

This document was prepared by Yong Kim and Geoff Garner.

Revision 2, July 19, 2011

2. Normative references

Change the Text of 2., as shown

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

~~IEEE P802.11vTM (D15.0, September 2010), Draft Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications—Amendment 8: IEEE 802.11TM Wireless Network Management.²~~

IEEE Std 802.11vTM – 2011, IEEE Standard for Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: IEEE 802.11 Wireless Network Management.

IEEE Std 802.1DTM-2004, IEEE Standard for Local and metropolitan area networks—Media Access Control (MAC) Bridges.^{3,4}

IEEE Std 802.1QTM- 2005, IEEE Standard for Local and metropolitan area networks—Virtual Bridged Local Area Networks.

IEEE Std 802.1agTM-2007, IEEE Standard for Local and metropolitan area networks—Virtual Bridged Local Area Networks—Amendment 5: Connectivity Fault Management.

IEEE Std 802.3TM-2008, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area network—Specific requirements, Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.

IEEE Std 802.3avTM-2009, IEEE Standard for Information technology—Part 3: Amendment 1: Physical Layer Specifications and Management Parameters for 10 Gb/s Passive Optical Networks.

~~²IEEE P802.11v/D16 (November 2010) was approved by the IEEE SA Standards Board on 2 February 2011. It was published as IEEE Std 802.11v-2011 on 9 February 2011 and is available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (<http://standards.ieee.org>).~~

³IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (<http://standards.ieee.org>).

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

IEEE Std 802.11™-2007, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

IEEE Std 1588™-2008, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.

IETF RFC 3410 (December 2002), Introduction and Applicability Statements for Internet Standard Management Framework, Case, J., Mundy, R., Partain, D, and Stewart, B.⁵

ITU-T Recommendation G.9960 (ex. G.hn), Unified high-speed wire-line based home networking transceivers—System architecture and physical layer specification, June 2010.⁶

ITU-T Recommendation G.9961, Data link layer (DLL) for unified high-speed wire-line based home networking transceivers, June 2010.

ITU-T Recommendation G.984.3, Amendment 2 (2009-11) Gigabit-capable Passive Optical Networks (G-PON): Transmission convergence layer specification—Time-of-day distribution and maintenance updates and clarifications, November 2009.

MoCA® MAC/PHY Specification v2.0, MoCA-M/P-SPEC-V2.0-20100507, Multimedia over Coax Alliance (MoCA).⁷

8. IEEE 802.1AS concepts and terminology

8.6 Time-aware system characterization

8.6.2 Time-aware system attributes

8.6.2.4 offsetScaledLogVariance

Change the Text of 8.6.2.4, as shown

The offsetScaledLogVariance is scaled, offset representation of an estimate of the PTP variance. The PTP variance characterizes the precision and frequency stability of the ClockMaster. The PTP variance is the square of PTPDEV (see B.1.3.2).

The value shall be selected as follows:

⁵IETF RFCs are available from the Internet Engineering Task Force Web site at <http://www.ietf.org/rfc.html>.

⁶ITU-T publications are available from the International Telecommunications Union, Place des Nations, CH-1211, Geneva 20, Switzerland/Suisse (<http://www.itu.int/>).

⁷MoCA specifications are available from the Multimedia over Coax Alliance at <http://www.mocalliance.org/specs>.

- a) `offsetScaledLogVariance` is set to the value that reflects the combination of the `LocalClock` and `ClockSource` entities; else
- b) if the value that reflects these entities is not specified or not known, `offsetScaledLogVariance` is set to ~~16640-17258~~ (~~4100₁₆436A₁₆~~). This value corresponds to the value of `PTPDEV` for observation interval equal to the default Sync message transmission interval (i.e., observation interval of 0.125 s, see 11.5.2.3 and B.1.3.2).

See 7.6.3 of IEEE Std 1588-2008 for more detailed description of PTP variance and `offsetScaledLogVariance` (7.6.3.3 of IEEE Std 1588-2008 provides a detailed description of the computation of `offsetScaledLogVariance` from PTP variance, along with an example).

10. Media-independent layer specification

10.2 Time-synchronization state machines

10.2.10 `ClockMasterSyncReceive` state machine

10.2.10.2 State machine functions

10.2.10.2.2 `updateMasterTime()`

Change the Text of 10.2.10.2.2, as shown

`updateMasterTime()`: updates the global variable `masterTime` (see 10.2.3.21), based on information received from the `ClockSource` and `LocalClock` entities. It is the responsibility of the application to filter master times appropriately. As one example, `masterTime` can be set equal to the `sourceTime` member of the `ClockSourceTime.invoke` function when this function is received, and can be incremented by `localClockTickInterval` (see 10.2.3.18) ~~divided~~multiplied by `gmRateRatio` (see 10.2.3.14) when `revdLocalClockTick` is `TRUE`.

10.2.12 `ClockSlaveSync` state machine

10.2.12.2 State machine functions

10.2.12.2.1 `updateSlaveTime()`

Change the Text of 10.2.12.2.1, as shown

`updateSlaveTime()`: updates the global variable `clockSlaveTime` (see 10.2.3.3), based on information received from the `SiteSync` and `LocalClock` entities. It is the responsibility of the application to filter slave times appropriately (see B.3 and B.4 for examples). As one example, `clockSlaveTime` can be incremented by `localClockTickInterval` (see 10.2.3.18) ~~divided~~multiplied by `rateRatio` member of the received `PortSyncSync` structure. If no time-aware system

is grandmaster-capable, i.e., gmPresent is FALSE, then clockSlaveTime is set to the time provided by the LocalClock. This function is invoked when revdLocalClockTick is TRUE.

10.3 Best master clock selection and announce interval setting state machines

10.3.8 Per-time-aware system global variables

10.3.8.1 reselect

Change the Text of 10.3.8.1, as shown

reselect: a Boolean array of length numberPorts (see 8.6.1). Setting reselect[j], where $0 \leq j < \text{numberPorts}$, to TRUE causes the ROLE_SELECTION block of the PortRoleSelection state machine (see 10.3.12) to be re-entered, which in turn causes the port role of each port of the time-aware system to be updated (via the function [updateRolesTree\(\)](#), see 10.3.12.1.4).

10.3.12 PortRoleSelection state machine

10.3.12.1 State machine functions

10.3.12.1.4 updtRolesTree ()

Change the Text of 10.3.12.1.4, as shown

updtRolesTree(): performs the following operations (see 10.3.4 and 10.3.5 for details on the priority vectors):

- a) Computes the gmPathPriorityVector for each port that has a portPriorityVector and for which neither announce receipt timeout nor, if gmPresent is TRUE, sync receipt timeout have occurred,
- b) Saves gmPriority (see 10.3.8.19) in lastGmPriority (see 10.3.8.20), computes the gmPriorityVector for the time-aware system and saves it in gmPriority, chosen as the best of the set consisting of the systemPriorityVector (for this time-aware system) and the gmPathPriorityVector for each port for which the clockIdentity of the master port is not equal to thisClock (see 10.2.3.22),
- c) Sets the per-time-aware system global variables leap61, leap59, currentUtcOffsetValid, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource as follows:
 - 1) If the gmPriorityVector was set to the gmPathPriorityVector of one of the ports, then leap61, leap59, currentUtcOffsetValid, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource are set to annLeap61, annLeap59, annCurrentUtcOffsetValid, annTimeTraceable, annFrequencyTraceable, annCurrentUtcOffset, and annTimeSource, respectively, for that port.
 - 2) If the gmPriorityVector was set to the systemPriorityVector, then leap61, leap59, currentUtcOffsetValid, timeTraceable, frequencyTraceable, currentUtcOffset, and timeSource are set to sysLeap61, sysLeap59, sysCurrentUtcOffsetValid,

sysTimeTraceable, sysFrequencyTraceable, sysCurrentUtcOffset, and sysTimeSource, respectively.

- d) Computes the first three components ~~and the clockIdentity of the fourth component~~ of the masterPriorityVector for each port_i ~~(i.e., everything except the portNumber of the fourth component)~~
- e) Computes masterStepsRemoved, which is equal to:
 - 1) messageStepsRemoved (see 10.3.9.7) for the port associated with the gmPriorityVector, incremented by 1, if the gmPriorityVector is not the systemPriorityVector, or
 - 2) 0 if the gmPriorityVector is the systemPriorityVector,
- f) assigns the port role for port j and sets selectedRole[j] equal to this port role, as follows, for j = 1, 2, ..., numberPorts:
 - 3) If the port is disabled (infoIs == Disabled), selectedRole[j] is set to DisabledPort.
 - 4) If announce receipt timeout, or sync receipt timeout with gmPresent set to TRUE, have occurred (infoIs = Aged), updInfo is set to TRUE and selectedRole[j] is set to MasterPort.
 - 5) If the portPriorityVector was derived from another port on the time-aware system or from the time-aware system itself as the root (infoIs == Mine), selectedRole[j] is set to MasterPort. In addition, updInfo is set to TRUE if the portPriorityVector differs from the masterPriorityVector or portStepsRemoved differs from masterStepsRemoved.
 - 6) If the portPriorityVector was received in an Announce message and announce receipt timeout, or sync receipt timeout with gmPresent TRUE, have not occurred (infoIs == Received), and the gmPriorityVector is now derived from the portPriorityVector, selectedRole[j] is set to SlavePort and updInfo is set to FALSE.
 - 7) If the portPriorityVector was received in an Announce message and announce receipt timeout, or sync receipt timeout with gmPresent TRUE, have not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, the masterPriorityVector is not higher than the portPriorityVector, and the sourcePortIdentity component of the portPriorityVector does not reflect another port on the time-aware system, selectedRole[j] is set to PassivePort and updInfo is set to FALSE.
 - 8) If the portPriorityVector was received in an Announce message and announce receipt timeout, or sync receipt timeout with gmPresent TRUE, have not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, the masterPriorityVector is not higher than the portPriorityVector, and the sourcePortIdentity component of the portPriorityVector does reflect another port on the time-aware system, selectedRole[j] set to PassivePort and updInfo is set to FALSE.
 - 9) If the portPriorityVector was received in an Announce message and announce receipt timeout, or sync receipt timeout with gmPresent TRUE, have not occurred (infoIs == Received), the gmPriorityVector is not now derived from the portPriorityVector, and the masterPriorityVector is better than the portPriorityVector, selectedRole[j] set to MasterPort and updInfo is set to TRUE.
- g) Updates gmPresent as follows:
 - 10) gmPresent is set to TRUE if the priority1 field of the rootSystemIdentity of the gmPriorityVector is less than 255.
 - 11) gmPresent is set to FALSE if the priority1 field of the rootSystemIdentity of the

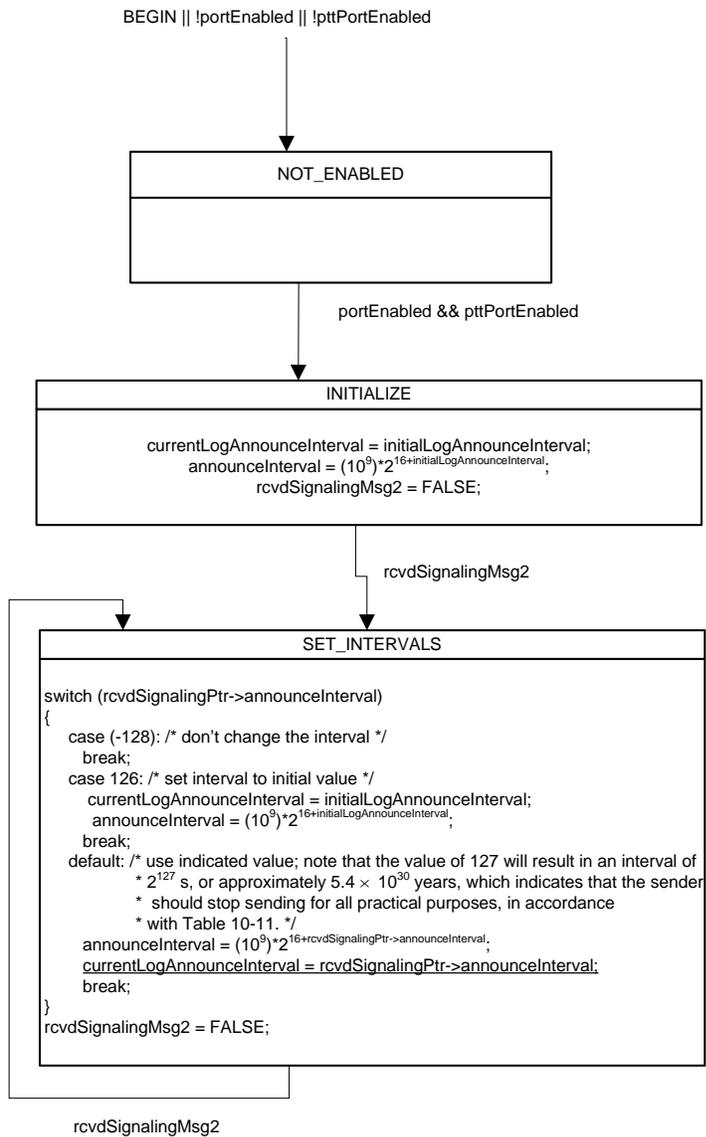
gmPriorityVector is equal to 255.

- h) Assigns the port role for port 0, and sets selectedRole[0], as follows:
 - 12) if selectedRole[j] is set to SlavePort for any port with portNumber j, j = 1, 2, ..., numberPorts, selectedRole[0] is set to PassivePort.
 - 13) if selectedRole[j] is not set to SlavePort for any port with portNumber j, j = 1, 2, ..., numberPorts, selectedRole[0] is set to SlavePort.
- i) If the clockIdentity member of the systemIdentity (see 10.3.2) member of gmPriority (see 10.3.8.19) is equal to thisClock (see 10.2.3.22), i.e., if the current time-aware system is the grandmaster, the pathTrace array is set to contain the single element thisClock (see 10.2.3.22).

10.3.14 AnnounceIntervalSetting state machine

10.3.14.2 State diagram

Change the Figure of 10-16, as shown



11. Media-dependent layer specification for full-duplex, point-to-point links

11.1 Overview

11.1.3 Transport of time-synchronization information

Change the Text of 11.1.3, as shown

The transport of time-synchronization information by a time-aware system, using Sync and Follow_Up messages, is illustrated in Figure 11-2. The mechanism is mathematically equivalent to the mechanism described in IEEE Std 1588-2008 for a two-step,¹⁵ peer-to-peer transparent clock that is synchronized (see 11.4.5.1, 11.5.1, and 11.5.2.2 of IEEE Std 1588-2008). However, as will be seen shortly, the processes of transporting synchronization by a two-step,

peer-to-peer transparent clock that is syntonized and by a two-step boundary clock are mathematically and functionally equivalent.⁸ The main functional difference between the two types of clocks is that the boundary clock participates in best master selection and invokes the BMCA, while the peer-to-peer transparent clock does not participate in best master selection and does not invoke the BMCA (and implementations of the two types of clocks can be different).

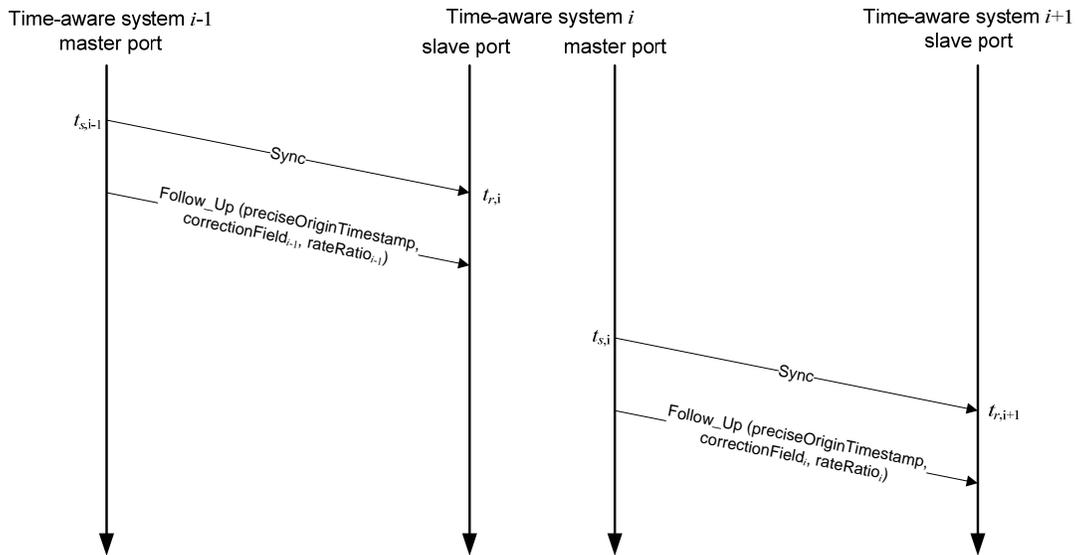


Figure 11-2 – Transport of time-synchronization information

Figure 11-2 shows three adjacent time-aware systems, indexed $i-1$, i , and $i+1$. Synchronization is transported from time-aware system $i-1$ to time-aware system i , and then to time-aware system $i+1$. Time-aware system $i-1$ send a Sync message to time-aware system i at time $t_{s,i-1}$, relative to the LocalClock entity of time-aware system $i-1$. At a later time, time-aware system $i-1$ sends an associated Follow_Up message to time-aware system i , which contains a preciseOriginTimestamp, correctionField $_{i-1}$, and rateRatio $_{i-1}$. The preciseOriginTimestamp contains the time of the grandmaster when it originally sent this synchronization information. It is not indexed here because it normally does not change as the Sync and Follow_Up messages traverse the network. The quantity correctionField $_{i-1}$ contains the difference between the synchronized time when the Sync message is sent (i.e., the synchronized time that corresponds to the local time $t_{s,i-1}$) and the preciseOriginTimestamp. The sum of preciseOriginTimestamp and correctionField $_{i-1}$ gives the synchronized time that corresponds to $t_{s,i-1}$. The quantity rateRatio $_{i-1}$ is the ratio of the grandmaster frequency to the frequency of the LocalClock entity of time-aware system $i-1$.

Time-aware system i receives the Sync message from time-aware system $i-1$ at time $t_{r,i}$ relative to its LocalClock entity. It timestamps the receipt of the Sync message, and the timestamp value is $t_{r,i}$. It receives the associated Follow_Up message some time later.

Time-aware system i will eventually send a new Sync message at time $t_{s,i}$, relative to its LocalClock entity. It will have to compute correctionField $_i$, i.e., the difference between the synchronized time that corresponds to $t_{s,i}$ and the preciseOriginTimestamp. To do this, it must compute the value of the time interval between $t_{s,i-1}$ and $t_{s,i}$, expressed in the grandmaster time base. This interval is equal to the sum of the following quantities:

⁸ The same mathematical and functional equivalence exists for one-step boundary and syntonized peer-to-peer transparent clocks. One-step clocks are not discussed here because time-aware systems described in this standard are two-step devices from the standpoint of IEEE Std 1588-2008.

- a) The propagation delay on the link between time-aware systems $i-1$ and i , expressed in the grandmaster time base, and
- b) The difference between $t_{s,i}$ and $t_{r,i}$ (i.e., the residence time), expressed in the grandmaster time base.

The mean propagation delay on the link between time-aware systems $i-1$ and i , relative to the LocalClock entity of time-aware system $i-1$, is equal to neighborPropDelay (see 10.2.4.7). This must be ~~divided~~ multiplied by rateRatio $_{i-1}$ to express it in the grandmaster time base. The total propagation delay is equal to the mean propagation delay plus the quantity delayAsymmetry (see 8.3 and 10.2.4.8); delayAsymmetry is already expressed in the grandmaster time base. The residence time, $t_{s,i}-t_{r,i}$, must be multiplied by rateRatio $_i$ to express it in the grandmaster time base.

The preceding computation is organized slightly differently in the state machines of 11.2.13 and 11.2.14. Rather than explicitly expressing the link propagation delay in the grandmaster time base, the local time at time-aware system i that corresponds to $t_{s,i-1}$ is computed; this is the upstreamTxTime member of the MDSyncReceive structure (see 10.2.2.2.6; recall that $t_{s,i-1}$ is relative to the LocalClock entity of time-aware system $i-1$). upstreamTxTime is equal to the quantity $t_{r,i}$ minus the link propagation delay expressed relative to the LocalClock entity of time-aware system i . The link propagation delay expressed relative to the LocalClock entity of time-aware system i is equal to the sum of the following:

- c) The quantity neighborPropDelay (see 10.2.4.7) divided by neighborRateRatio (see 10.2.4.6), and
- d) The quantity delayAsymmetry (see 10.2.4.8) divided by rateRatio $_i$.

The division of delayAsymmetry by rateRatio $_i$ is performed after rateRatio $_i$ has been updated, as described shortly. The computation of upstreamTxTime is done by the MDSyncReceiveSM state machine in the function setMDSyncReceive() (see 11.2.13.2.1). When time-aware system i sends a Sync message to time-aware system $i+1$, it computes the sum of the link propagation delay and residence time, expressed in the grandmaster time base, as:

- e) The quantity $(t_{s,i} - \text{upstreamTxTime})(\text{rateRatio}_i)$.

As in item The quantity delayAsymmetry (see 10.) above, this computation is performed after rateRatio $_i$ has been updated, as described shortly. The quantity of item The quantity () is added to correctionField $_{i-1}$ to obtain correctionField $_i$. The computation of item The quantity () and correctionField $_i$ is done by the MDSyncSendSM state machine in the function setFollowUp() (see 11.2.14.2.3). The quantity correctionField $_i$ is inserted in the Follow_Up message sent by time-aware system i .

Note that the difference between mean propagation delay relative to the grandmaster time base and relative to the time bases of the time-aware system at the other end of the attached link or of the current time-aware system is usually negligible. To see this, note that the former can be obtained from the latter by multiplying the latter by the ratio of the grandmaster frequency to the frequency of the LocalClock entity of the time-aware system at the other end of the link attached to this port. This ratio differs from 1 by 200 ppm or less. For example, for a worst-case frequency offset of the LocalClock entity of the time-aware system at the other end of the link, relative to the grandmaster, of 200 ppm, and a measured propagation time of 100 ns, the difference in D relative to the two time bases is 20 ps. The corresponding difference for link delay asymmetry in this example is also negligible because the magnitude of the link delay asymmetry is of the same order of magnitude as the mean propagation time, or less. However, the difference is usually not negligible for residence time, because residence time can be much larger (see B.2.2).

It was previously indicated that the processes of transporting synchronization by a two-step, peer-to-peer transparent clock that is syntonized and by a two-step boundary clock are mathematically and functionally equivalent. This is because the computations described above compute the synchronized time when the Sync message is sent by the time-aware system. The same computations are done if time-aware system i sends a Sync message without having received a new Sync message, i.e., if Sync receipt timeout occurs (see 10.6.3.1). In this case, time-aware system i uses the most recently received time-synchronization information from time-aware system $i-1$, which would be prior to time-aware system i having sent its most recent Sync message. The synchronized time corresponding to the sending of a Sync message is equal to the sum of the preciseOriginTimestamp and correctionField. Normally a

boundary clock places this entire value, except for any sub-nanosecond portion, in the `preciseOriginTimestamp`, while a transparent clock retains the `preciseOriginTimestamp` and updates the correction field. However, the sum of the two fields is equal to the synchronized time when the Sync message is sent in both cases.

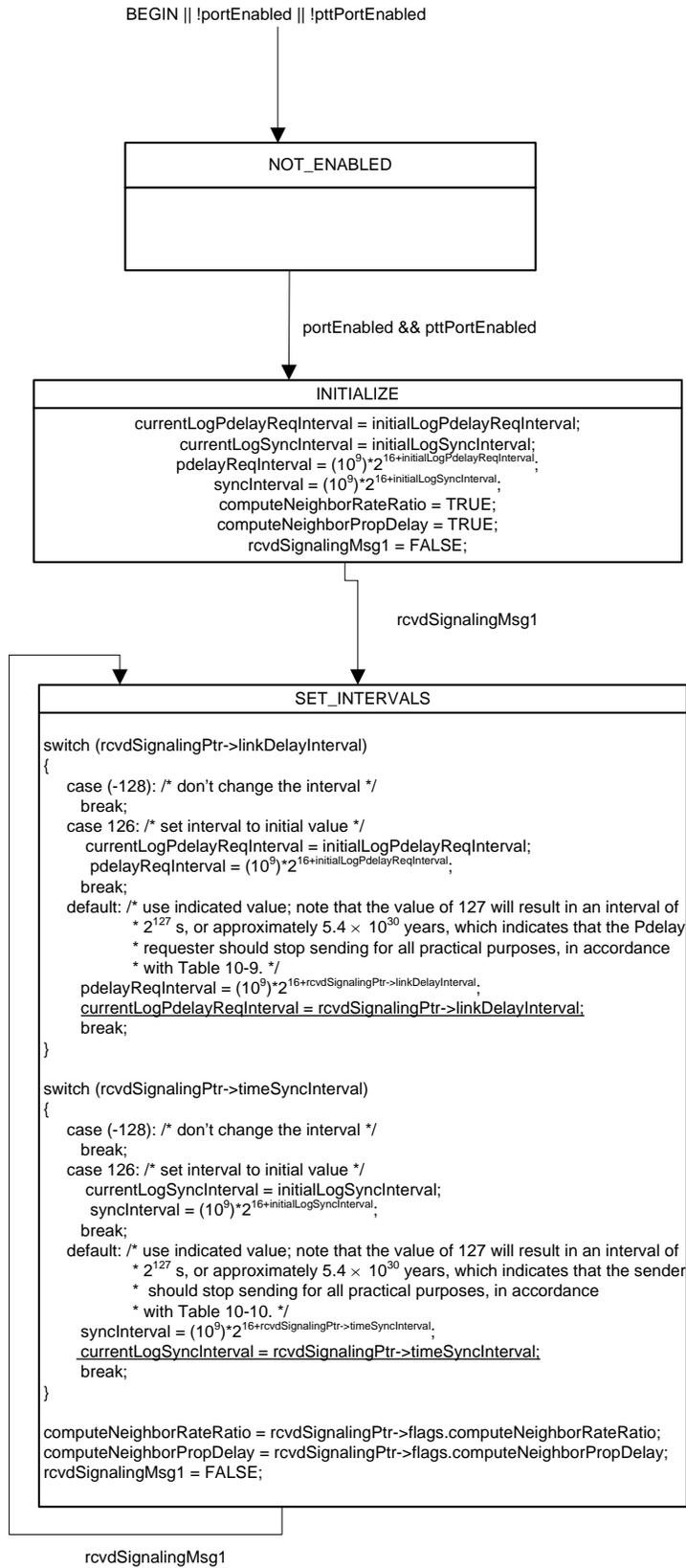
The ratio of the grandmaster frequency to the frequency of the LocalClock entity at time-aware system i , `rateRatioi`, is equal to the same quantity at time-aware system $i-1$, `rateRatioi-1`, multiplied by the ratio of the frequency of the LocalClock entity at time-aware system $i-1$ to the frequency of the LocalClock entity at time-aware system i , `neighborRateRatio` (see 10.2.4.6). If `neighborRateRatio` is sufficiently small, this is approximately equal to the sum of `rateRatioi-1` and the quantity `neighborRateRatio-1`, which is the frequency offset of time-aware system $i-1$ relative to time-aware system i . This computation is done by the `PortSyncSyncReceive` state machine (see 10.2.7).

11.2 State machines for MD entity specific to full-duplex, point-to-point links

11.2.17 LinkDelaySyncIntervalSetting state machine

11.2.17.2 State Diagram

Change the Figure of 11-10, as shown



15. Managed object definitions

Change the Text of 15.5, ieee8021AsTimeSyncMib as shown

```
ieee8021AsTimeSyncMib MODULE-IDENTITY
  LAST-UPDATED "201011110000Z201105100000Z" -- November 11, 2010May 10,
2011
  ORGANIZATION "IEEE 802.1 Working Group"
  CONTACT-INFO
    "WG-URL: http://www.ieee802.org/1/index.html
    WG-EMail: STDS-802-1@IEEE.ORG

    Contact: Geoffrey M. Garner
    Postal: 196 Ambassador Drive
           Red Bank, NJ 07701
           USA
    E-mail: gmgarner@alum.mit.edu"

  DESCRIPTION
    "The Management Information Base module for
    IEEE 802.1AS time synchronization protocol."

  REVISION "201011110000Z201105100000Z" -- November 11, 2010May 10, 2011

  DESCRIPTION
    "Published as part of IEEE Std 802.1AS

    Copyright (C) IEEE (2011)."
```

Change the Text of 15.5, ieee8021AsParentDSGrandmasterPriority1 as shown

```
ieee8021AsParentDSGrandmasterPriority1 OBJECT-TYPE
  SYNTAX      Unsigned32(0..255)
  MAX-ACCESS  read-writeonly
  STATUS      current
  DESCRIPTION
    "Grandmaster's most-significant priority declaration in
    the execution of the best master clock algorithm.
    Lower values take precedence. The default value is set
    to ieee8021AsDefaultDSPriority1."
  REFERENCE   "14.4.7"
  ::= { ieee8021AsParentDS 8 }
```

Change the Text of 15.5, ieee8021AsParentDSGrandmasterPriority2 as shown

```
ieee8021AsParentDSGrandmasterPriority2 OBJECT-TYPE
```

```

SYNTAX      Unsigned32(0..255)
MAX-ACCESS  read-writeonly
STATUS      current
DESCRIPTION
    "Grandmaster's least-significant priority declaration in
    the execution of the best master clock algorithm.
    Lower values take precedence. The default value is set to
    ieee8021AsDefaultDSDSPriority2."
REFERENCE   "14.4.8"
 ::= { ieee8021AsParentDS 9 }

```

Change the Text of 15.5, ieee8021AsTimePropertiesDSTimeSource as shown

```

ieee8021AsTimePropertiesDSTimeSource OBJECT-TYPE
    SYNTAX      IEEE8021ASTimeSourceValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value is timeSource for the current grandmaster
        (see 14.2.15). It is equal to the global variable
        timeTraceable (see 10.3.8.10).

        Indicates the source of time used by the grandmaster
        clock.
        The default value is set to
        ieee8021AsDefaultDSTimeSource."
```

Change the Text of 15.5, ieee8021AsPortDSAnnounceReceiptTimeout as shown

```

ieee8021AsPortDSAnnounceReceiptTimeout OBJECT-TYPE
    SYNTAX      Unsigned32(02..255)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION

        "The value of this attribute tells a slave port the number
        of sync intervals to wait without receiving
        synchronization information, before assuming that the
        master is no longer transmitting synchronization
        information, and that the BMC algorithm needs to be
        run, if appropriate. The condition of the slave port
        not receiving synchronization information for
        syncReceiptTimeout sync intervals is referred to as
        'sync receipt timeout'.

        "The value is the number of Announce message
        transmission intervals that a slave port waits without
        receiving an Announce message, before assuming

```

~~that the master is no longer transmitting
Announce messages, and that the BMC Algorithm needs to
be run, if appropriate. The condition of the slave port
not receiving an Announce message for
announceReceiptTimeout announce intervals is referred to as
'announce receipt timeout'.~~

The default value is ~~2~~3 (see 10.6.3.2).

The contents of this variable SHALL be maintained across a restart of the system.

Change the Text of 15.5, ieee8021AsPortDSSyncReceiptTimeout as shown

ieee8021AsPortDSSyncReceiptTimeout OBJECT-TYPE

SYNTAX Unsigned32(~~0~~2..255)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

~~"The value is the number of time synchronization
transmission intervals that a slave port waits without
receiving synchronization information, before assuming
that the master is no longer transmitting
synchronization information and that the BMCA needs to
be run, if appropriate.~~

"The value of this attribute tells a slave port the number
of sync intervals to wait without receiving
synchronization information, before assuming that the
master is no longer transmitting synchronization
information, and that the BMC algorithm needs to be
run, if appropriate. The condition of the slave port
not receiving synchronization information for
syncReceiptTimeout sync intervals is referred to as
'sync receipt timeout'.

The ~~initialziation~~initialization value is 3 (see 10.6.3.1).

The contents of this variable SHALL be maintained across a restart of the system.

Annex B

(normative)

Performance requirements

B.1 LocalClock requirements

B.1.3 Noise generation

B.1.3.2 Wander generation

Change the Text of B.1.3.2, as shown

Wander generation is specified using the Time Deviation (TDEV) parameter. The corresponding values of the Allan Deviation (ADEV) and PTP Deviation (PTPDEV) are given for information; the former is also useful in describing the wander generation of clocks and oscillators, and the latter is related to the offsetScaledLogVariance attribute (see 8.6.2.4). Information on ADEV and TDEV are contained in ITU-T G.810 [B13] and IEEE Std 1139TM-1999 [B6]. Information on ADEV and PTPDEV are contained in 7.6.3 of IEEE Std 1588-2008.

TDEV, denoted $\sigma_x(\tau)$, is estimated from a set of measurements, as shown in Equation (B.1):

$$\sigma_x(\tau) = \sqrt{\frac{1}{6n^2(N-3n+1)} \sum_{j=1}^{N-3n+1} \left[\sum_{i=j}^{n+j-1} (x_{i+2n} - 2x_{i+n} + x_i) \right]^2}, \quad n = 1, 2, \dots, \left\lfloor \frac{N}{3} \right\rfloor$$

where

$\tau = n\tau_0 =$ observation interval

$\tau_0 =$ sampling interval

$N =$ total number of samples [$(N-1)\tau_0 =$ measurement interval]

$\lfloor y \rfloor$ denotes the floor function, i.e., the greatest integer less than or equal to y

$x_i =$ measured phase (time) error at the i^{th} sampling time [the units of x_i and $\sigma_x(\tau)$ are the same]

ADEV, denoted $\sigma_y(\tau)$, is estimated from a set of measurements, as shown in Equation (B.2):

$$\sigma_y(\tau) = \sqrt{\frac{1}{2n^2\tau_0^2(N-2n)} \sum_{i=1}^{N-2n} (x_{i+2n} - 2x_{i+n} + x_i)^2}, \quad n = 1, 2, \dots, \left\lfloor \frac{N-1}{2} \right\rfloor$$

where the notation is the same as defined above for TDEV.

PTPDEV, denoted $\sigma_{PTP}(\tau)$, is estimated from a set of measurements, as shown in Equation (B.3):

$$\sigma_{PTP}(\tau) = \sqrt{\frac{1}{6n^2(N-2n)} \sum_{i=1}^{N-2n} (x_{i+2n} - 2x_{i+n} + x_i)^2}, \quad n = 1, 2, \dots, \left\lfloor \frac{N-1}{2} \right\rfloor$$

$$\sigma_{PTP}(\tau) = \sqrt{\frac{1}{6(N-2n)} \sum_{i=1}^{N-2n} (x_{i+2n} - 2x_{i+n} + x_i)^2} \quad , \quad n = 1, 2, \dots, \left\lfloor \frac{N-1}{2} \right\rfloor$$

where the notation is the same as defined above for TDEV.

TDEV, ADEV, and PTPDEV are second-order statistics on the phase error. All three statistics are functions of second differences of the phase error. This means that these statistics are not affected by a constant frequency offset. This behavior is desired, because these statistics are used here to constrain noise generation.

TDEV for the LocalClock entity shall not exceed the mask of Table B.1 and Figure B.1, when measured using

- a) A measurement interval that is at least 120 s (i.e., at least 12 times the longest observation interval),
- b) A low-pass filter with 3 dB bandwidth of 10 Hz, first-order characteristic, and 20 dB/decade roll-off, and
- c) A sampling interval t_0 that does not exceed 1/30 s.

Table B.1 – Wander generation TDEV requirement for LocalClock entity

TDEV limit	Observation interval τ
No requirement	$\tau < 0.05$ s
5.0τ ns	$0.05 \leq \tau \leq 10$ s
No requirement	$\tau > 10$ s

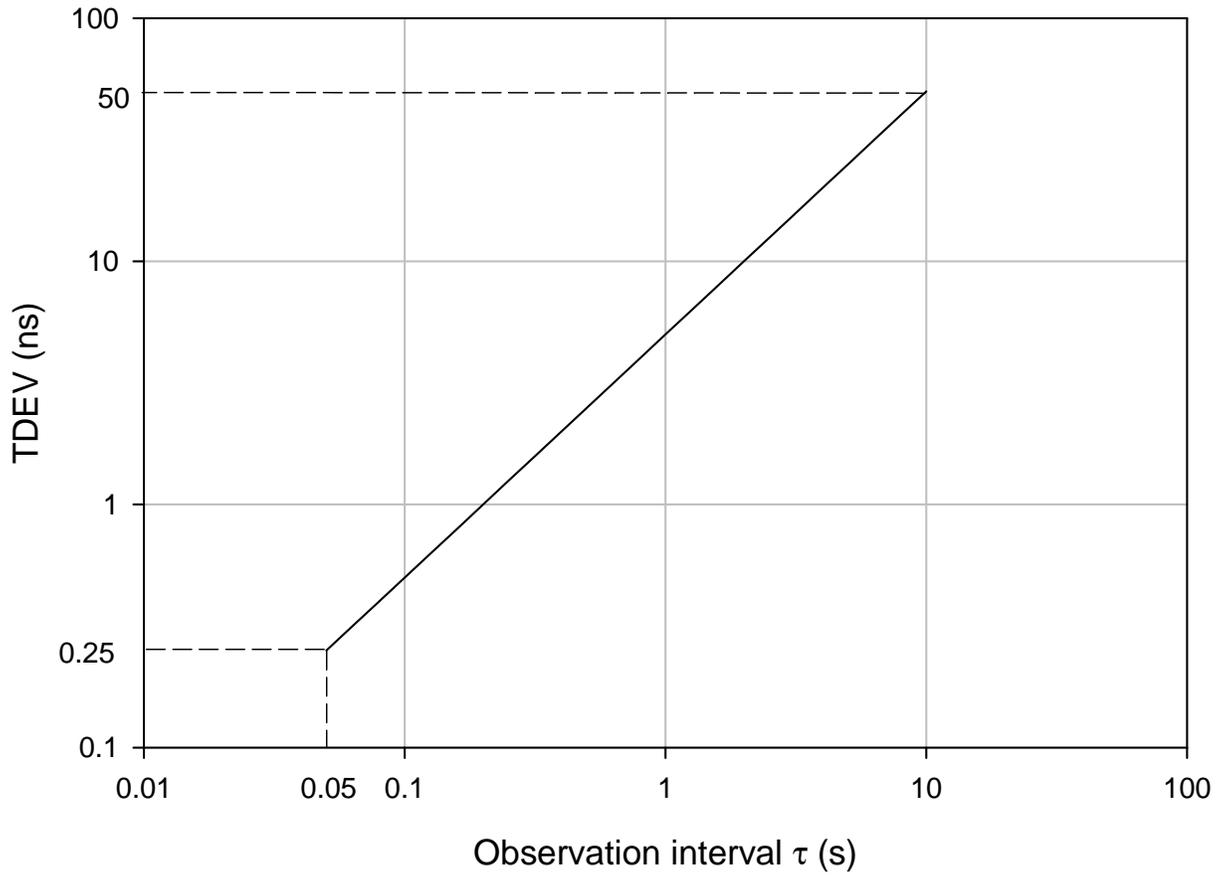


Figure B.1 – Wander generation (TDEV) requirement for LocalClock entity

The ADEV limit that corresponds to the TDEV requirement of Table B.1 and Figure B.1 is shown in Table B.2 and Figure B.2, respectively.

Table B.2 – ADEV limit corresponding to wander generation requirement of Table B.1

ADEV limit	Observation interval τ
No requirement	$\tau < 0.05$ s
1.054×10^{-8}	$0.05 \leq \tau \leq 10$ s
No requirement	$\tau > 10$ s

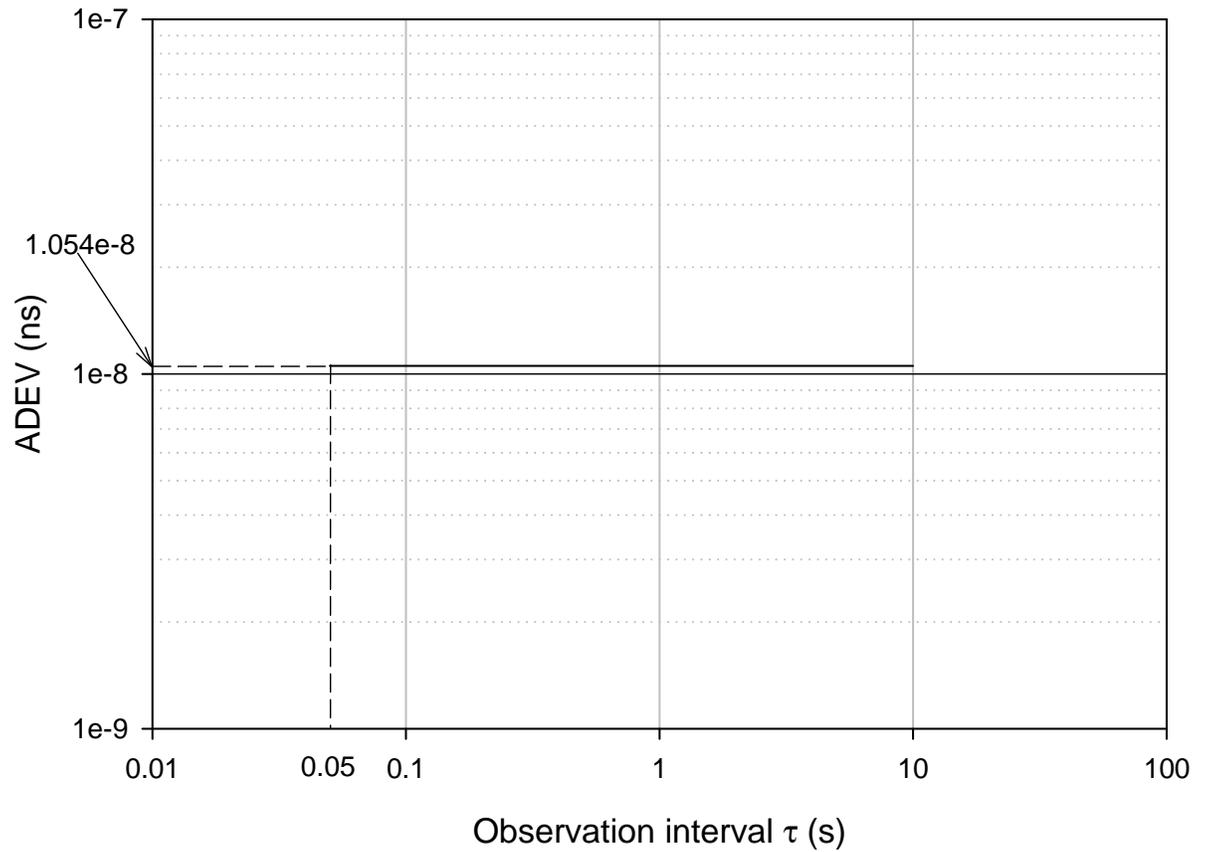


Figure B.2 – ADEV limit corresponding to wander generation requirement of Figure B.1

The PTPDEV limit that corresponds to the TDEV requirement of Table B.1 and Figure B.1 is shown in Table B.3 and Figure B.3, respect, respectively.

Table B.3 – PTPDEV limit corresponding to wander generation requirement of Table B.1

ADEV limit	Observation interval τ
No requirement	$\tau < 0.05$ s
6.08τ ns	$0.05 \leq \tau \leq 10$ s
No requirement	$\tau > 10$ s

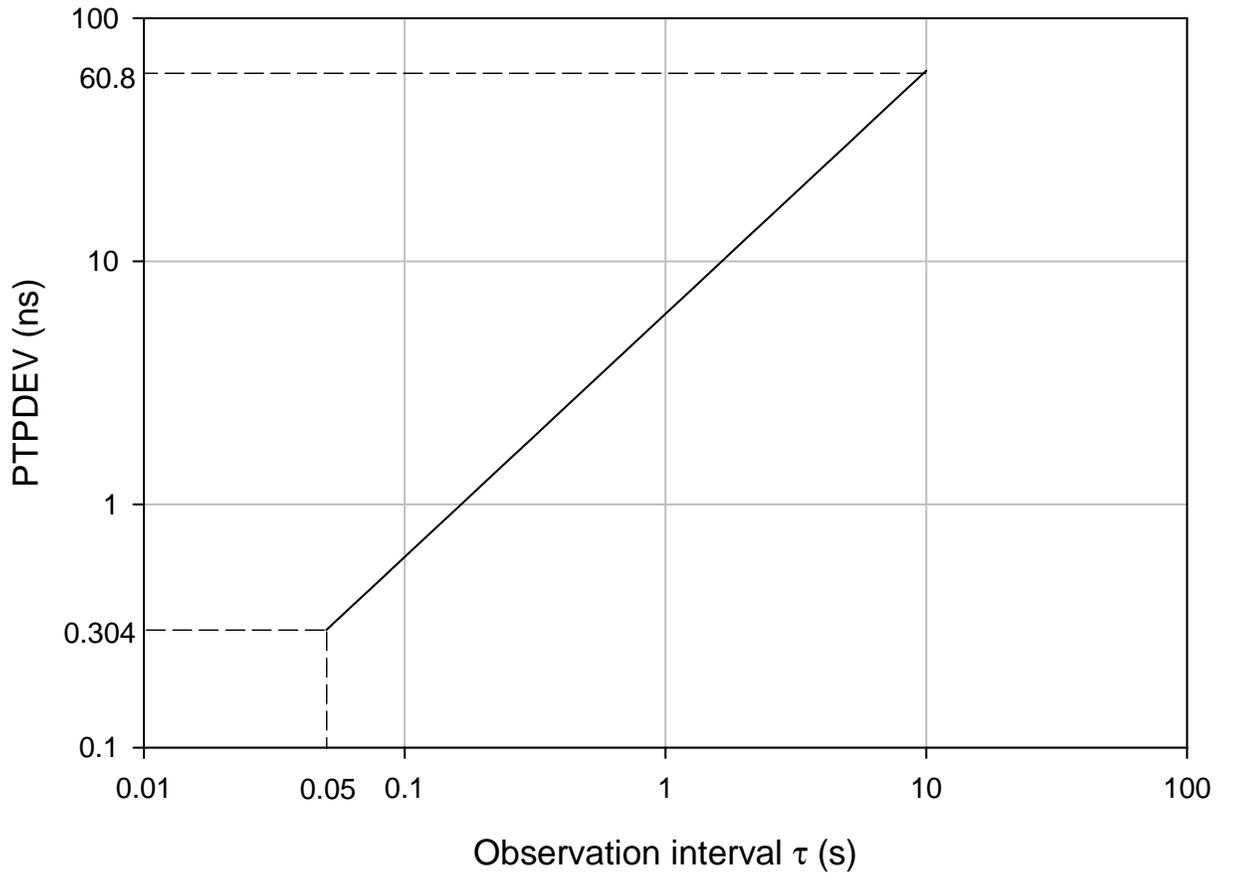


Figure B.3 – PTPDEV limit corresponding to wander generation requirement of Figure B.1