) be from the same measurement, i.e., from the same peer delay message exchange.
 18 The number of (t3, t4) sets can be decided as required and is outside the scope of gPTP.
- 19 d) The tester manually exchanges the transmit and receive fibers of Port2 of ACC1 and Port1 of ACC2.
 20 Then the tester waits until the port status and protocol status become stable again.
- 21 e) Port1 of ACC2 will again make periodic peer delay measurements and save each set of
 22 measurement values (t3', t4') (the primes are used to denote measurements that have occurred after
 23 the transmit and receive fibers have been exchanged).
- 24 f) The NMS reads and saves the multiple sets of (t3', t4') from ACC2 through the MIB. Then the NMS
 25 can compute the delay asymmetry, in units of time, as $(t4' - t4) \times \text{nrrPdelay} - (t3' - t3)$. The NMS
 26 can use multiple sets of (t3, t4, t3', t4') to compute average values to get a more accurate result. The
 27 averaging method can be decided as required and is outside the scope of gPTP. If ACC1 and ACC2
 28 are frequency synchronized, then **nrrPdelay** is 1, and the delay asymmetry is $(t4' - t3') - (t4 - t3)$.

- 1 g) Based on the above result, the NMS sets the asymmetry value for Port1 of ACC2 and Port2 of ACC1
2 through the MIB.
- 3 h) After the NMS sets Port 1 of ACC2 and Port2 of ACC1 into normal mode (i.e., by setting the
4 managed object `asymmetryMeasurementMode` for each port to FALSE), the delay asymmetry
5 measurement of the testing link is completed. ACC1 and ACC2 will use the computed delay
6 asymmetry as compensation in the PTP calculations.

7 It is also possible to compute the asymmetry ratio, i.e., the ratio of the delay on the receive fiber at ACC2
8 (`Delay_rx_fiber`, the delay of the `Pdelay_Resp`) to the delay on the transmit fiber at ACC2 (`Delay_tx_fiber`,
9 the delay of the `Pdelay_Req`), both after line swapping. In this case, the NMS should collect multiple sets of
10 (t_1, t_2, t_3, t_4) and (t'_1, t'_2, t'_3, t'_4) before and after line-swapping, respectively, where t_1 is the
11 `pdelayReqEventEgressTimestamp` (taken when the `Pdelay_Req` message timestamp point crosses the
12 reference plane at Port1 of ACC2 on transmission), t_2 is the `pdelayReqEventIngressTimestamp` (carried in
13 the `requestReceiptTimestamp` field of the `Pdelay_Resp` message), and t_3 and t_4 are as given above. The
14 NMS can compute the delay on the receive fiber as $[(t'_4 - t_1) \times \text{nrrPdelay} - (t'_3 - t_2)] / 2$ and the delay on the
15 transmit fiber as $[(t_4 - t'_1) \times \text{nrrPdelay} - (t_3 - t'_2)] / 2$. Then the asymmetry ratio is as follows:

16 $\text{Delay_rx_fiber}/\text{Delay_tx_fiber} = [(t'_4 - t_1) \times \text{nrrPdelay} - (t'_3 - t_2)] / [(t_4 - t'_1) \times \text{nrrPdelay} - (t_3 - t'_2)]$

¹ Annex H

² (informative)

³ Bibliography

⁴ Bibliographical references are resources that provide additional or helpful material but do not need to be
⁵ understood or used to implement this standard. Reference to these resources is made for informational use
⁶ only.

⁷ [B1] AES3-2009 (reaffirmed in 2014), AES Recommended practice for digital audio engineering — Serial
⁸ transmission format for two-channel linearly represented digital audio data, Audio Engineering Society.²⁹

⁹ [B2] AES11-2009 (reaffirmed in 2014), AES recommended practice for digital audio engineering —
¹⁰ Synchronization of digital audio equipment in studio operations, Audio Engineering Society.

¹¹ [B3] Allan, David W., Neil Ashby, and Clifford C. Hodge, “The Science of Timekeeping,” Hewlett Packard
¹² Application 1289, 1997.³⁰

¹³ [B4] Garner, Geoffrey M., Derivation of FTM Parameters in 12.6 of 802.1AS-Rev, presentation to IEEE
¹⁴ 802.1 TSN TG, 28 Oct. 2018.³¹

¹⁵ Garner, Geoffrey M., End-to-End Jitter and Wander Requirements for ResE Applications, presentation to
¹⁶ IEEE 802.3 Residential Ethernet Study Group, May 2005.³²

¹⁷ [B5a] “Guidelines for Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI),
¹⁸ and Company ID (CID),” IEEE Registration Authority (<https://standards.ieee.org/products-programs/regauth/>).

²⁰ [B6] IEC 60958-3, Digital Audio Interface — Part 3: Consumer Applications, International Electrotechnical
²¹ Commission, Geneva, 2016.³³

²² [B7] IEC 60958-4, Digital Audio Interface — Part 4: Professional Applications (TA4), International
²³ Electrotechnical Commission, Geneva, 2016.

²⁴ [B8] IEEE Std 1003.1™-2008, IEEE Standard for Information Technology—Portable Operating System
²⁵ Interface (POSIX®) Base Specifications, Issue 7.^{34,35}

²⁶ [B9] IEEE Std 1139™-1999, IEEE Standard Definitions of Physical Quantities for Fundamental Frequency
²⁷ and Time Metrology—Random Instabilities.

²⁸ [B10] IEEE Std 1588™-2008, IEEE Standard for a Precision Clock Synchronization Protocol for
²⁹ Networked Measurement and Control Systems.

²⁹ AES publications are available from the Audio Engineering Society (<http://www.aes.org>).

³⁰ Available at (<http://www.allanstime.com>).

³¹ Available at (<http://www.ieee802.org/1/files/public/docs2018/as-garner-derivation-of-ftm-parameters-1118.pdf>).

³² Available at (http://www.ieee802.org/3/re_study/public/200505/garner_3_0505.pdf).

³³ IEC publications are available from the International Electrotechnical Commission (<https://www.iec.ch>).

³⁴ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org>).

³⁵ The IEEE standards or products referenced in this annex are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

- 1 [B11] IETF RFC 2578 (STD 58), Structure of Management Information Version 2 (SMIV2),
2 McCloghrie, K., D. Perkins, and J. Schoenwaelder, Apr. 1999.
- 3 [B12] IETF RFC 2579 (STD 58), Textual Conventions for SMIV2, McCloghrie, K., D. Perkins,
4 J. Schoenwaelder, J. Case, M. Rose, and S. Waldbusser, Apr. 1999.
- 5 IETF RFC 2580 (STD 58), Conformance Statements for SMIV2, McCloghrie, K., D. Perkins,
6 J. Schoenwaelder, J. Case, M. Rose, and S. Waldbusser, Apr. 1999.
- 7 [B13a] IETF RFC 6241, Network Configuration Protocol (NETCONF), June 2011.³⁶
- 8 [B13b] IETF RFC 6242, Using the NETCONF Protocol over Secure Shell (SSH), June 2011.
- 9 [B13c] IETF RFC 8341, Network Configuration Access Control Model, March 2018.
- 10 [B13d] IETF RFC 7589, Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual
11 X.509 Authentication, June 2015.
- 12 [B13e] IETF RFC 8040, RESTCONF Protocol, January 2017.
- 13 [B13f] IETF RFC 8340, YANG Tree Diagrams, March 2018.
- 14 [B13g] IETF RFC 8575, YANG Data Model for the Precision Time Protocol (PTP).
- 15 [B14] “International System of Units (SI), The” 9th edition, Bureau International des Poids et Mesures,
16 2019.³⁷
- 17 [B15] ISO 8601:2004, Data elements and interchange formats—Information interchange—Representation
18 of dates and times.³⁸
- 19 [B16] ISO/IEC 8802-2, Standard for Information technology—Telecommunications and information
20 exchange between systems—Local and metropolitan area networks—Specific requirements—Part 2:
21 Logical link control.³⁹
- 22 [B17] ISO/IEC 9945:2003, Information technology. Portable Operating System Interface (POSIX®).
- 23 [B18] ISO/IEC 14882:2003, Programming languages—C++.
- 24 [B19] IS-GPS-200J, Global Positioning Systems Directorate Systems Engineering & Integration Interface
25 Specification IS-GPS-200, Navstar GPS Space Segment/Navigation User Segment Interfaces, 25 Apr. 2018.
- 26 [B20] ITU-R Recommendation TF.460-6, Standard-frequency and time-signal emissions, 2002.⁴⁰
- 27 [B21] ITU-T Recommendation G.810, Definitions and Terminology for Synchronization Networks, ITU-T,
28 Geneva, Aug., 1996, Corregendum 1, Nov., 2001.

³⁶ IETF RFCs are available from the Internet Engineering Task Force (<https://www.ietf.org/>).

³⁷ Available at (https://www.bipm.org/en/publications/si_brochure/).

³⁸ ISO publications are available from the International Organization for Standardization (<https://www.iso.org>) and the American National Standards Institute (<https://www.ansi.org>).

³⁹ ISO/IEC publications are available from the International Organization for Standardization (<https://www.iso.org>), the International Electrotechnical Commission (<https://www.iec.ch>), and the American National Standards Institute (<https://www.ansi.org>).

⁴⁰ ITU publications are available from the International Telecommunications Union (<https://www.itu.int>).

1 [B22] Jekeli, Christopher, “Geometric Reference Systems in Geodesy,” Division of Geodesy and Geospatial
 2 Science, School of Earth Sciences, Ohio State University, July 2006.⁴¹

3 [B23] MoCA MAC/PHY Specification v1.0, MoCA-M/P-SPEC-V1.0-07122009, Multimedia over Coax
 4 Alliance (MoCA), July 12, 2009.⁴²

5 [B24] MoCA® MAC/PHY Specification Extensions v1.1, MoCA-M/P-SPEC-V1.1-06162009, Multimedia
 6 over Coax Alliance (MoCA), June 16, 2009.

7 [B24a] [OMG Unified Modeling Language \(OMG UML\), Version 2.5, March 2015.](#)⁴³

8 [B25] Papoulis, Athanasios, “Probability, Random Variables, and Stochastic Processes (Third Edition),”
 9 McGraw-Hill, 1991.

10 [B26] Petit, Gerard, and Brian Luzum (eds.), IERS Conventions, IERS Technical Note No. 36, 2010.⁴⁴

11 [B27] Proceedings of the 21st General Assembly of the IAU, IAU Trans., 1991, vol. XXIB, Kluwer.

12 [B28] Service de la Rotation Terrestre, Observatoire de Paris, 61, Av. de l’Observatoire 75014 Paris
 13 (France).

14 [B29] U.S. Naval Observatory.⁴⁵

15

⁴¹ Available at (https://kb.osu.edu/dspace/bitstream/1811/24301/1/Geom_Ref_Sys_Geodesy.pdf).

⁴² MoCA® specifications are available from the Multimedia over Coax Alliance (<http://www.mocalliance.org/specs>).

⁴³ OMG documents are available from the Object Management Group (<https://www.omg.org/>).

⁴⁴ Available at (<https://www.iers.org/SharedDocs/Publikationen/EN/IERS/Publications/tn/TechnNote36/tn36.pdf>).

⁴⁵ (<https://www.usno.navy.mil/USNO>).