



IEEE Standard for Local and Metropolitan Area Networks— Timing and Synchronization for Time-Sensitive Applications Amendment 3: Hot Standby and Clock Drift Error Reduction

IEEE Computer Society

Developed by the
LAN/MAN Standards Committee

IEEE Std 802.1ASdm™-2024

(Amendment to IEEE Std 802.1AS™-2020
as amended by IEEE Std 802.1AS™-2020/Cor 1-2021,
IEEE Std 802.1ASdr™-2024,
and IEEE Std 802.1ASdn™-2024)

IEEE Std 802.1ASdm™-2024
(Amendment to IEEE Std 802.1AS™-2020
as amended by IEEE Std 802.1AS™-2020/Cor 1-2021,
IEEE Std 802.1ASdr™-2024,
and IEEE Std 802.1ASdn™-2024)

IEEE Standard for Local and Metropolitan Area Networks—

Timing and Synchronization for Time-Sensitive Applications

Amendment 3: Hot Standby and Clock Drift Error Reduction

Developed by the
LAN/MAN Standards Committee
of the
IEEE Computer Society

Approved 26 September 2024
IEEE SA Standards Board

Abstract: This amendment to IEEE Std 802.1AS™-2020 specifies hot standby and addresses errors and omissions in the description of existing functionality.

Keywords: amendment, best timeTransmitter, frequency offset, Grandmaster Clock, Grandmaster PTP Instance, hot standby, IEEE 802.1AS™, IEEE 802.1ASdm™, phase offset, PTP End Instance, PTP Relay 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 4 October 2024 Printed in the United States of America.

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

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

PDF: ISBN 979-8-8557-1240-7 STD27316
Print: ISBN 979-8-8557-1241-4 STDPD27316

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/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.

Important Notices and Disclaimers Concerning IEEE Standards Documents

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

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

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

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

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

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

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

Translations

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

Use by artificial intelligence systems

In no event shall material in any IEEE Standards documents be used for the purpose of creating, training, enhancing, developing, maintaining, or contributing to any artificial intelligence systems without the express, written consent of the IEEE SA in advance. “Artificial intelligence” refers to any software, application, or other system that uses artificial intelligence, machine learning, or similar technologies, to analyze, train, process, or generate content. Requests for consent can be submitted using the [Contact Us](#) form.¹

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual is not, and shall not be considered or inferred to be, the official position of IEEE or any of its committees and shall not be considered to be or be relied upon as, a formal position of IEEE or IEEE SA. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter’s views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s). News releases about IEEE standards issued by entities other than IEEE SA should be considered the view of the entity issuing the release rather than the formal position of IEEE or IEEE SA.

Comments on standards

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

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or revisions to an IEEE standard is welcome to join the relevant IEEE SA working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).² An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.¹

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory

¹ Available at: <https://standards.ieee.org/about/contact/>.

² Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

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

Copyrights

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

Photocopies

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

Updating of IEEE Standards documents

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

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

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

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).⁴ Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the

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

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

Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

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

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

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

IMPORTANT NOTICE

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

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

Participants

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

Glenn Parsons, *Chair*

Jessy V. Rouyer, *Vice Chair*

János Farkas, *TSN Task Group Chair*

Geoffrey Garner, *Editor*

Katsuyuki Akizuki	Yoshihiro Ito	Atsushi Sato
Venkat Arunarthi	Michael Karl	Frank Schewe
Ralf Assmann	Stephan Kehrer	Michael Seaman
Rudy Belliardi	Marcel Kiessling	Maik Seewald
Christian Boiger	Gavin Lai	Ramesh Sivakolundu
Paul Bottorff	Yunping (Lily) Lyu	Johannes Specht
Radhakrishna Canchi	Christophe Mangin	Nemanja Stamenic
Feng Chen	Scott Mansfield	Marius Stanica
Abhijit Choudhury	Olaf Mater	Guenther Steindl
Anna Engelmann	David McCall	Karim Traore
Donald Fedyk	Martin Mittelberger	Max Turner
Norman Finn	Hiroki Nakano	Balazs Varga
Craig Gunther	Takumi Nomura	Ganesh Venkatesan
Stephen Haddock	Donald R. Pannell	Leon Wessels
Mark Hantel	Dieter Proell	Ludwig Winkel
Marc Holness	Karen Randall	Jordon Woods
Daniel Hopf	Maximilian Riegel	Takahiro Yamaura
Woojung Huh	Silvana Rodrigues	Nader Zein
Satoko Itaya	Rajeev Roy	

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

Thomas Alexander	Lokesh Kabra	Dieter Proell
Boon Chong Ang	Piotr Karocki	Denis Reilly
Butch Anton	Stephan Kehrer	Maximilian Riegel
Douglas Arnold	Stuart Kerry	Silvana Rodrigues
Stefan Aust	Yongbum Kim	Benjamin Rolfe
Denis Beaudoin	Jeff Koftinoff	Zoltan Roman
Christian Boiger	Gavin Lai	Jessy V. Rouyer
Vern Brethour	Hyeong Ho Lee	Reinhard Schrage
Juan Carreon	Joseph Levy	Jhony Sembiring
Pin Chang	Greg Luri	Johannes Specht
David Chen	Jingfei Lv	Guenther Steindl
Rodney Cummings	Christophe Mangin	Walter Struppler
János Farkas	William-Rogelio Marchand-Niño	David Tepen
Donald Fedyk	Brett McClellan	Richard Tse
Avraham Freedman	Jonathon McLendon	Max Turner
Geoffrey Garner	Martin Mittelberger	Ganesh Venkatesan
Craig Gunther	Ronald Murias	John Vergis
Marco Hernandez	Rajesh Murthy	Xiaohui Wang
Werner Hoelzl	Mfanasibili Nkonyane	Stephen Webb
Oliver Holland	Satoshi Obara	Karl Weber
Tetsushi Ikegami	Glenn Parsons	Scott Willy
Yoshihiro Ito	Bansi Patel	Ludwig Winkel
Abdul Jabbar	Arumugam Paventhan	Andreas Wolf
Raj Jain	Cam Posani	Oren Yuen
SangKwon Jeong	Clinton Powell	Qiyue Zou
Pranav Jha	Venkatesha Prasad	

When the IEEE SA Standards Board approved this standard on 26 September 2024, it had the following membership:

David J. Law, *Chair*
Jon W. Rosdahl, *Vice Chair*
Gary Hoffman, *Past Chair*
Alpesh Shah, *Secretary*

Sara R. Biyabani
Ted Burse
Stephen Dukes
Doug Edwards
J. Travis Griffith
Guido R. Hiertz
Ronald W. Hotchkiss

Hao Hu
Yousef Kimiagar
Joseph L. Koepfinger*
Howard Li
Xiaohui Liu
John Haiying Lu
Kevin W. Lu

Hiroshi Mano
Paul Nikolich
Robby Robson
Lei Wang
F. Keith Waters
Sha Wei
Philip B. Winston
Don Wright

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 802.1ASdm-2024, IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications—Amendment 3: Hot Standby and Clock Drift Error Reduction.

The first edition of IEEE Std 802.1AS was published in 2011. A first corrigendum, IEEE Std 802.1AS™-2011/Cor 1-2013, provided technical and editorial corrections. A second corrigendum, IEEE Std 802.1AS™-2011/Cor 2-2015 provided additional technical and editorial corrections.

The second edition, IEEE Std 802.1AS-2020, added support for multiple gPTP domains, Common Mean Link Delay Service, external port configuration, and Fine Timing Measurement for 802.11 transport. Backward compatibility with IEEE Std 802.1AS-2011 was maintained. A corrigendum, IEEE Std 802.1AS-2020/Cor 1-2021, provides technical and editorial corrections.

This amendment to IEEE Std 802.1AS-2020 specifies hot standby and addresses errors and omissions in the description of existing functionality. Hot standby guards against the failure of a single Grandmaster PTP Instance or the failure of communication from that Grandmaster PTP Instance to a Clock Target.

Contents

3. Definitions	16
4. Acronyms and abbreviations	17
5. Conformance.....	18
5.4 PTP Instance requirements and options.....	18
5.5 MAC-specific timing and synchronization methods for full-duplex IEEE 802.3 links	20
7. Time-synchronization model for a packet network	22
7.2 Architecture of a time-aware network	22
7.4 PTP Instance architecture	28
8. IEEE 802.1AS concepts and terminology	30
8.1 gPTP domain.....	30
8.5 Ports	31
9. Application interfaces	32
9.2 ClockSourceTime interface	32
9.3 ClockTargetEventCapture interface	32
9.4 ClockTargetTriggerGenerate interface	33
9.5 ClockTargetClockGenerator interface.....	34
9.6 ClockTargetPhaseDiscontinuity interface	35
10. Media-independent layer specification	37
10.2 Time-synchronization state machines.....	37
10.3 Best timeTransmitter clock selection, external port configuration, and announce interval setting state machines	49
10.4 State machines related to signaling gPTP capability	71
10.5 Message attributes.....	76
10.6 Message formats	76
10.7 Protocol timing characterization	79
11. Media-dependent layer specification for full-duplex point-to-point links.....	81
11.1 Overview.....	81
11.2 State machines for MD entity specific to full-duplex point-to-point links.....	82
11.3 Message attributes.....	104
11.4 Message formats	105
11.5 Protocol timing characterization	108
11.6 Control of computation of neighborRateRatio	110
13. Media-dependent layer specification for interface to IEEE 802.3 Ethernet passive optical network link	111
13.3 Message format.....	111
14. Timing and synchronization management.....	112
14.1 General.....	112
14.2 Default Parameter Data Set (defaultDS).....	115

14.3	Current Parameter Data Set (currentDS)	115
14.4	Parent Parameter Data Set (parentDS).....	115
14.7a	PTP Instance Synchronization Parameter Data Set (ptpInstanceSyncDS)	116
14.7b	Drift Tracking Parameter Data Set (driftTrackingDS)	117
14.8	Port Parameter Data Set (portDS).....	118
14.10	Port Parameter Statistics Data Set (portStatisticsDS).....	120
14.13	Asymmetry Measurement Mode Parameter Data Set (asymmetryMeasurementModeDS)	121
14.16	Common Mean Link Delay Service Link Port Parameter Data Set (cmlDsLinkPortDS)....	122
14.17	Common Mean Link Delay Service Link Port Parameter Statistics Data Set (cmlDsLinkPortStatisticsDS)	123
14.19	Hot Standby System Parameter Data Set (hotStandbySystemDS)	123
14.20	Hot Standby System Description Parameter Data Set (hotStandbySystemDescriptionDS)	125
16.	Media-dependent layer specification for CSN.....	126
16.4	Path delay measurement over a CSN backbone	126
16.5	Synchronization messages	126
17.	YANG data model	127
17.1	YANG framework	127
17.2	IEEE 802.1AS YANG data model	127
17.3	Structure of the YANG data model	132
17.5	YANG schema tree definitions	132
17.6	YANG modules	133
18.	Hot standby	141
18.1	General.....	141
18.2	Overview.....	141
18.3	PTP Instance configuration.....	142
18.4	PtpInstanceSyncStatus state machine	142
18.5	HotStandbySystem state machine.....	146
18.6	PrimarySecondaryOffset state machine.....	151
Annex A (normative)	Protocol Implementation Conformance Statement (PICS) proforma	153
A.5	Major capabilities	153
A.7	Minimal time-aware system.....	154
A.8	Signaling	155
A.9	Best timeTransmitter clock	155
A.10	Grandmaster-capable PTP Instance	156
A.11	Media-independent timeTransmitter.....	156
A.13	Media-dependent, full-duplex point-to-point link	156
A.19	Remote management.....	157
Annex B (normative)	Performance requirements	158
B.2	PTP Instance requirements	158
Annex F (informative)	PTP profile included in this standard	159
F.3	PTP attribute values	159
Annex G (informative)	The asymmetry compensation measurement procedure based on line-swapping .	160
G.3	Measurement procedure.....	160

Annex H (informative) Bibliography	162
--	-----

List of Figures

Figure 7-3—Time-aware network example for multiple gPTP domains	23
Figure 7-4—Time-aware network example for synchronization path redundancy, with one clock source providing time to two domains.....	25
Figure 7-5—Time-aware network example for GM redundancy with one primary GM and one secondary GM, which are separated in two gPTP domains	26
Figure 7-6—Time-aware network example for hot standby with both GM and partial path redundancy.....	27
Figure 7-8—Model for a PTP Instance and its interfaces to higher-layer applications model.....	29
Figure 10-2—Time-synchronization state machines—overview and interrelationships.....	38
Figure 10-6—ClockTimeTransmitterSyncOffset state machine	47
Figure 10-9—ClockTimeReceiverSync state machine.....	49
Figure 10-12—External port configuration state machines—overview and interrelationships	50
Figure 10-16—PortAnnounceInformationExt state machine	62
Figure 10-17—PortStateSettingExt state machine	65
Figure 10-18—PortAnnounceTransmit state machine	66
Figure 10-19—AnnounceIntervalSetting state machine.....	68
Figure 10-20—SyncIntervalSetting state machine	70
Figure 10-21—GptpCapableTransmit state machine	72
Figure 10-22—GptpCapableReceive state machine	73
Figure 10-23—GptpCapableIntervalSetting state machine.....	75
Figure 11-4—Detail of MD entity time-synchronization state machines for full-duplex point-to-point links.....	83
Figure 11-5—Peer-to-peer delay mechanism state machines—overview and interrelationships	84
Figure 11-6—MDSyncReceiveSM state machine.....	91
Figure 11-7—MDSyncSendSM state machine.....	94
Figure 11-9—MDPdelayReq state machine	101
Figure 17-1—Overview of YANG tree	128
Figure 17-2—PTP Instance detail.....	129
Figure 17-3—PTP Port detail	130
Figure 17-4—Common services detail	131
Figure 18-1—Model for hot standby entity in a time-aware system, and its interfaces to higher-layer applications.....	141
Figure 18-2—PtpInstanceSyncStatus state machine	145
Figure 18-3—HotStandbySystem state machine	151
Figure 18-4—PrimarySecondaryOffset state machine	152
Figure G-1—Asymmetry compensation measurement procedure	160

List of Tables

Table 10-1—Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5).....	41
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)	51
Table 10-9—Values of flag bits.....	77
Table 10-10—messageTypeSpecific semantics	78
Table 10-13—Signaling message fields	78
Table 11-1—Value of meanLinkDelayThresh for various links	85
Table 11-2—Summary of scope of global variables used by time synchronization state machines (see 10.2.4 and 10.2.5).....	86
Table 11-6—Value of correctionField.....	105
Table 11-10—Follow_Up message fields	106
Table 11-8—Sync message fields if twoStep flag is TRUE.....	106
Table 11-9—Sync message fields if twoStep flag is FALSE.....	106
Table 11-11a—Drift_Tracking TLV	107
Table 14-3—parentDS table	116
Table 14-6a—ptpInstanceSyncDS table.....	117
Table 14-6b—driftTrackingDS table.....	118
Table 14-8—delayMechanism enumeration.....	118
Table 14-10—portDS table.....	120
Table 14-12—portStatisticsDS table	121
Table 14-18—cmlDsLinkPortDS table	122
Table 14-15—asymmetryMeasurementModeDS table	122
Table 14-19—cmlDsLinkPortStatisticsDS table	123
Table 14-22—hotStandbySystemDescriptionDS table	125
Table 14-21—hotStandbySystemDS table	125
Table 17-1—Summary of the YANG modules	132

IEEE Standard for Local and Metropolitan Area Networks—

Timing and Synchronization for Time-Sensitive Applications

Amendment 3: Hot Standby and Clock Drift Error Reduction

(This amendment is based on IEEE Std 802.1AS™-2020 as amended by IEEE Std 802.1AS™-2020/Cor 1-2021, IEEE Std 802.1ASdr-2024, and IEEE Std 802.1ASdn-2024.)

NOTE—The editing instructions contained in this amendment define how to merge the material contained therein into the existing base standard and its amendments to form the comprehensive standard.

The editing instructions are shown in ***bold italic***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strike~~~~through~~ (to remove old material) and underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Deletions and insertions may require renumbering. If so, renumbering instructions are given in the editing instruction. ***Replace*** is used to make changes in figures or equations by removing the existing figure or equation and replacing it with a new one. Editing instructions, change markings, and this NOTE will not be carried over into future editions because the changes will be incorporated into the base standard.⁶

⁶ Notes in text, tables, and figures are given for information only, and do not contain requirements needed to implement the standard.

3. Definitions

Insert 3.3a after 3.3 “clock”, as follows:

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

Insert 3.6a after 3.6 “end station”, as follows:

3.6a epoch: The origin of a timescale.

Change 3.14 as follows:

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

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

Insert 3.20a after 3.20 “primary reference”, as follows:

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

Insert 3.27a after 3.27 “residence time”, as follows:

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

Insert 3.32a after 3.32 “time-aware system”, as follows:

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

4. Acronyms and abbreviations

Insert the following new abbreviation into Clause 4 in alphanumeric order:

QSDO Qualified Standards Development Organization

5. Conformance

5.4 PTP Instance requirements and options

Change the title and text of 5.4.1 (as amended by IEEE Std 802.1AS-2020/Cor 1-2021 and IEEE Std 802.1ASdr-2024) as follows:

5.4.1 ~~Summary of~~ PTP Instance requirements

An implementation of a PTP Instance shall:

- a) Implement the generalized precision time protocol (gPTP) requirements specified in Clause 8.
- b) Support the requirements for time-synchronization state machines (10.1.2, 10.2.1, 10.2.2, 10.2.3, 10.2.4, 10.2.5, and 10.2.6).
- c) Support at least one PTP Port.
- d) On each supported PTP Port, implement the PortSyncSyncReceive state machine (10.2.8).
- e) Implement the ClockTimeReceiverSync state machine (10.2.13).
- ~~f) Support the following best time Transmitter clock algorithm (BTCA) requirements:~~
 - ~~1) 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).~~
 - ~~2) For domain 0, implement specifications for externalPortConfigurationEnabled value of FALSE (10.3.1).~~
 - ~~3) Implement the PortAnnounceReceive state machine (10.3.11).~~
 - ~~4) Implement the PortAnnounceInformation state machine (10.3.12).~~
 - ~~5) Implement the PortStateSelection state machine (10.3.13).~~
 - ~~6) Have the BTCA as the default mode of operation, with externalPortConfiguration FALSE, on domain 0.~~
 - ~~7) Implement at least one of the possibilities for externalPortConfigurationEnabled (i.e., FALSE, meaning the BTCA is used, and TRUE, meaning external port configuration is used) on domains other than domain 0.~~
- ~~f)~~ g) Implement the SiteSyncSync state machine (10.2.7).
- ~~g)~~ h) Implement the state machines related to signaling gPTP capability (~~10.4~~10.4.a, 10.4.1, 10.4.2).

NOTE 1—5.4.1 g) does not include the GtpCapableIntervalSetting state machine (10.4.3), despite its name containing “GtpCapable”.

NOTE 2—The GtpCapableTransmit and GtpCapableReceive state machines are required; however, they can be disabled by setting the managed object gtpCapableStateMachinesEnabled (see 14.8.54a) to FALSE.

- ~~h)~~ i) For receipt of all messages and for transmission of all messages except Announce (see 10.6.3) ~~and Signaling (see 10.6.4)~~, support the message requirements as specified in 10.5, 10.6, and 10.7.

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

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

5.4.2 PTP Instance options

Change item c) in the lettered list in 5.4.2 (as amended by IEEE Std 802.1ASdr-2024) as follows:

- c) Support the following for Grandmaster PTP Instance capability:
 - 1) Support the media-independent timeTransmitter capability specified in item b) of 5.4.2.
 - 2) Support the requirements for a grandmaster-capable PTP Instance (10.1.3).
 - 3) Implement the ClockTimeTransmitterSyncSend state machine (10.2.9).
 - 4) Implement the ClockTimeTransmitterSyncOffset state machine (10.2.10).
 - 5) Implement the ClockTimeTransmitterSyncReceive state machine (10.2.11).

Change items e) to l) in the lettered list in 5.4.2 (as amended by IEEE Std 802.1ASdr-2024 and IEEE Std 802.1ASdn-2024) as follows:

- ~~e) Support transmit of the Signaling message according to the message requirements as specified in 10.5 and 10.6.~~
- ~~e)~~ e) Support more than one PTP Instance; such support allows for more than one domain (7.2.3).
- ~~f)~~ f) Support the following external port configuration capability on at least one port:
 - 1) Implement specifications for externalPortConfigurationEnabled value of true (10.3.1).
 - 2) Implement the PortAnnounceInformationExt state machine (10.3.14).
 - 3) Implement the PortStateSettingExt state machine (10.3.15).
- ~~g)~~ g) Implement the SyncIntervalSetting state machine (10.3.18).
- ~~h)~~ h) Implement the GtpCapableIntervalSetting state machine.
- i) Implement one or more of the application interfaces specified in Clause 9; A PTP Instance that claims to support application interfaces shall state which application interfaces are supported.
- j) Support timing and synchronization management as specified in Clause 14.
- k) Support the use of a remote management protocol. A PTP Instance that claims to support remote management shall:
 - 1) State which remote management protocol standard(s) or specification(s) are supported (see A.19).
 - 2) State which standard(s) or specification(s) for managed object definitions and encodings are supported for use by the remote management protocol (see A.19).
 - 3) If the Simple Network Management Protocol (SNMP) is supported as a remote management protocol, support the managed object definitions specified as Structure of Management Information version 2 (SMIv2) Management Information Base (MIB) modules in Clause 15.
 - 4) If YANG is supported with a remote management protocol, support the YANG data model ieee802-dot1as-gtp in Clause 17.
 - 5) If YANG is supported with a remote management protocol, and if hot standby is supported, support the YANG data model ieee802-dot1as-hs in Clause 17.
- l) Implement both BTCA and external port configuration on domains other than domain 0; if both possibilities are implemented on domains other than domain 0, the default value of externalPortConfigurationEnabled shall be FALSE.
- 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.
- 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, and 11.4.4.

Insert 5.4.2a after 5.4.2 as follows:

5.4.2a PTP Instance defaults and recommendations

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

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

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

- 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).
- b) For domain 0, implement specifications for an externalPortConfigurationEnabled value of FALSE (10.3.1).
- c) Implement the PortAnnounceReceive state machine (10.3.11).
- d) Implement the PortAnnounceInformation state machine (10.3.12).
- e) Implement the PortStateSelection state machine (10.3.13).
- f) Have the BTCA as the default mode of operation, with externalPortConfigurationEnabled FALSE, on domain 0.
- g) Implement at least one of the possibilities for externalPortConfigurationEnabled (i.e., FALSE, meaning the BTCA is used, and TRUE, meaning external port configuration is used) on domains other than domain 0.

5.4.3 PTP Relay Instance requirements

Change 5.4.3 as follows:

An implementation of a PTP Relay Instance shall:

- a) Support more than one PTP Port.
- ~~b) Support the PTP Instance requirements specified in 5.4.~~
- b) ~~e)~~ Support the media-independent timeTransmitter capability specified in item b) of 5.4.2.

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

Change 5.5 (as amended by IEEE Std 802.1AS-2020/Cor 1-2021) as follows:

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

- a) Support full-duplex operation, as specified in 4.2 and Annex 4A of IEEE Std 802.3-2018.

- b) Support the requirements as specified in Clause 11 [with the exception of requirements more specifically addressed in this subclause (5.5)].
- e) ~~Implement the SyncIntervalSetting state machine (10.3.18).~~
- c) Support two-step capability on receive as specified in 11.2.14.
- d) Support two-step capability on transmit as specified in 11.2.15.
- e) ~~Provide the Common Mean Link Delay Service (CMLDS) if the time-aware system implements more than one domain, as specified in 11.2.17 and 11.2.18.~~

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

- 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.~~
- g) ~~Support one-step capability on receive as specified in 11.2.14.~~
- h) ~~Support one-step capability on transmit as specified in 11.2.15.~~
- i) ~~Support the OneStepTxOperSetting state machine specified in 11.2.16.~~
- j) ~~Support propagation delay averaging, as specified in 11.2.19.3.4.~~
- k) ~~Provide the Common Mean Link Delay Service (CMLDS) if the time-aware system implements only one domain, as specified in 11.2.17 and 11.2.18.~~
- l) Implement the LinkDelayIntervalSetting state machine (11.2.21).

7. Time-synchronization model for a packet network

7.2 Architecture of a time-aware network

7.2.1 General

Change 7.2.1 as follows:

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

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

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

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

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

7.2.2 Time-aware network consisting of a single gPTP domain

Change the second paragraph of 7.2.2 (as amended by IEEE Std 802.1ASdr-2024) as follows:

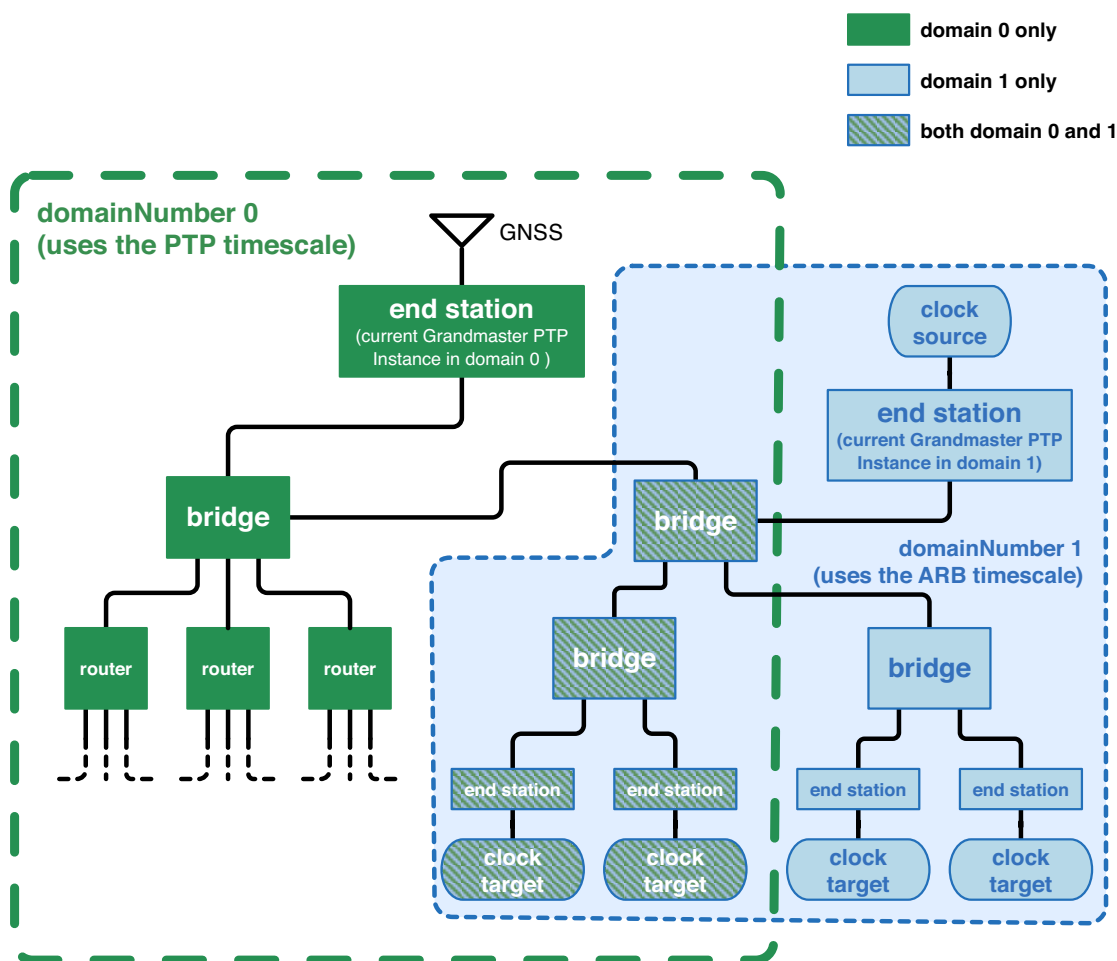
Any PTP Instance with clock sourcing capabilities can be a potential Grandmaster PTP Instance, and a selection method (the *best timeTransmitter clock algorithm*, or BTCA) ensures that all of the PTP Instances in ~~a~~^{the} gPTP domain use the same Grandmaster PTP Instance.⁷ The BTCA is largely identical to that used in IEEE Std 1588-2019, but somewhat simplified. In Figure 7-1 the BTCA process has resulted in the Grandmaster PTP Instance being on the network backbone. If, however, the access network fails, the systems on a local network automatically switch over to one of the potential Grandmaster PTP Instances on the local network that is as least as “good” as any other. For example, in Figure 7-2, the access network link has failed, and a potential Grandmaster PTP Instance that has a GNSS reference source has become the

⁷ 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.

active Grandmaster PTP Instance. As a result, now two gPTP domains exist where there used to be one. ~~Finally, note that when a time-aware system supports more than one domain, one of the domains supported must be domain 0 for backward compatibility with the 2011 edition of this standard, though domain 0 is not necessarily active in a time-aware system.~~

7.2.3 Time-aware network consisting of multiple gPTP domains

Replace Figure 7-3 with the following figure:



NOTE 1—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.

NOTE 2—This figure differs from the 2020 edition of this standard, with Corrigendum 1 applied, in that “domain number” is replaced by “domainNumber” in two places, the sentence indicating that the PTP links in the figure can use any of the media specified in this standard is added, and the text at the top of the figure is made a NOTE to this figure.

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

Change the title of 7.2.4 as follows:

7.2.4 Time-aware networks using BTCA ~~with dundant Grandmaster PTP Instances and/or redundant paths~~

Change the titles and content of 7.2.4.1 and 7.2.4.2 (as amended by IEEE Std 802.1ASdr-2024) as follows:

7.2.4.1 General

~~Redundancy has many levels of sophistication, performance, and cost. Therefore, the appropriate level and/or amount of redundancy required in a time-aware network can be very different for each application. Nonetheless, all solutions for redundancy consist of a detection component, a correction component, and an action component. The detection component detects that something is not working correctly. The correction component determines the appropriate corrective action. The action component performs the required action(s) to fix the detected problem.~~

7.2.4.2 Redundancy specified in this standard (BTCA)

This standard provides a basic level of redundancy as follows:

- A detection component that triggers when no Sync or Announce messages are received for a defined period of time ~~the current Grandmaster PTP Instance stops working (i.e., loss of Sync messages and Announce messages for a period of time) or if the link to the Grandmaster PTP Instance goes down (i.e., immediate loss of Sync messages and Announce messages).~~
- 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.
- An action component, where the winning Grandmaster PTP Instance starts sending Announce messages and Sync messages ~~and all the PTP Instances listen to this new Grandmaster PTP Instance.~~

Change the title and the first four paragraphs of 7.2.4.3 as follows:

7.2.4.3 Redundacy not fully specified in this standard

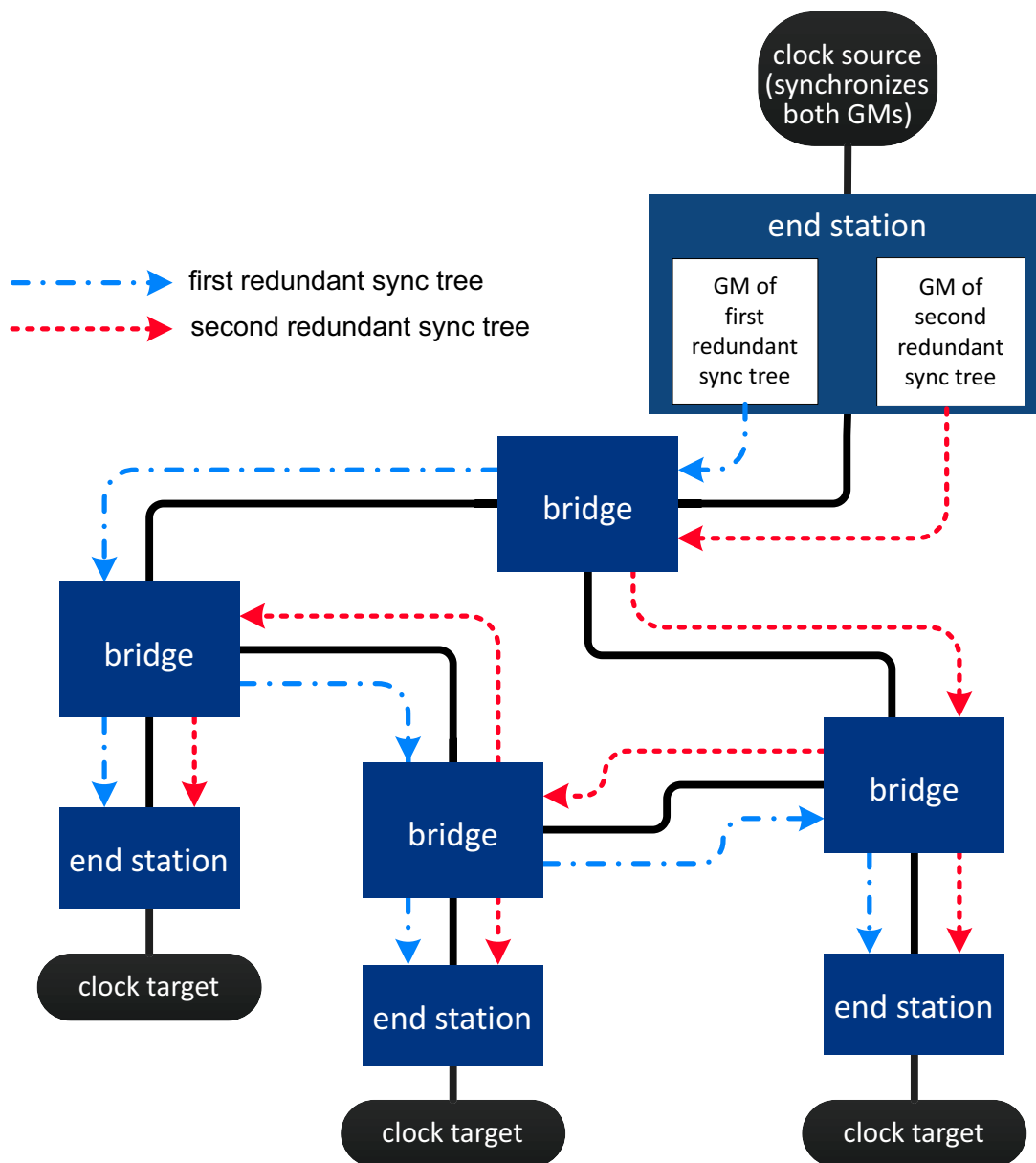
In addition to providing the basic level of redundancy, this standard provides the ability to support more sophisticated network configurations that provide additional levels of Grandmaster PTP Instance and clock path redundancy. Figure 7-4 ~~through and Figure 7-6~~ Figure 7-5 are examples of such networks 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.

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

~~Other types of faults involve instability of the Grandmaster Clock, such as time glitches, excess jitter, or wander, or various other impairments that could occur in the Grandmaster Clock. Techniques for identifying these types of failures, and the appropriate correction necessary, are not specified in this standard. However, if other techniques or standards are used for detection and correction of these types of failures, this standard provides the means to recover from these errors.~~

Figure 7-4 shows an example network realizing two redundant synchronization trees from a single GM, each with its own GM. The two GMs receive timing from the single clock source ~~with each synchronization tree in a different gPTP domain (there are a total of two gPTP domains).~~

Replace Figure 7-4 with the following figure:



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

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

The diagram illustrates the synchronization of two clock domains (primary and secondary) across a network of end stations and bridges. The primary clock domain (left) has a 'clock source' at the top, which connects to an 'end station (primary GM)'. This end station connects to a 'bridge'. The secondary clock domain (right) has a 'clock source' at the top, which connects to an 'end station (secondary GM)'. This end station connects to a 'bridge'. Both bridges are connected to each other. Each bridge connects to two 'end station' nodes. Each 'end station' node connects to a 'clock target' node. The diagram shows two types of connections: a 'sync tree of the primary GM' (blue dashed arrows) and a 'sync tree of the secondary GM' (red dashed arrows). The primary sync tree flows from the primary clock source through the primary end station and bridge to the primary end stations. The secondary sync tree flows from the secondary clock source through the secondary end station and bridge to the secondary end stations. Additionally, there are cross-domain connections: a blue dashed arrow from the primary bridge to the secondary bridge, and a red dashed arrow from the secondary bridge to the primary bridge, indicating synchronization between the two domains.

NOTE 3—GM denotes Grandmaster PTP Instance.

AVA Documents
Document Sharing Hub

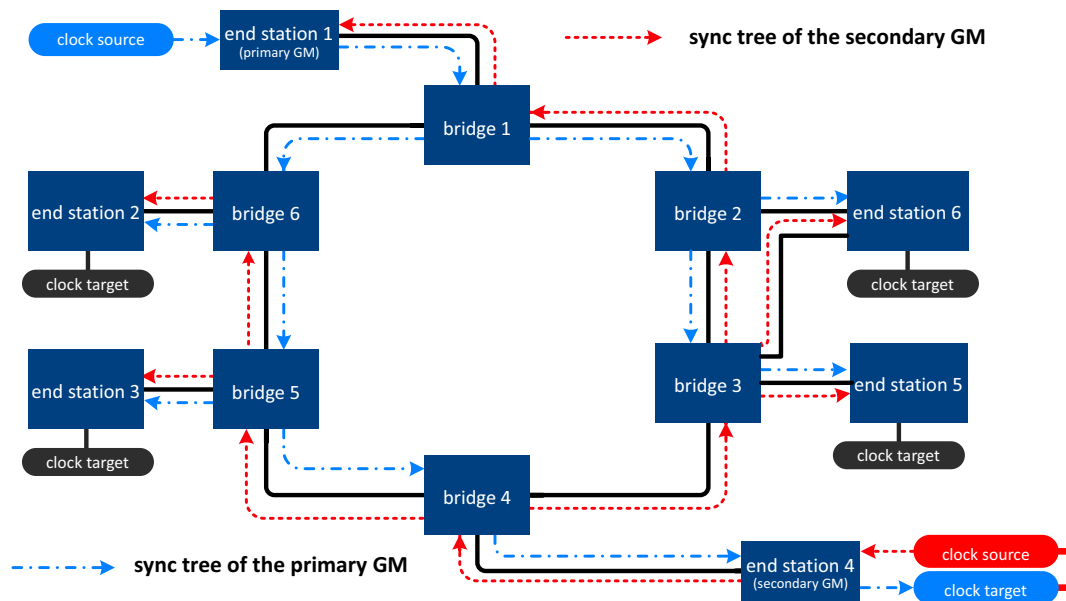
Delete the last two paragraphs of 7.2.4.3 (beginning “Figure 7-6 shows ...”) and also delete Figure 7-6.

Insert 7.2.5 (including a new Figure 7-6) after 7.2.4 as follows:

7.2.5 Time-aware network with hot standby

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

End station 1 is the primary GM (see Clause 18), and its synchronization spanning tree is illustrated by the arrows of the primary grandmaster sync tree. End station 4 is the secondary GM, and its synchronization spanning tree is illustrated by the arrows of the secondary grandmaster sync tree. Except for the links to end stations 2, 3, and 5, the network can tolerate the failure of any single link and the Bridges and end stations remain synchronized by either the primary or the secondary GM. For example, if the link between bridge 3 and end station 6 fails, end station 6 is still synchronized by the primary domain via bridge 2. If the link between bridge 5 and bridge 6 fails, bridge 6 is still synchronized by the primary domain and bridge 5 is synchronized by the secondary domain. If the secondary GM fails, all the Bridges and end stations are still synchronized by the primary GM.



NOTE 1—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 2—GM denotes Grandmaster PTP Instance.

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

If the primary GM fails, all the Bridges and end stations are still synchronized by the secondary GM. In this case, if syncLocked is TRUE the primary PTP Instance of end station 4 stops receiving Sync messages and isSynced changes to FALSE; the hotStandbySystemState of end station 4 is then changed to NOT_REDUNDANT (see Clause 18). If syncLocked is FALSE, and assuming the Bridges and end stations other than the GMs are not grandmaster capable, the time difference between the primary and secondary

domain eventually exceeds the primarySecondaryOffsetThresh (see Clause 18) and the hotStandbySystemState of end station 4 changes to NOT_REDUNDANT. Irrespective of the syncLocked state, the secondary GM no longer takes timing from the primary domain.

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

7.4 PTP Instance architecture

Change the first paragraph of 7.4 and the caption of Figure 7-8 as follows:

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

Change the title of Figure 7-8 (as amended by IEEE Std 802.1ASdr-2024) as shown on the following page:

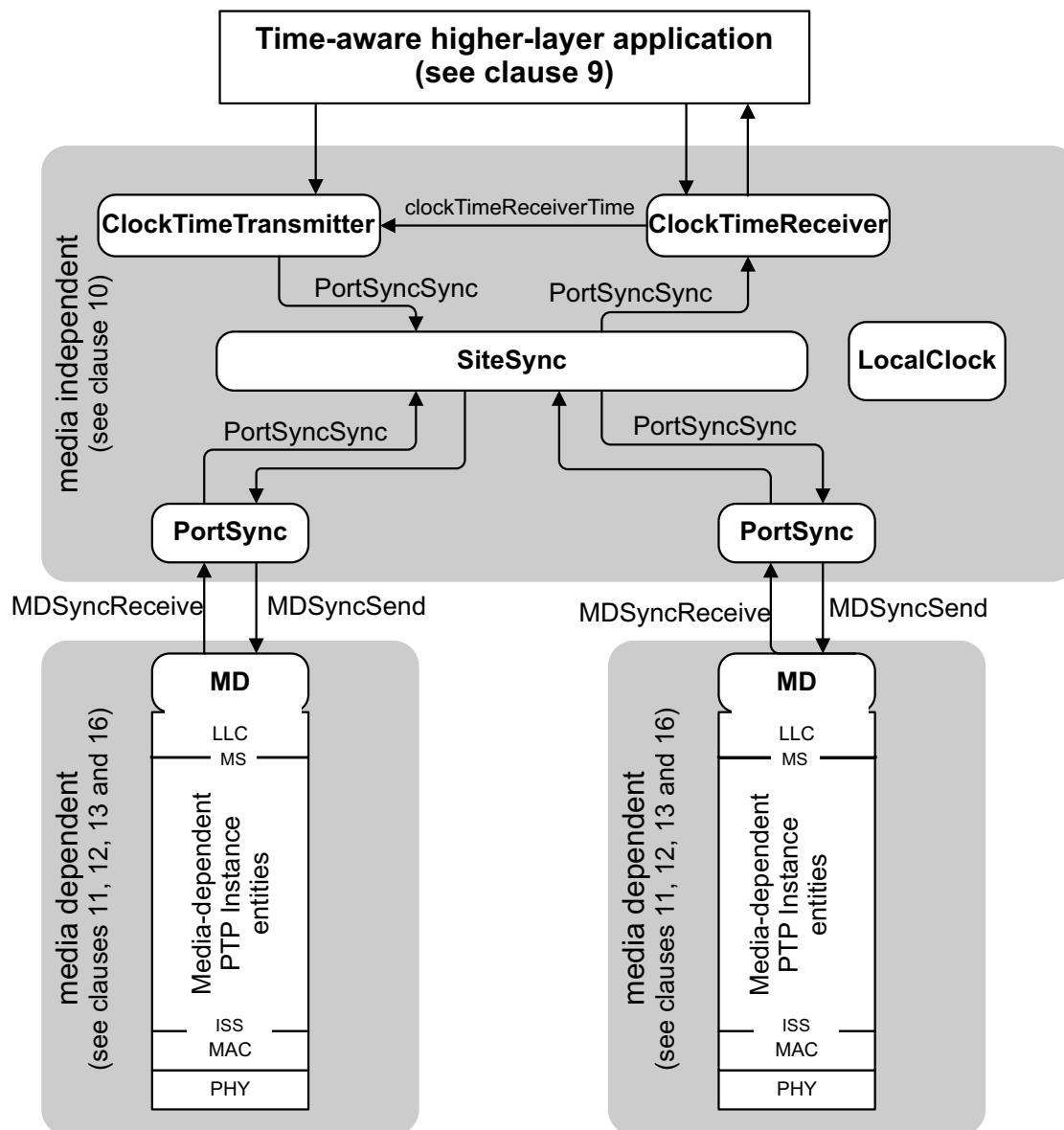


Figure 7-8—Model for a PTP Instance and its interfaces to higher-layer applicationsmodel

8. IEEE 802.1AS concepts and terminology

8.1 gPTP domain

Change 8.1 as follows:

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

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

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

A time-aware system shall support one or more domains, each with a distinct ~~domain number~~domainNumber in the range 0 through 127. ~~A time-aware system shall support the domain whose domain-number is 0, and that domain-number shall not be changed to a nonzero value.~~ Unless otherwise specified in this standard, the operation of gPTP and the timescale in any given domain is independent of operation in any other domain.

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

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

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

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

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

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

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

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

8.5 Ports

8.5.2 Port identity

8.5.2.3 Port number

Change 8.5.2.3 (as amended by IEEE Std 802.1ASdr-2024) as follows:

The portNumber values for the PTP Ports on a time-aware system shall be distinct in the range ~~1, 2, ...,~~
0x0001 through 0xFFFE.

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

9. Application interfaces

9.2 ClockSourceTime interface

9.2.2 ClockSourceTime.invoke function parameters

9.2.2.1 domainNumber (UInteger8)

Change 9.2.2.1 as follows:

This parameter is the ~~domain-number~~domainNumber of the gPTP domain to which this ClockSource entity is providing time.

9.2.2.3 timeBaseIndicator (UInteger16)

Change 9.2.2.3 as follows:

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

NOTE—While the clock that supplies time to the ClockSource entity can be lost, ~~i.e., the PTP Instance can enter holdover~~, the ClockSource entity itself is not lost. The ClockSource entity ensures that timeBaseIndicator changes if the source of time is lost.

9.3 ClockTargetEventCapture interface

9.3.2 ClockTargetEventCapture.invoke parameters

9.3.2.1 domainNumber (UInteger8)

Change 9.3.2.1 (as amended by IEEE Std 802.1ASdr-2024) as follows:

This parameter is the ~~domain-number~~domainNumber of the ClockTimeReceiver entity that is requested to provide the synchronized time of the signaled event.

Change 9.3.3 and its subclauses (as amended by IEEE Std 802.1ASdr-2024) as follows:

9.3.3 ClockTargetEventCapture.result parameters

```
ClockTargetEventCapture.result {  
    domainNumber,  
    timeReceiverTimeCallback,  
    gmPresent,  
    isSynced,  
    grandmasterIdentity  
}
```

9.3.3.1 domainNumber (UInteger8)

This parameter is the ~~domain-number~~domainNumber of the ClockTimeReceiver entity that is providing the synchronized time of the signaled event.

9.3.3.2 timeReceiverTimeCallback (ExtendedTimestamp)

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

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

9.3.3.3 gmPresent (Boolean)

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

9.3.3.4 isSynced (Boolean)

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

9.3.3.5 grandmasterIdentity (ClockIdentity)

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

9.4 ClockTargetTriggerGenerate interface

9.4.2 ClockTargetTriggerGenerate.invoke parameters

9.4.2.1 domainNumber (UInteger8)

Change 9.4.2.1 (as amended by IEEE Std 802.1ASdr-2024) as follows:

This parameter is the ~~domain-number~~domainNumber of the ClockTimeReceiver entity that is requested to signal an event at the specified time.

Change 9.4.3 and its subclauses (as amended by IEEE Std 802.1ASdr-2024) as follows:

9.4.3 ClockTargetTriggerGenerate.result parameters

```
ClockTargetTriggerGenerate.result {  
    domainNumber,  
    errorCondition,  
    gmPresent,  
    isSynced,  
    grandmasterIdentity  
}
```

9.4.3.1 domainNumber (UInteger8)

This parameter is the ~~domain-number~~domainNumber of the ClockTimeReceiver entity that is triggering an event at the specified time.

9.4.3.2 errorCondition (Boolean)

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

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

9.4.3.3 gmPresent (Boolean)

~~The value of gmPresent is set equal to the value of the global variable gmPresent (see 10.2.4.13). This parameter indicates to the ClockTarget whether a Grandmaster PTP Instance is present.~~ As specified in 9.3.3.3.

9.4.3.4 isSynced (Boolean)

As specified in 9.3.3.4.

9.4.3.5 grandmasterIdentity (ClockIdentity)

As specified in 9.3.3.5.

9.5 ClockTargetClockGenerator interface

9.5.2 ClockTargetClockGenerator.invoke parameters

9.5.2.1 domainNumber (UInteger8)

Change 9.5.2.1 as amended by IEEE Std 802.1ASdr-2024) as follows:

This parameter is the ~~domain-number~~ domainNumber of the ClockTimeReceiver entity that is requested to deliver a periodic clock signal.

Change 9.5.3 and its subclauses (as amended by IEEE Std 802.1ASdr-2024) as follows:

9.5.3 ClockTargetClockGenerator.result parameters

```
ClockTargetClockGenerator.result {  
    domainNumber,  
    gmPresent,  
    timeReceiverTimeCallback,  
    isSynced,  
    grandmasterIdentity  
}
```

9.5.3.1 domainNumber (UInteger8)

This parameter is the ~~domain-number~~ domainNumber of the ClockTimeReceiver entity that is delivering a periodic clock signal.

9.5.3.2 gmPresent

As specified in 9.3.3.3.

9.5.3.3 ~~9.5.3.2~~ timeReceiverTimeCallback (ExtendedTimestamp)

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

9.5.3.4 isSynced Boolean)

As specified in 9.3.3.4.

9.5.3.5 grandmasterIdentity (ClockIdentity)

As specified in 9.3.3.5.

9.6 ClockTargetPhaseDiscontinuity interface

Change 9.6.2 and its subclauses (as amended by IEEE Std 802.1ASdr-2024) as follows:

9.6.2 ClockTargetPhaseDiscontinuity.result parameters

```
ClockTargetPhaseDiscontinuity.result {  
    domainNumber,  
    grandmasterIdentity,  
    gmIdentity,  
    gmTimeBaseIndicator,  
    lastGmPhaseChange,  
    lastGmFreqChange,  
    isSynced,  
}
```

9.6.2.1 domainNumber (UInteger8)

This parameter is the ~~domain-number~~domainNumber of the ClockTimeReceiver entity that is providing discontinuity information.

9.6.2.2 grandmasterIdentity (ClockIdentity)

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

9.6.2.3 gmTimeBaseIndicator (UInteger16)

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

9.6.2.4 lastGmPhaseChange (ScaledNs)

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

9.6.2.5 lastGmFreqChange (Float64)

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

9.6.2.6 isSynced (Boolean)

As specified in 9.3.3.4.

9.6.2.7 grandmasterIdentity (ClockIdentity)

As specified in 9.3.3.5.

10. Media-independent layer specification

10.2 Time-synchronization state machines

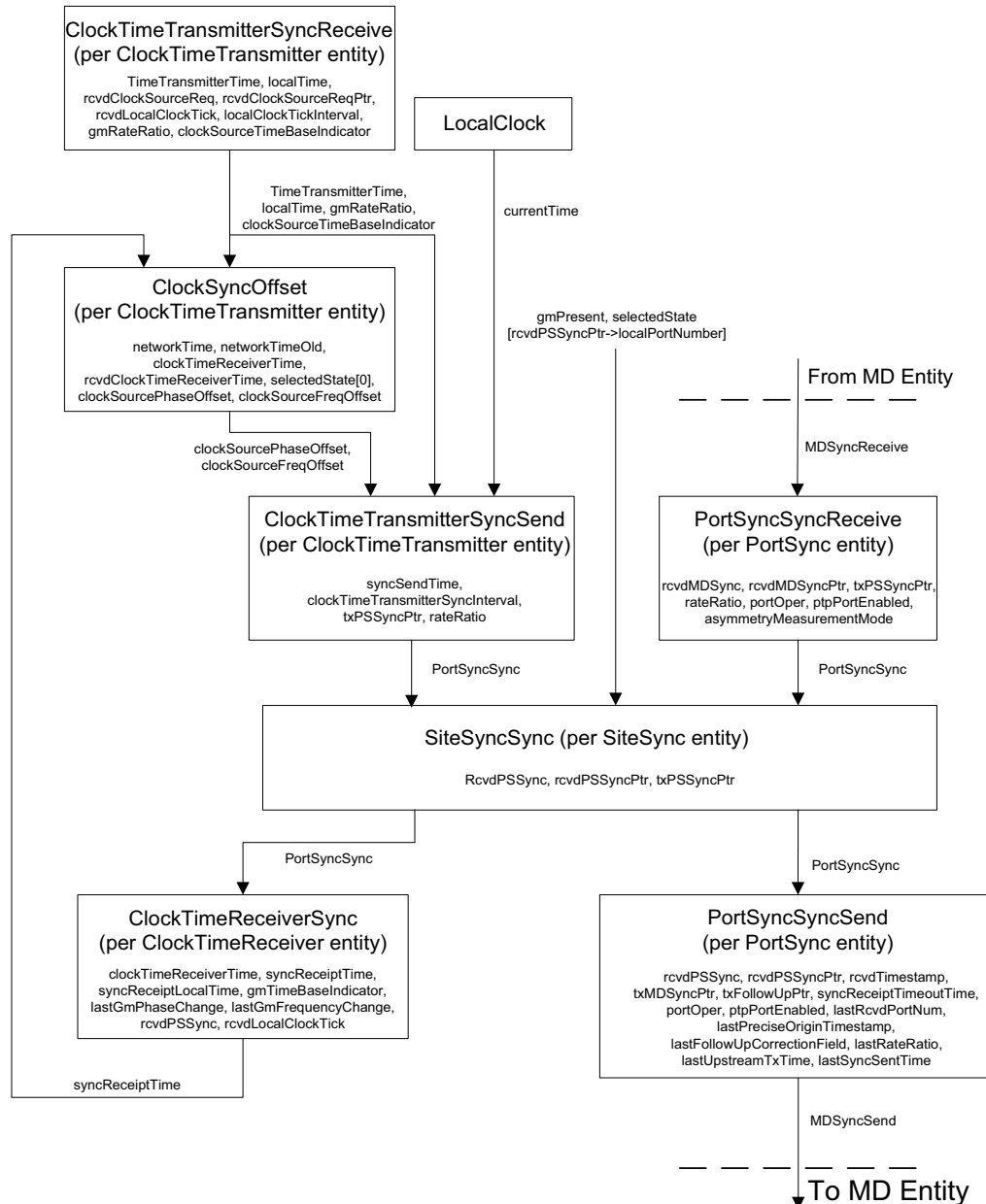
Change the first two paragraphs of 10.2.1 (as amended by IEEE Std 802.1ASdr-2024) as follows:

10.2.1 Overview

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

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

Replace Figure 10-2 (as amended by IEEE Std 802.1ASdr-2024) with the following figure:



Notes:

- selectedState for each port and gmPresent are set by Port State Selection state machine (see 10.3.12)
- currentTime is a global variable that is always equal to the current time relative to the local oscillator
- application interfaces to higher layers are not shown
- 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

Change the last paragraph and the lettered list in 10.2.1 (as amended by IEEE Std 802.1ASdr-2024) as follows:

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

- a) ClockTimeTransmitterSyncReceive (one instance per PTP Instance): receives ClockSourceTime.invoke functions from the ClockSource entity and notifications of LocalClock entity ticks (see 10.2.4.18), updates timeTransmitterTime, and provides timeTransmitterTime to ClockTimeTransmitterSyncOffset and ClockTimeTransmitterSyncSend state machines.
- b) ClockTimeTransmitterSyncOffset (one instance per PTP Instance): receives syncReceiptTime from the ClockTimeReceiver entity and timeTransmitterTime from the ClockTimeTransmitterSyncReceive state machine, computes phase offset and frequency offset between timeTransmitterTime and syncReceiptTime if the PTP Instance is not the Grandmaster PTP Instance, and provides the frequency and phase offsets to the ClockTimeTransmitterSyncSend state machine.
- c) ClockTimeTransmitterSyncSend (one instance per PTP Instance): receives timeTransmitterTime from the ClockTimeTransmitterSyncReceive state machine, receives phase and frequency offset between timeTransmitterTime and syncReceiptTime from the ClockTimeTransmitterSyncOffset state machine, and provides timeTransmitterTime (i.e., synchronized time) and the phase and frequency offset to the SiteSync entity using a PortSyncSync structure.
- d) PortSyncSyncReceive (one instance per PTP Instance, per PTP Port): receives time-synchronization information from the MD entity of the corresponding PTP Port, computes accumulated rateRatio, computes syncReceiptTimeoutTime, and sends the information to the SiteSync entity.
- e) SiteSyncSync (one instance per PTP Instance): receives time-synchronization information, accumulated rateRatio, and syncReceiptTimeoutTime from the PortSync entity of the current timeReceiver port or from the ClockTimeTransmitter entity; and sends the information to the PortSync entities of all the ports and to the ClockTimeReceiver entity.
- f) PortSyncSyncSend (one instance per PTP Instance, per PTP Port): receives time-synchronization information from the SiteSync entity, requests that the MD entity of the corresponding PTP Port send a time-synchronization event message, receives the syncEventEgressTimestamp for this event message from the MD entity, uses the most recent time-synchronization information received from the SiteSync entity and the timestamp to compute time-synchronization information that will be sent by the MD entity in a general message (e.g., for full-duplex IEEE 802.3 media) or a subsequent event message (e.g., for IEEE 802.11 media), and sends this latter information to the MD entity.
- g) ClockTimeReceiverSync (one instance per PTP Instance): receives time-synchronization information from the SiteSync entity; computes ClockTimeReceiverTime and syncReceiptTime; sets syncReceiptLocalTime, GmTimeBaseIndicator, lastGmPhaseChange, and lastGmFreqChange; sends ClockTimeReceiverTime to the ClockTimeTransmitter entity; and provides information to the ClockTarget entity (via the ClockTargetPhaseDiscontinuity interface; see 9.6) to enable that entity to determine if a phase or frequency discontinuity has occurred.

10.2.2 Data structures communicated between state machines

10.2.2.1 MDSyncSend

10.2.2.1.2 domainNumber (UInteger8)

Change 10.2.2.1.2 as follows:

This parameter is the ~~domain-number~~domainNumber of the gPTP domain in which this structure is sent.

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

10.2.2.1.7 upstreamTxTime (UScaledNs)

Change 10.2.2.1.7 (as amended by IEEE Std 802.1ASdr-2024) as follows:

The upstreamTxTime is given by the following equation:

$$\text{upstreamTxTime} = \text{syncEventIngressTimestamp} - \frac{\text{meanLinkDelay}}{\text{neighborRateRatio}}$$

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

where

syncEventIngressTimestamp	corresponds to the receipt of the time-synchronization information at the timeReceiver port of this PTP Instance
meanLinkDelay	is defined in 10.2.5.8
~~neighborRateRatio~~ nrrPdelay	is defined in ~~10.2.5.7~~ 11.2.13.13
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)

10.2.2.2 MDSyncReceive

10.2.2.2.2 domainNumber (UInteger8)

Change 10.2.2.2.2 as follows:

This parameter is the ~~domain-number~~domainNumber of the gPTP domain in which this structure is sent.

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

10.2.2.2.7 upstreamTxTime (UScaledNs)

Change 10.2.2.2.7 (as amended by IEEE Std 802.1ASdr-2024) as follows:

The upstreamTxTime is given by the following equation:

$$\text{upstreamTxTime} = \text{syncEventIngressTimestamp} - \frac{\text{meanLinkDelay}}{\text{neighborRateRatio}}$$

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

where

syncEventIngressTimestamp	corresponds to the receipt of the time-synchronization information at the timeReceiver port of this PTP Instance (i.e., at this PTP Port)
meanLinkDelay	is defined in 10.2.5.8
~~neighborRateRatio~~ nrrPdelay	is defined in ~~10.2.5.7~~ 11.2.13.13

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

10.2.2.3 PortSyncSync

10.2.2.3.2 domainNumber (UInteger8)

Change 10.2.2.3.2 as follows:

This parameter is the ~~domain-number~~domainNumber of the gPTP domain in which this structure is sent.

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

10.2.3 Overview of global variables used by time synchronization state machines

Change 10.2.3 (as amended by IEEE Std 802.1ASdr-2024) as follows:

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

- Per PTP Instance (i.e., per domain)
- Per PTP Instance, per PTP Port
- Instance used by the Common Mean Link Delay Service (CMLDS) (see 11.2.17) (i.e., variable is common across all ~~LinkPorts~~Link Ports)
- Instance used by CMLDS, per ~~LinkPort~~Link Port

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 ~~LinkPorts~~ <u>Link Ports</u>)	Instance used by CMLDS, per ~~LinkPort~~ <u>Link Port</u>
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
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

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~~ Link Ports)	Instance used by CMLDS, per ~~Link Port~~ Link Port
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 ¹	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 ⁺	No	~~Yes~~ No
meanLinkDelay	10.2.5.8	No	Yes ⁺	No	Yes
delayAsymmetry	10.2.5.9	No	Yes ⁺	No	Yes
computeNeighborRateRatio	10.2.5.10	No	Yes ⁺	No	Yes
computeMeanLinkDelay	10.2.5.11	No	Yes ⁺	No	Yes
portOper ²	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
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

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 ~~LinkPorts~~ <u>Link Ports</u>)	Instance used by CMLDS, per ~~LinkPort~~ <u>Link Port</u>
gPtpCapableMessageInterval	10.2.5.20	No	Yes	No	No
oldGtpCapableMessageInterval	10.2.5.21	No	Yes	No	No
currentLogGtpCapableMessageInterval	10.2.5.22	No	Yes	No	No
initialLogGtpCapableMessageInterval	10.2.5.23	No	Yes	No	No
<u>syncGrandmasterIdentity</u>	<u>10.2.4.26</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>
<u>syncStepsRemoved</u>	<u>10.2.4.27</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>
<u>driftTrackingTlvSupport</u>	<u>10.2.4.28</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>
<u>rcvdPSSyncCSS</u>	<u>10.2.4.29</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>
<u>rcvdLocalClockTickCSS</u>	<u>10.2.4.30</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>
<u>rateRatioDrift</u>	<u>10.2.4.31</u>	<u>Yes</u>	<u>No</u>	<u>No</u>	<u>No</u>

¹ ~~The instance of this variable that is per PTP Instance, per PTP Port exists only for domain 0.~~

² There is one instance of this variable per physical port, which is accessible by all PTP Ports and ~~LinkPorts~~ Link Ports associated with the physical port.

10.2.4 Per PTP Instance global variables

Change 10.2.4.20 as follows:

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

Insert 10.2.4.26, 10.2.4.27, 10.2.4.28, 10.2.4.29, 10.2.4.30, and 10.2.4.31 after 10.2.4.25 as follows:

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

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

10.2.4.28 driftTrackingTlvSupport: An indicator of whether the PTP Instance supports the Drift_Tracking TLV and the feature is enabled. The value is TRUE if the Drift_Tracking TLV is supported and the managed

object `driftTrackingTlvSupported` (see 14.7b) is TRUE. The value is FALSE if the `Drift_Tracking` TLV is not supported, or if the TLV is supported and the managed object `driftTrackingTlvSupported` is FALSE. The data type for `driftTrackingTlvSupported` is Boolean.

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

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

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

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

10.2.5 Per-port global variables

Change 10.2.5.2 as follows:

10.2.5.2 `asymmetryMeasurementMode`: A Boolean that contains the value of the managed object `asymmetryMeasurementMode` (see 14.8.45). For full-duplex IEEE 802.3 media, the value is TRUE if an asymmetry measurement is being performed for the link attached to this port and FALSE otherwise. For all other media, the value is FALSE. 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.

Change 10.2.5.7 through 10.2.5.11 as follows:

10.2.5.7 `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 port, to the frequency of the `LocalClock` entity of this time-aware system. The data type for `neighborRateRatio` is Float64. There is one instance of this variable for ~~all the domains~~ each domain, i.e., ~~all the~~ each PTP Instances (per port). ~~The variable is accessible by all the domains.~~

10.2.5.8 `meanLinkDelay`: The measured mean propagation delay (see 8.3) on the link attached to this port, relative to the `LocalClock` entity of the time-aware system at the other end of the link (i.e., expressed in the time base of the time-aware system at the other end of the link). The data type for `meanLinkDelay` 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 `meanLinkDelay` was named `neighborPropDelay` in the 2011 edition of this standard.

10.2.5.9 delayAsymmetry: The asymmetry in the propagation delay on the link attached to this port. If propagation delay asymmetry is not modeled, then delayAsymmetry is zero. The data type for delayAsymmetry is ScaledNs. There is one instance of this variable for CMLDS (see 11.2.17), and there is also one instance of this variable for each domain ~~that uses the instance-specific peer-to-peer delay mechanism~~. The instance of this variable for CMLDS is relative to the local clock. The instance of this variable for ~~each~~ domain ~~that uses the instance-specific peer-to-peer delay mechanism~~ is relative to the ~~grandmaster~~ grandmaster time base for that domain. The instance of delayAsymmetry for CMLDS is used where needed in all computations done by the CMLDS, and also by computations using the syncEgressTimestamp field of the Drift_Tracking TLV (see 11.4.4.2.3) if CMLDS is present. The instance of delayAsymmetry for a domain is used where needed in all computations done for that domain, and also by computations using the Drift_Tracking TLV if CMLDS is not present.

10.2.5.10 computeNeighborRateRatio: ~~A Boolean, set by the LinkDelayIntervalSetting state machine (see 11.2.21), that indicates whether neighborRateRatio is to be computed by this port. There is one instance of this variable for all the domains, i.e., all the PTP Instances (per port). The variable is accessible by all the domains.~~ A Boolean, set by the LinkDelayIntervalSetting state machine (see 11.2.21), that indicates whether nrrPdelay is computed for this port. There is one instance of this variable for each domain, i.e., each PTP Instance (per port), and one instance of this variable for CMLDS. If the instance of this variable for a domain is TRUE, nrrPdelay for that domain is computed.

10.2.5.11 computeMeanLinkDelay: A Boolean, set by the LinkDelayIntervalSetting state machine (see 11.2.21), that indicates whether meanLinkDelay is to be computed by this port. 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.

Change 10.2.5.13 as follows:

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

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

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

10.2.8 PortSyncSyncReceive state machine

10.2.8.1 State machine variables

Insert 10.2.8.1.5 after 10.2.8.1.4 as follows:

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

10.2.9 ClockTimeTransmitterSyncSend state machine

10.2.9.2 State machine functions

10.2.9.2.1 setPSSyncCMSS (gmRateRatio):

Insert two new NOTES in 10.2.9.2.1 after list item c) 2) as follows and renumber existing NOTE 1 and NOTE 2 to be NOTE 3 and NOTE 4, respectively:

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

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

Change list item m) of 10.2.9.2.1 as follows:

- m) domainNumber is set equal to the ~~domain-number~~domainNumber of this gPTP domain.

10.2.9.3 State diagram

Change the first paragraph of 10.2.9.3 (as amended by IEEE Std 802.1ASdr-2024) as follows:

The ClockTimeTransmitterSyncSend state machine shall implement the function specified by the state diagram in Figure 10-5, the local variables specified in 10.2.9.1, the functions specified in 10.2.9.2, the structure specified in 10.2.2.3, and the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives timeTransmitterTime and clockSourceTimeBaseIndicator from the ClockTimeTransmitterSyncReceive state machine, and phase and frequency offset between timeTransmitterTime and syncReceiptTime from the Clock~~TimeTransmitter~~SyncOffset state machine. It provides timeTransmitterTime (i.e., synchronized time) and the phase and frequency offset to the SiteSync entity via a PortSyncSync structure.

Change the title of 10.2.10 as follows:

10.2.10 Clock~~TimeTransmitter~~SyncOffset state machine

10.2.10.3 State diagram

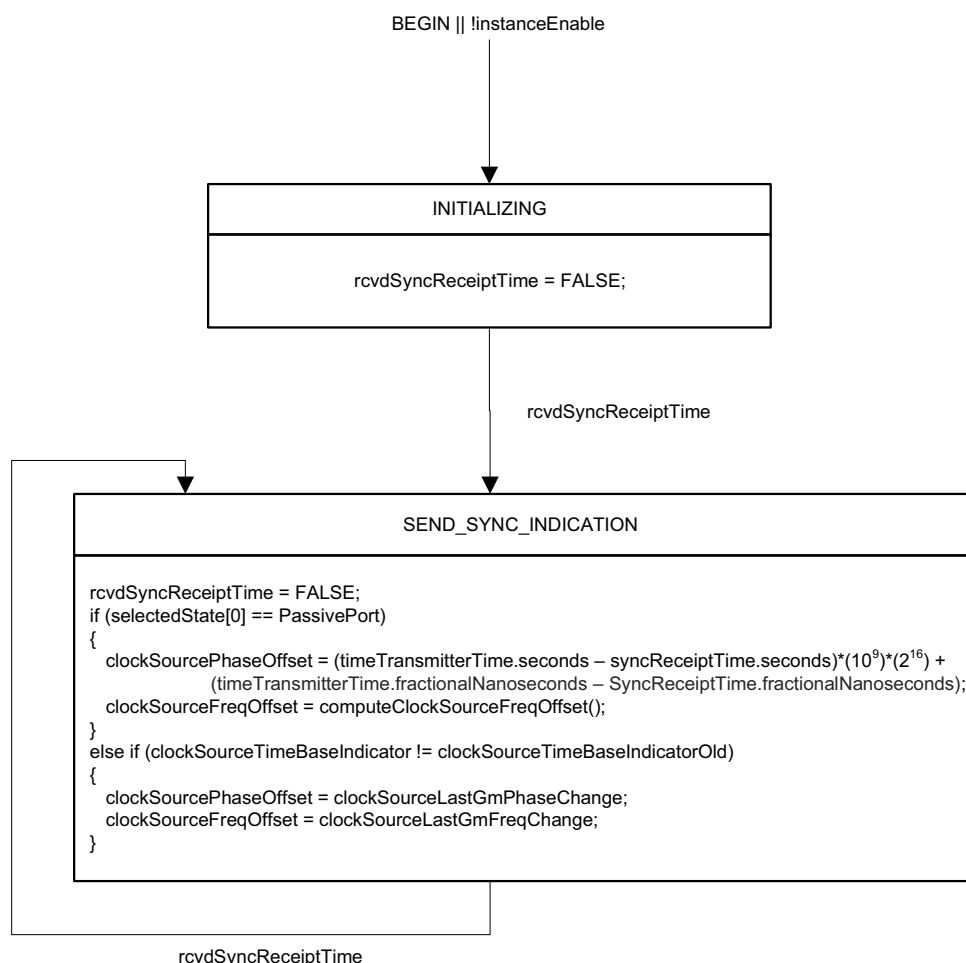
Change the text of 10.2.10.3 (as amended by IEEE Std 802.1ASdr-2024) as follows:

The Clock~~TimeTransmitter~~SyncOffset state machine shall implement the function specified by the state diagram in Figure 10-6, the local variable specified in 10.2.10.1, the function specified in 10.2.10.2, and the relevant global variables and functions specified in 10.2.4 through 10.2.6. The state machine receives syncReceiptTime from the ClockTimeReceiverSync state machine and timeTransmitterTime from the

ClockTimeTransmitterSyncReceive state machine. It computes clockSourcePhaseOffset and clockSourceFrequency offset if this PTP Instance is not currently the Grandmaster PTP Instance, i.e., if selectedState[0] is equal to PassivePort.

The Clock~~TimeTransmitter~~SyncOffset state machine is optional for PTP Instances that are not grandmaster-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 grandmaster-capable; however, any information supplied by it, via the ClockTimeTransmitterSyncSend state machine, to the SiteSyncSync state machine is not used by the SiteSyncSync state machine if the PTP Instance is not grandmaster-capable.

Replace Figure 10-6 (as amended by IEEE Std 802.1ASdr-2024) with the following figure:



NOTE—This figure differs from the 2020 edition of this standard, with Corrigendum 1 and IEEE Std 802.1ASdr-2024 applied, in that the statement:

clockSourcePhaseOffset = timeTransmitterTime.seconds – syncReceiptTime;

in the SEND_SYNC_INDICATION state is replaced by:

clockSourcePhaseOffset = (timeTransmitterTime.seconds – syncReceiptTime.seconds)*(10⁹)*(2¹⁶) +
(timeTransmitterTime.fractionalNanoseconds – SyncReceiptTime.fractionalNanoseconds);

Figure 10-6—Clock~~TimeTransmitter~~SyncOffset state machine

10.2.12 PortSyncSyncSend state machine

10.2.12.2 State machine functions

10.2.12.2.1 setMDSync():

Change list item j) in 10.2.12.2.1 as follows:

- j) domainNumber is set equal to the ~~domain number~~ domainNumber of this gPTP domain (see 8.1).

10.2.13 ClockTimeReceiverSync state machine

Change 10.2.13.1 and its subclauses as follows:

10.2.13.1 State machine variables

The following variables ~~are~~ is used in the state diagram in Figure 10-9 (in 10.2.13.3):

~~10.2.13.1.1 rcvdPSSyncCSS: A Boolean variable that notifies the current state machine when a PortSyncSync structure is received from the SiteSyncSync state machine of the SiteSync entity. This variable is reset by this state machine.~~

~~10.2.13.1.2 rcvdLocalClockTickCSS: A Boolean variable that notifies the current state machine when the LocalClock entity updates its time. This variable is reset by this state machine.~~

10.2.13.1.1 ~~10.2.13.1.3~~ rcvdPSSyncPtrCSS: A pointer to the received PortSyncSync structure.

10.2.13.3 State diagram

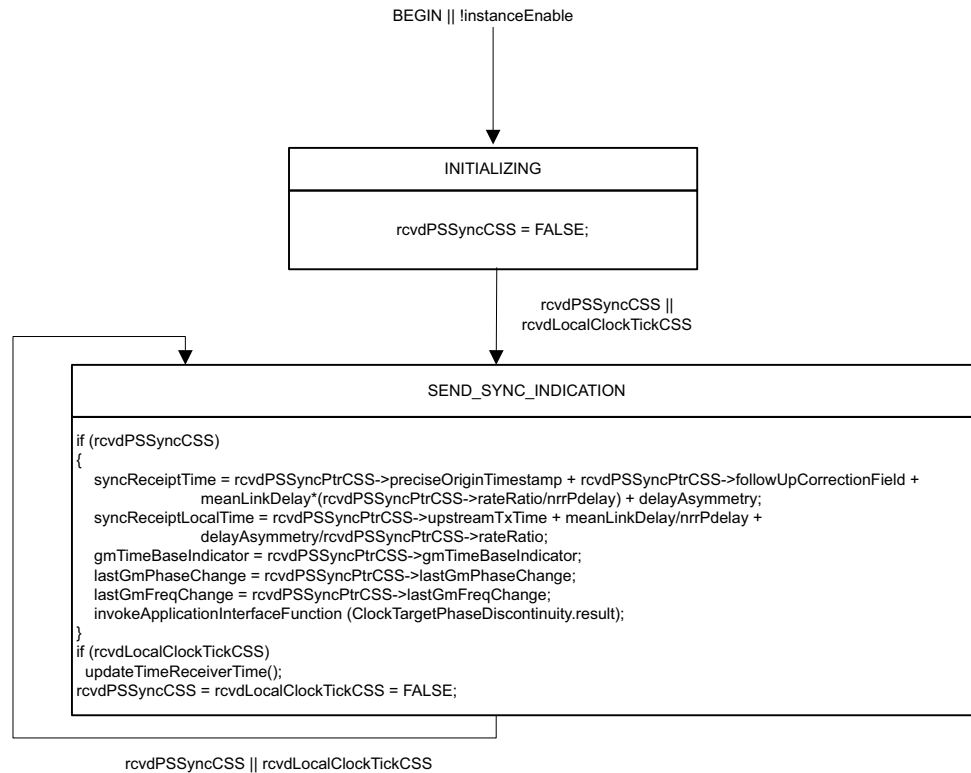
Change the text of 10.2.13.3 (as amended by IEEE Std 802.1ASdr-2024) as follows:

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

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

- a) If rcvdPSSyncPtrCSS->localPortNumber > 0, the per-PTP Port global variables of PTP Port number rcvdPSSyncPtrCSS->localPortNumber are used.
- b) If rcvdPSSyncPtrCSS->localPortNumber == 0, the values of the used per-PTP Port global variables are fixed as follows:
 - 1) meanLinkDelay = 0
 - 2) delayAsymmetry = 0
 - 3) ~~neighborRateRatio~~ nrrPdelay = 1.0

Replace Figure 10-9 (as amended by IEEE Std 802.1ASdr-2024) with the following figure:



NOTE—This figure differs from the 2020 edition of this standard, with Corrigendum 1 and IEEE Std 802.1ASdr-2024 applied, in that the variable `neighborRateRatio` is replaced by `nrrPdelay` in the first two statements of the `SEND_SYNC_INDICATION` block.

Figure 10-9—ClockTimeReceiverSync state machine

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

10.3.1 Best timeTransmitter clock selection and external port configuration overview

10.3.1.3 External port configuration overview

Change 10.3.1.3 as follows:

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

In the case of external port configuration, the time-synchronization spanning tree and the desired PTP Port states are controlled by an external entity, but the Grandmaster PTP Instance might change. If the