Draft for Local and Metropolitan Area Networks-Timing and Synchronization for Time-Sensitive Applications

Amendment: Fault-Tolerant Timing with Time Integrity

Prepared by: **Richard Tse, Microchip Technology**

Abdul Jabbar, GE Research

Stephen Bush, GE Research

Tim Wood, GE Research

Rob Donnelly, Jet Propulsion Laboratory, California Institute of Technology

Andrew Loveless, NASA-JSC

Brendan Hall, NASA-JSC (Jacobs)

Paul Miner, NASA-LaRC

Abstract:

Keywords:

Introduction

This introduction is not part of P<designation>/D0, Draft <Gde./Rec. Prac./Std.> for Fault-Tolerant Timing with Time Integrity.

Contents

Amendment: Fault-Tolerant Timing with Time Integrity

1. Overview

2. Normative references

3. Definitions

4. Abbreviations and acronyms

Insert the following acronyms into the existing list of acronyms

DTSF	Dependent Time Selection Function
FTTM	Fault-Tolerant Timing Module
ITSF	Independent Time Selection Function
MVTISA	mid-value time-index selection algorithm
TSF	Time Selection Function

5. Conformance

6. Conventions

6.1 General

6.2 Service specification method and notation

6.3 Lexical form syntax

6.4 Data types and on-the-wire formats

6.4.1 General

6.4.2 Primitive data types specifications

6.4.3 Derived data type specifications

6.4.3.1 ScaledNs

6.4.3.2 UscaledNs

6.4.3.3 TimeInterval

6.4.3.4 Timestamp

6.4.3.5 ExtendedTimestamp

6.4.3.6 ClockIdentity

6.4.3.7 PortIdentity

6.4.3.8 ClockQuality

Insert the following subclauses

6.4.3.9 fttmInputTrustStatus

The fttmInputTrustStatus type is an Enumerated value that holds a trust status of an input ClockTarget Interface. The data type is as follows:

— Typedef Enumeration1 fttmInputTrustStatus;

- 0: TRUSTED
- 1: NOT_TRUSTED

6.4.3.10 fttmInterfaceValueArray

The fttmInterfaceValueArray type provides a UInteger32 value that corresponds to input ClockTarget interfaces x and y for a DTSF or the ITSF of the FTTM. The data type is as follows:

- Typedef UInteger32 [fttmNumActiveTimeIndexes][fttmNumActiveTimeIndexes]
  fttmInterfaceValueArray;

6.4.3.11 fttmOutputTrustState

The fttmOutputTrustState type is an Enumerated value that holds the output trust state of a TSF or of the FTTM. The data type is as follows:

- Typedef Enumeration3 fttmOutputTrustState;
  - 000: NOT_TRUSTED (see 20.3.3.3)
  - 001: TIME_TRUSTED (see 20.3.3.3)
  - 010: FREQ_TRUSTED (see 20.3.3.3)
  - 011: NOT_VALID (see 20.3.3.3)
  - 100: TRUST_STATE_4 (for use by an alternate (i.e., non default) time-index selection algorithm)
  - 101: TRUST_STATE_5 (for use by an alternate (i.e., non default) time-index selection algorithm
  - 110: TRUST_STATE_6 (for use by an alternate (i.e., non default) time-index selection algorithm
  - 111: TRUST_STATE_7 (for use by an alternate (i.e., non default) time-index selection algorithm

6.4.3.12 fttmSelectedIndex (UInteger16)

The fttmSelectedIndex type gives the selected ClockTarget interface index for a DTSF or for the ITSF, with the following meanings.

- 0 to 255: The index number of the selected input ClockTarget interface.
- 256 to 510: Reserved.
- 511: The NQ index.
- >511: Reserved.

6.4.3.13 fttmUint16NumActiveDtsfs

The fttmUint16NumActiveDtsfs type is a vector of UInteger16 values with NumActiveDtsfs members. The data type is as follows:

— Typedef UInteger16 [fttmNumActiveDtsfs];

The UInteger16 values have the following representations:

— 0 to 255: Represents the index number of a selected input ClockTarget interface.
— 256 to 510: Reserved.
— 511: Represents the NQ index.
— > 511: Reserved.

6.4.3.14 fttmUint16NumActiveTsfs

The fttmUint16NumActiveTsfs type is a vector of UInteger16 values with NumActiveDtsfs + 1 members. The data type is as follows:

— Typedef UInteger16 [fttmNumActiveDtsfs + 1];

The UInteger16 values have the following representations:

— 1 to 127: Represents the number of TSFs (ITSF + DTSFs) active in the FTTM.
— > 128: Reserved.

6.4.3.15 fttmExtsfttmNumActiveTimeIndexes

The fttmExtsfttmNumActiveTimeIndexes type is a vector of ExtendedTimestamp values with fttmNumActiveTimeIndexes members. The data type is as follows:

— Typedef ExtendedTimestamp [fttmNumActiveTimeIndexes];

6.4.3.16 fttmOctet128NumActiveTsfs

The fttmOctet128NumActiveTsfs type is a vector of Octet128 values with fttmNumActiveDtsfs + 1 members. The data type is as follows:

— Typedef Octet128 [fttmNumActiveDtsfs + 1];

6.4.3.17 fttmUint8Uint8fttmNumActiveTimeIndexes

The fttmUint8Uint8fttmNumActiveTimeIndexes type is a vector of a pair of UInteger8 values with fttmNumActiveTimeIndexes members. The data type is as follows:

— Typedef UInteger8 UInteger8 [fttmNumActiveTimeIndexes];

6.4.3.18 fttmUint32fttmNumActiveTimeIndexes

The fttmUint32fttmNumActiveTimeIndexes type is a vector of Uint32 values with fttmNumActiveTimeIndexes members. The data type is as follows:

— Typedef UInteger32 [fttmNumActiveTimeIndexes];

7. Time-synchronization model for a packet network

7.1 General

7.2 Architecture of a time-aware network

7.2.1 General

7.2.2 Time-aware network consisting of a single gPTP domain

7.2.3 Time-aware network consisting of multiple gPTP domains

7.2.4 Time-aware networks with redundant Grandmaster PTP Instances and/or redundant paths

7.2.5 Time-aware network with hot standby

7.2.6 Time-aware network with fault-tolerant timing and time integrity

Figure 1 shows an example of a time-aware network with fault-tolerant timing and time integrity (see Clause 20). The network has three independent GMs whose clock sources (CS1, CS2, and CS3) have achieved time agreement generation and preservation (see 20.2.3), independent time distribution paths (see 20.2.4 and 20.2.6) from GMs 1 and 3 to the target, two dependent time distribution paths (see 20.2.4 and 20.2.5) from GM 2 through a common bridge (B1) to the target, and a Fault-Tolerant Timing Module (FTTM, see 20.3) at the target to select an input time that has integrity to deliver to the destination application's ClockTarget.

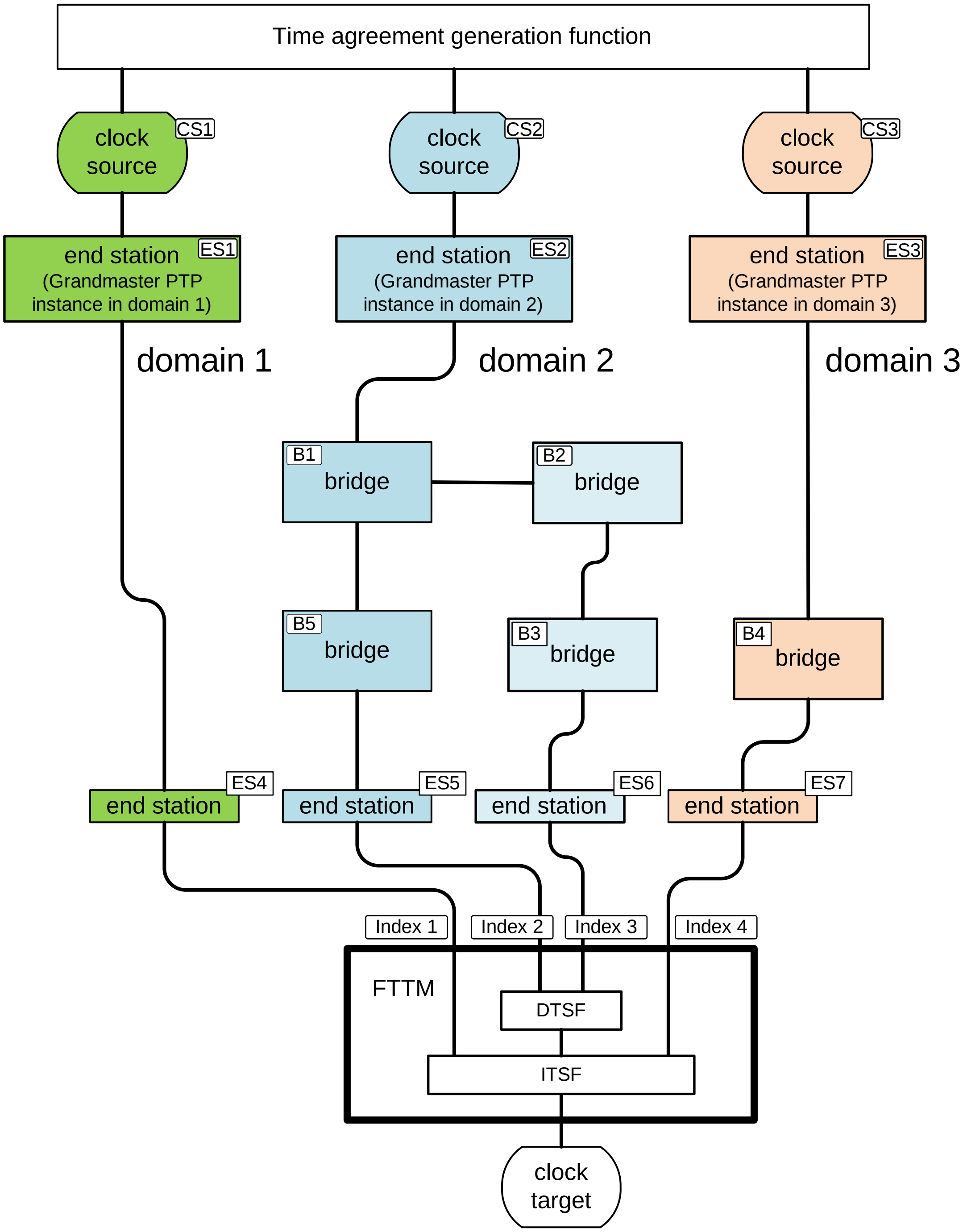


Figure 1—Time-aware network example for fault-tolerant timing with time integrity

8. IEEE 802.1AS concepts and terminology

9. Application Interfaces

10. Media-independent layer specification

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

12. Media-dependent layer specification for IEEE 802.11 links

13. Media-dependent layer specification for interface to IEEE 802.3 Ethernet passive optical network link

14. Timing and synchronization management

14.1 General

14.1.1 Data set hierarchy

14.1.2 Data set descriptions

14.2 Default Parameter Data Set (defaultDS)

14.3 Current Parameter Data Set (currentDS)

14.4 Parent Parameter Data Set (parentDS)

14.5 Time Properties Parameter Data Set (timePropertiesDS)

14.6 Path Trace Parameter Data Set (pathTraceDS)

14.7 Acceptable TimeTransmitter Table Parameter Data Set (acceptableTimeTransmitterTableDS)

14.8 PTP Instance Synchronization Parameter Data Set (ptpInstanceSyncDS)

14.9 Drift Tracking Parameter Data Set (driftTrackingDS)

14.10 Port Parameter Data Set (portDS)

14.11 Description Port Parameter Data Set (descriptionPortDS

14.12 Port Parameter Statistics Data Set (portStatisticsDS)

14.13 Acceptable TimeTransmitter Port Parameter Data Set (acceptableTimeTransmitterPortDS)

14.14 External Port Configuration Port Parameter Data Set (externalPortConfigurationPortDS)

14.15 Asymmetry Measurement Mode Parameter Data Set (asymmetryMeasurementModeDS)

14.16 Common Services Port Parameter Data Set (commonServicesPortDS)

14.17 Common Mean Link Delay Service Default Parameter Data Set (cmldsDefaultDS)

14.18 Common Mean Link Delay Service Link Port Parameter Data Set (cmldsLinkPortDS)

14.19 Common Mean Link Delay Service Link Port Parameter Statistics Data Set (cmldsLinkPortStatisticsDS)

14.20 Common Mean Link Delay Service Asymmetry Measurement Mode Parameter Data Set (cmldsAsymmetryMeasurementModeDS)

14.21 Hot Standby System Parameter Data Set (hotStandbySystemDS)

14.22 Hot Standby System Description Parameter Data Set (hotStandbySystemDescriptionDS)

Insert the following subclause

14.23 Fault Tolerant Timing Module System Parameter Data Set (fttmSystemDS)

14.23.1 General

The fttmSystemDS describes the attributes of the respective instance of the Fault-Tolerant Timing Module Service.

14.23.2 dtsfMaxNumTimeIndexes (UInteger8)

The dtsfMaxNumTimeIndexes object gives the maximum number of input ClockTarget Interfaces available on each of the DTSFs in the FTTM.

The dtsfMaxNumTimeIndexes object is read-only, with a value from 2 to 127. The default value is implementation specific.

The dtsfMaxNumTimeIndexes object shall be provided if the FTTM service is used.

14.23.3 fttmCollectedTod (fttmExtsfttmNumActiveTimeIndexes)

The fttmCollectedTod object is a vector of collected extended timestamps that correspond to the latest ClockTarget invoke event. The vector member fttmCollectedTod[x] holds the latest timeReceiverTimeCallback result for input ClockTarget interface x to the FTTM.

The value of fttmCollectedToD object is valid when fttmInvokeStatusAvail is TRUE.

The fttmCollectedTod object is read-only, with a default value of 0 for every member of the vector.

The fttmCollectedToD object shall optionally be provided if the FTTM service is used.

14.23.4 fttmMapDtsfToItsf (fttmUint16NumActiveDtsfs)

The fttmMapDtsfToItsf object provides the mapping for all the FTTM's DTSF output ClockTarget Interfaces to the ITSF and its input indexes. See K.7.

Each fttmMapDtsfToItsf[x] object, where x is a value from 1 to fttmNumActiveDtsfs (see 14.23.13), consists of the item:

- — The index number of the DTSF's/ITSF's input ClockTarget Interface that the FTTM's input ClockTarget Interface is connected to.
  - — ITSF instance's input ClockTarget Interface's index values range from 1 to fttmNumActiveTimeIndexes (see 14.23.14).
  - — A value of 0 means the DTSF's output ClockTarget Interface is not connected.

The fttmMapDtsfToItsf object is read/write, with a default value of 0 for all vector members.

The fttmMapDtsfToItsf object shall be provided if the FTTM service is used and if fttmMaxNumDtsfs is not equal to 0.

An example of a FTTM that has 2 DTSFs (fttmNumActiveDTSFs = 2) that has the following connections is given below:

- The output of DTSF instance 1 connects to the ITSF input index 3.
- The output of DTSF instance 2 connects to the ITSF input index 2.

The corresponding example fttmMapDtsfToItsf object would be as follows:

- fttmMapDtsfToItsf[1] = 3 // DTSF #1's output to ITSF input index #3
- fttmMapDtsfToItsf[2] = 2 // DTSF #2's output to ITSF input index #2

14.23.5 fttmMapIndexToTsf (fttmUint8Uint8fttmNumActiveTimeIndexes)

The fttmMapIndexToTsf object provides the mapping for all of the FTTM's input ClockTarget Interface index numbers to a Time Selection Function (TSF) instance number and its input ClockTarget Interface index number. See K.6.

Each fttmMapIndexToTsf[x] object, where x is a value from 1 to fttmNumActiveTimeIndexes (see 14.23.14 and is equal to the index number of the FTTM input ClockTarget Interface, consists of two items:

- — The TSF instance number that the FTTM's input ClockTarget Interface is connected to.
  - — The ITSF instance number is 0.
  - — The DTSF instance numbers range from 1 to fttmMaxNumDtsfs (see 14.23.11).
- — The index number of the DTSF's/ITSF's input ClockTarget Interface that the FTTM's input ClockTarget Interface is connected to.
  - — DTSF instance's input ClockTarget Interface's index values range from 1 to dtsfMaxNumTimeIndexes (see 14.23.2).
  - — ITSF instance's input ClockTarget Interface's index values range from 1 to fttmNumActiveTimeIndexes (see 14.23.14).
  - — A value of 0 means that the associated DTSF's/ITSF's input ClockTarget Interface is not active.

The fttmMapIndexToTsf object is read/write, with a default value set of {0,0} for all vector members.

The fttmMapIndexToTsf object shall be provided if the FTTM service is used.

An example of a FTTM that uses 5 ClockTarget Interfaces (fttmNumActiveTimeIndexes = 5) and 2 DTSFs (fttmNumActiveDTSFs = 2) that has the following connections is given below:

- The first and second FTTM input ClockTarget Interfaces, with indexes 1 and 2, connect to DTSF instance 1's input indexes 1 and 2, respectively.
- The third and fourth ClockTarget Interfaces, with indexes 3 and 4, connect to DTSF instance 2's input indexes 1 and 2, respectively.
- The fifth ClockTarget Interface, with index 5, connects to the ITSF index 1.

The corresponding example fttmMapIndexToTsf object would be as follows:

- fttmMapIndexToTsf[1] = {1,1} //FTTM input index #1 to DTSF instance #1's input index #1
- fttmMapIndexToTsf[2] = {1,2} //FTTM input index #2 to DTSF instance #1's input index #2
- fttmMapIndexToTsf[3] = {2,1} //FTTM input index #3 to DTSF instance #2's input index #1
- fttmMapIndexToTsf[4] = {2,2} //FTTM input index #4 to DTSF instance #2's input index #2
- fttmMapIndexToTsf[5] = {0,1} //FTTM input index #5 to ITSF input index #1

The connections of the two DTSF's output ClockTarget interfaces to the ITSF's input ClockTarget interfaces would be defined by fttmMapDtsfToItsf (see 14.23.4).

14.23.6 fttmMapPtpInstanceToIndex (fttmUint32fttmNumActiveTimeIndexes)

The fttmMapPtpInstanceToIndex object provides the mapping of the instance index number of the PTP Instance to the FTTM input index number. See K.5.

Each fttmMapPtpInstanceToIndex[x] member, where x is a value from 1 to fttmNumActiveTimeIndexes (see 14.23.14) and represents the index number of the FTTM input ClockTarget interface, is a UInteger32 value that represents the instance index number of the PTP Instance that is connected to the FTTM input port with the input index number x.

The fttmMapPtpInstanceToIndex object is read/write and shall be provided if the FTTM service is used.

14.23.7 fttmHyst (fttmInterfaceValueArray)

This fttmHyst object is a two-dimensional array of UInteger32 values, each in units of 2^{-16} nanoseconds. The array has a size of fttmNumActiveTimeIndexes in each dimension. The object fttmHyst[x][y] holds the hysteresis to be added to fttmMaxAS[x][y] (see 14.23.10) for the times of the two input FTTM input ClockTarget interfaces with index numbers x and y.

The hysteresis enables the use of one time skew level to set the trust status and another time skew level to clear the trust status.

The fttmHyst object is read/write and has a default value of 0 for all array members.

The fttmHyst object shall be provided if the FTTM service is used.

14.23.8 fttmInvokeStatusAvail (Boolean)

The fttmInvokeStatusAvail object indicates that the FTTM has updated the values of the following read-only status objects, after an invoke event from the ClockTarget:

— fttmCollectedTod

When fttmInvokeStatusAvail is TRUE, the above read-only status objects have been updated.

When fttmInvokeStatusAvail is FALSE, the above read-only status objects have not been updated.

fttmInvokeStatusAvail is read-only and has a default value of FALSE.

fttmInvokeStatusAvail is cleared to FALSE by assertion of fttmInvokeStatusAvailClr to TRUE.

To detect each update, fttmInvokeStatusAvailClr must be asserted to TRUE to clear fttmInvokeStatusAvail before each update occurs.

The fttmInvokeStatusAvail object shall optionally be provided if the FTTM service is used.

14.23.9 fttmInvokeStatusAvailClr (Boolean)

The fttmInvokeStatusAvailClr object is used to clear the fttmInvokeStatusAvail object to FALSE. When fttmInvokeStatusAvailClr is changed from FALSE to TRUE, fttmInvokeStatusAvail is cleared to FALSE.

The fttmInvokeStatusAvailClr object must be written to FALSE before it can be used again.

The fttmInvokeStatusAvailClr object is read/write, with a default value of FALSE.

The fttmInvokeStatusAvailClr object shall optionally be provided if the FTTM service is used.

14.23.10 fttmMaxAs (fttmInterfaceValueArray)

The fttmMaxAs object is a two-dimensional array of UInteger32 values, each in units of 2^{-16} nanoseconds. The array has a size of fttmNumActiveTimeIndexes in each dimension. The fttmMaxAS[x][y] object gives the maximum magnitude of expected skew between times provided by the FTTM input ClockTarget interfaces of index x and index y when those times are not faulty. This value is used as the criteria to determine the trustworthiness of the times being compared. See $maxAS_{xy}$ in M.5.

The fttmMaxAs object is read/write and has a default value of 0 for all array members.

The fttmMaxAs object shall be provided if the FTTM service is used.

14.23.11 fttmMaxNumDtsfs (UInteger8)

The fttmMaxNumDtsfs objects gives the maximum number of DTSF instances available in the FTTM. The default value is implementation specific.

The fttmMaxNumDtsfs object is read-only, with a value from 0 to 126.

The fttmMaxNumDtsfs object shall be provided if the FTTM service is used.

14.23.12 fttmMaxNumTimeIndexes (UInteger8)

The fttmMaxNumTimeIndexes object gives the maximum number of input ClockTarget Interfaces available on the FTTM. The default value is implementation specific.

The fttmMaxNumTimeIndexes object is read-only, with a value from 1 to 255.

The fttmMaxNumTimeIndexes object shall be provided if the FTTM service is used.

14.23.13 fttmNumActiveDtsfs (UInteger8)

The fttmNumActiveDtsfs object gives the number of active DTSF instances currently used in the FTTM.

The fttmNumActiveDtsfs object is read-only, with a value from 0 to fttmMaxNumDtsfs.

The fttmNumActiveDtsfs object shall optionally be provided if the FTTM service is used.

14.23.14 fttmNumActiveTimeIndexes (UInteger8)

The fttmNumActiveTimeIndexes object gives the number of input ClockTarget Interfaces currently active on the FTTM, where an active input ClockTarget Interface is one that has been mapped to either a DTSF instance or to the ITSF.

The fttmNumActiveTimeIndexes object is read-only, with a value from 1 to fttmMaxNumTimeIndexes.

The fttmNumActiveTimeIndexes object shall optionally be provided if the FTTM service is used.

14.23.15 fttmSelInstanceIndex (UInteger32)

The fttmSelInstanceIndex object gives the instanceIndex value of the PTP Instance that is the source of the ClockTargetEventCapture interface that was selected by the FTTM as its trusted time. The value is only valid if the FTTM output ClockTargetEventCapture has isSynced and gmPresent both equal to TRUE.

The fttmSelInstanceIndex object is read-only.

14.23.16 fttmSelTimeIndexChangeCnt (Uinteger16)

The fttmItsfSelTimeIndexChangeCnt object gives the number of times the ITSF has changed its time index selection.

The fttmItsfSelTimeIndexChangeCnt object is read-only, with a value from 0 to 65535. The count rolls over to 0 if the count is incremented when its current value is 65535. The default value is 0.

The fttmItsfSelTImeIndexChangeCnt object shall optionally be provided if the FTTM service is used.

The fttmItsfSelTImeIndexChangeCnt object is used with fttmInvokeStatusAvail (see 14.23.8) and fttmInvokeStatusAvailClr (see 14.23.9).

14.23.17 fttmSelTimeRateRatioOffset (Uint32)

The fttmSelTimeRateRatioOffset object gives the maximum rate-ratio offset magnitude, between the FTTM's selected time clock rate and the FTTM's local clock (OSC_CLK) rate, that is deemed to be acceptable to go to or remain in the FREQ_TRUSTED state (see 20.3.3.3).

The rateRatio offset magnitude is expressed as a fractional frequency offset multiplied by 2^{41}.

The fttmSelTimeRateRatioOffset object is read/write and has a default rate-ratio value of 0 for all object members.

The fttmSelTimeRateRatioOffset object shall optionally be provided if the FTTM service is used.

14.23.18 fttmSelTimeStdDevRateRatioOffset (Uint32)

The fttmSelTimeStdDevRateRatioOffset object gives the maximum standard deviation of the rate-ratio offset, between the FTTM's selected time clock rate and the FTTM's local clock (OSC_CLK) rate, that is deemed to be acceptable to go to or remain in the FREQ_TRUSTED state (see 20.3.3.3).

The standard deviation of the rateRatio offset is expressed as a fractional frequency offset multiplied by 2^{41}.

The fttmSelTimeStdDevRateRatioOffset object is read/write and has a default rate-ratio value of 0 for all object members.

The fttmSelTimeStdDevRateRatioOffset object shall optionally be provided if the FTTM service is used.

14.23.19 fttmTrustState (fttmOutputTrustState)

The fttmTrustState object holds the output trust state of the FTTM. Valid values are NOT_TRUSTED, TIME_TRUSTED, FREQ_TRUSTED, and NOT_VALID.

The fttmTrustState object is read-only.

14.23.20 fttmTsfAlgoName (fttmOctet128NumActiveTsfs)

The fttmTsfAlgoName object is a vector of strings, where each vector member provides the name of the algorithm used by each active TSF instance number (the ITSF and the active DTSFs).

Each fttmTsfAlgoName[x] member, where x is a value from 0 to fttmNumActiveDtsfs (see14.23.13) and represents the TSF instance number, is a 128 octet that contains a string with the name of the algorithm for the TSF instance number.

- The TSF instance number is assigned as follows.
    - The ITSF instance number is 0.
    - The DTSF instance numbers range from 1 to fttmNumActiveDtsfs (see 14.23.13).

The fttmTsfAlgoName object is read-only.

The default TSF algorithm and its name are discussed in 20.3.5.

The fttmTsfAlgoName object shall optionally be provided if the FTTM service is used.

14.23.21 fttmTsfSelTimeIndex (fttmUint16NumActiveTsfs)

The fttmTsfSelTimeIndex object gives the input time index that is selected by each TSF instance.

Each fttmTsfSelTimeIndex[x] member, where x is a value from 0 to fttmNumActiveDtsfs (see 14.23.11) and represents the TSF instance number, is read-only and has a default value of 511 (i.e., the NQ index).

- The TSF instance number is assigned as follows.
    - The ITSF instance number is 0.
    - The DTSF instance numbers range from 1 to fttmNumActiveDtsfs (see 14.23.13).

The fttmTsfSelTimeIndex object is read-only.

The fttmTsfSelTimeIndex object shall optionally be provided if the FTTM service is used.

14.23.22 fttmUseOscClk (Boolean)

The fttmUseOscClk object defines whether the OSC_CLK frequency is used as a reference for time integrity.

If fttmUseOscClk is TRUE, then the OSC_CLK frequency is used as a reference for checking time integrity (e.g., for entering the FREQ_TRUSTED state in the FTTM state machine, per 20.3.3.3).

If fttmUseOscClk is FALSE, then the OSC_CLK frequency is not used as a reference for checking time integrity.

The fttmUseOscClk object is read-only and has an implementation-specific default value.

The fttmUseOscClk object shall optionally be provided if the FTTM service is used.

14.24 Fault Tolerant Timing Module System Description Parameter Data Set (fttmSystemDescriptionDS)

14.24.1 General

The fttmSystemDescriptionDS contains descriptive information for the respective instance of the Fault-Tolerant Timing Module Service.

14.24.2 userDescription

The user description is a character string whose maximum length is 128.

14.24.3 fttmSystemDescriptionDS table

There is one fttmSystemDescriptionDS table per fttmSystem instance, as detailed in Table 1.

Table 1— fttmSystemDescriptionDS table

Name	Data type	Operations supported	References
userDescription	Octet128	RW	14.24.2

NOTE—RW = read/write access

15. Managed object definitions

16. Media-dependent layer specification for CSN

17. YANG Data Model

17.1 YANG framework

Change 17.1.1 as follows

17.1.1 Relationship to the IEEE Std 1588 data model

The YANG data models specified in this standard are based on, and augment, those specified in IEEE Std 1588. In particular the ieee802-dot1as-gptp.yang module imports the ieee1588-ptp-tt module as a whole, augmenting that module as necessary to meet the requirements of this standard. ~~In addition, t~~The ieee802-dot1as-hs.yang module imports the ieee1588-ptp-tt and ieee802-dot1as-gptp modules as a whole, augmenting those modules as necessary to meet the requirements of this standard. ~~Also, t~~The ieee802-dot1as-hd.yang module imports the ieee1588-ptp-tt, the ieee802-dot1as-gpt, and the ieee802-dot1as-hs modules as a whole, augmenting those modules as necessary to meet the requirements of this standard. The ieee802dot1as-fttm.yang module imports the ieee1588-tt and ieee802-dot1as-gptp modules as a whole, augmenting those modules as necessary to meet the requirements of this standard.

Some of the data sets in Clause 14 (e.g., defaultDS) are derived from IEEE Std 1588, and some of the data sets are unique to IEEE Std 802.1AS (i.e., not derived from IEEE Std 1588). For each data set in Clause 14 that is derived from IEEE Std 1588, a portion of the members are derived from IEEE Std 1588, and the remaining members are unique to IEEE Std 802.1AS. For the members that are derived from IEEE Std 1588, the specifications in both standards are analogous (i.e., same name, data type, semantics, etc).

The YANG data model for IEEE Std 1588-2019 is published as amendment IEEE Std 1588e. The YANG module of IEEE Std 1588e (ieee1588-ptp-tt.yang) contains the hierarchy (tree) of data sets and their members.

The YANG modules of this clause (ieee802-dot1as-gptp.yang,~~ and~~ ieee802-dot1as-hs.yang, ieee802-dot1as-hd.yang, and ieee802-dot1as-fttm.yang) use the YANG "import" statement to import the YANG module of IEEE Std 1588e. This effectively uses the IEEE Std 1588 YANG tree as the foundation of the IEEE Std 802.1AS YANG tree. By importing the tree and its data set containers, all members from Clause 14 that are derived from IEEE Std 1588 are also imported.

17.2 IEEE 802.1AS YANG model

Change the following paragraph, as shown:

Figure 17-4 provides detail for the common services, including each data set member. The Common Mean Link Delay Service (cmlds) has a data sets for the service itself (e.g., default-ds), and data sets for each PTP Link Port. The Hot Standby Service has data sets for each HotStandbySystem. The Fault Tolerant Timing Module Service has data sets for each fttmSystem.

Replace Figure 17-1 with the following:

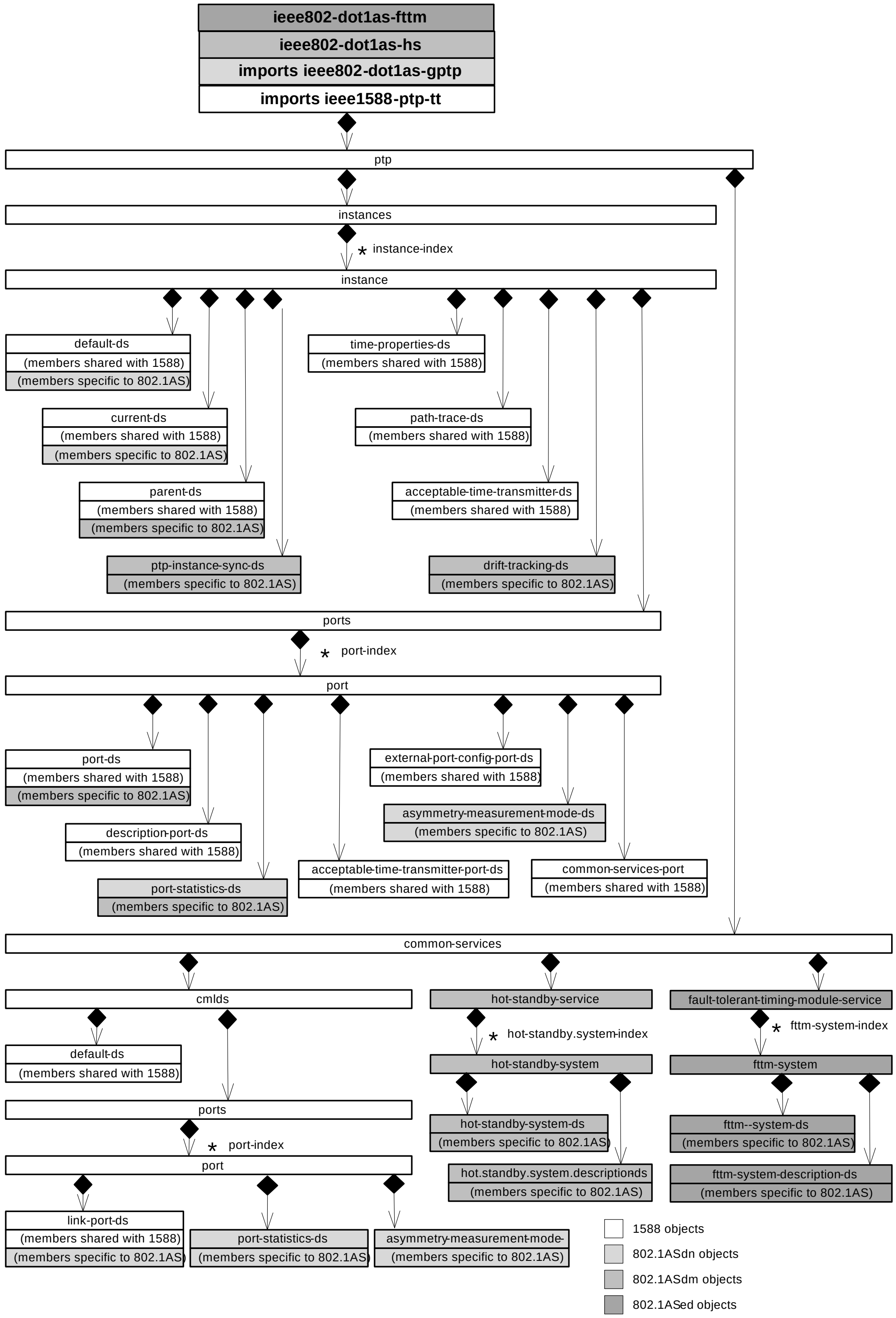


Figure 2—Overview of YANG tree

Replace Figure 17-4 with the following:

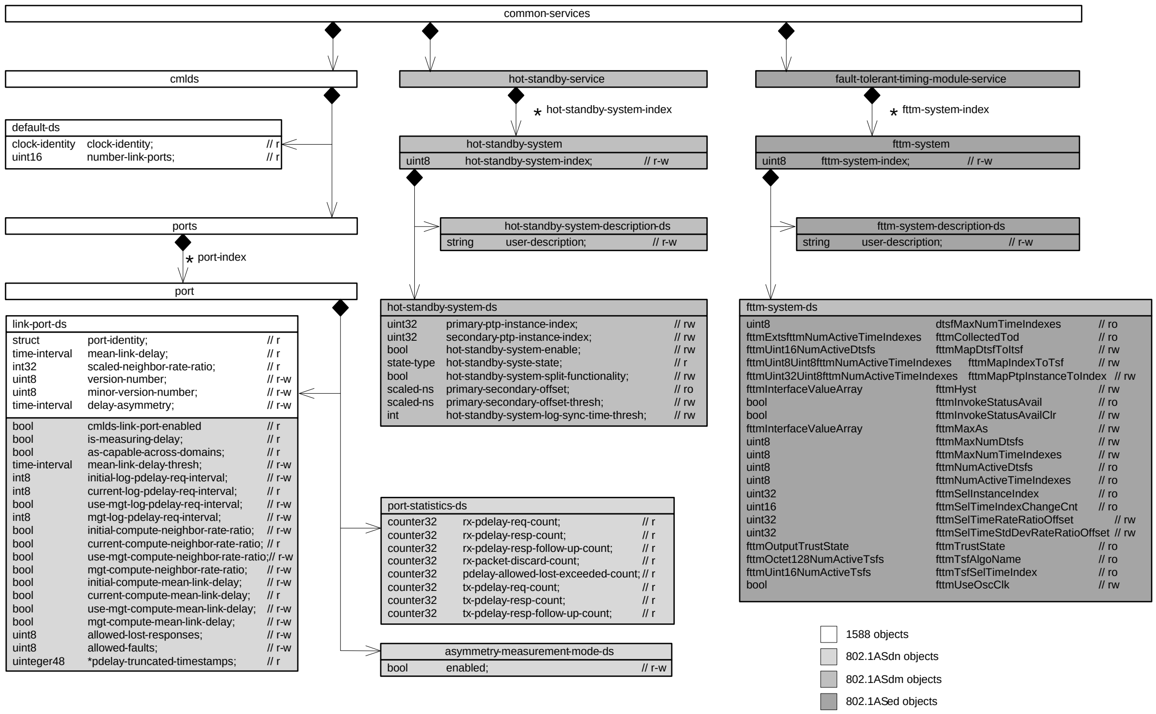


Figure 3—Common services detail

17.3 Structure of YANG data model

Change Table 17-1 as follows:

Table 17-1—Summary of the YANG modules

Module	Managed Functionality	YANG specification notes
ietf-yang-types	Type definitions	IETF RFC 6991 - Common YANG Data Types.
ieee1588-ptp-tt	Clause 14	IEEE Std 1588e - MIB and YANG Data Models. IEEE Std 802.1ASdn imports this YANG module as its foundational tree, including a subset of members from Clause 14.
ieee802-dot1as-gptp	Clause 14	IEEE Std 802.1ASdn - YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std 802.1AS.
ieee802-dot1as-hs	Clause 14	IEEE Std 802.1ASdm - YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std 802.1ASdm.
ieee802-dot1as-hd	Clause 14	IEEE Std 802.1ASds - YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std802.1ASds.
ieee802-dot1as-fttm	Clause 14	IEEE Std 802.1ASed - YANG Data Model. The YANG module of this clause uses YANG augments to add members from Clause 14 that are unique to IEEE Std 802.1ASed.

17.4 Security considerations

17.5 YANG schema tree definitions

17.5.1 Tree diagram for ieee802-dot1as-gptp.yang

17.5.2 Tree diagram for ieee802-dot1as-hs.yang

17.5.3 Tree diagram for ieee802-dotas-hd.yang

Insert the following subclause:

17.5.4 Tree diagram for ieee802-dot1as-fttm.yang

```
module: ieee802-dot1as-fttm

  augment /ptp-tt:ptp/ptp-tt:common-services:
```

```
+--rw fault-tolerant-timing-module-service {fttm}?
   +--rw fttm-system* [fttm-system-index]
      +--rw fttm-system-index             uint8
      +--rw fttm-system-ds
      |  +--ro dtsf-max-num-time-indexes?                 uint8
      |  +--rw fttm-collected-tod-list* [fttm-input-index-number]
      |  |  +--rw fttm-input-index-number    uint8
      |  |  +--ro extended-timestamp-list* [seconds fractional-nanoseconds]
      |  |     +--ro seconds                   uint48
      |  |     +--ro fractional-nanoseconds    uint48
      |  +--rw fttm-map-dtsf-to-itsf-list* [tsf-instance-number]
      |  |  +--rw tsf-instance-number        uint8
      |  |  +--rw itsf-input-index-number?   uint8
      |  +--rw fttm-map-index-to-tsf-list* [fttm-input-index-number]
      |  |  +--rw fttm-input-index-number    uint8
      |  |  +--rw tsf-instance-number?       uint8
      |  |  +--rw tsf-input-index-number?    uint8
      |  +--rw fttm-map-ptp-instance-to-index-list* [fttm-input-index-number]
      |  |  +--rw instance-index?            uint32
      |  |  +--rw fttm-input-index-number    uint8
      |  +--rw fttm-hyst-lists* [fttm-input-index-number]
      |  |  +--rw fttm-input-index-number    uint8
      |  |  +--rw fttm-hyst-list* [fttm-input-index-number]
      |  |     +--rw fttm-input-index-number    uint8
      |  |     +--rw fttm-hyst?                 uint32
      |  +--ro fttm-invoke-status-avail?                  boolean
      |  +--rw fttm-invoke-status-avail-clr?              boolean
      |  +--rw fttm-max-as-lists* [fttm-input-index-number]
      |  |  +--rw fttm-input-index-number    uint8
      |  |  +--rw fttm-max-as-list* [fttm-input-index-number]
      |  |     +--rw fttm-input-index-number    uint8
      |  |     +--rw fttm-max-as?               uint32
      |  +--ro fttm-max-num-dtsfs?                        uint8
      |  +--ro fttm-max-num-time-indexes?                 uint8
      |  +--ro fttm-num-active-dtsfs?                     uint8
      |  +--ro fttm-num-active-time-indexes?              uint8
      |  +--ro fttm-sel-instance-index?                   uint32
      |  +--ro fttm-sel-time-index-change-cnt?            uint16
      |  +--rw fttm-sel-time-rate-ratio-offset?           uint32
      |  +--rw fttm-sel-time-std-dev-rate-ratio-offset?   uint32
      |  +--ro fttm-trust-state?                          fttm-output-trust-state
      |  +--rw fttm-tsf-algo-name-list* [tsf-instance-number]
      |  |  +--rw tsf-instance-number    uint8
      |  |  +--ro fttm-tsf-algo-name?    string
      |  +--rw fttm-tsf-sel-time-index-list* [tsf-instance-number]
      |  |  +--rw tsf-instance-number        uint8
      |  |  +--ro fttm-tsf-sel-time-index?   uint16
      |  +--ro fttm-use-osc-clk?                          boolean
      +--rw fttm-system-description-ds
         +--rw user-description?   string
```

17.6 YANG modules

17.6.1 Module ieee802-dot1as-gptp.yang

17.6.2 Module ieee802-dot1as-hs.yang

17.6.3 Module ieee802-dot1as-hd.yang

Insert the following subclause:

17.6.4 Module ieee802-dot1as-fttm.yang

```
module ieee802-dot1as-fttm {
  yang-version 1.1;
  namespace "urn:ieee:std:802.1AS:yang:ieee802-dot1as-fttm";
  prefix dot1as-fttm;

  import ieee1588-ptp-tt {
    prefix ptp-tt;
  }

  organization
    "IEEE 802.1 Working Group";
  contact
    "WG-URL: http://www.ieee802.org/1/
    WG-EMail: stds-802-1-L@ieee.org

    Contact: IEEE 802.1 Working Group Chair
    Postal: C/O IEEE 802.1 Working Group
    IEEE Standards Association
    445 Hoes Lane
    Piscataway, NJ 08854
    USA

    E-mail: stds-802-1-chairs@ieee.org";
  description
    "This module provides for management of IEEE Std 802.1ASed components that support
fault-tolerant timing module.

    Copyright (C) IEEE (2024).
    This version of this YANG module is part of IEEE Std 802.1AS;
    see the standard itself for full legal notices.";

  revision 2024-09-02 {
    description
      "Published as part of IEEE Std 802.1ASed-2024.

      The following reference statement identifies each referenced IEEE
      Standard as updated by applicable amendments.";
    reference
      "IEEE Std 802.1AS-2020 as modified by
       IEEE Std 802.1AS-2020/Cor-1-2021, and amended by
       IEEE Std 802.1ASdr, IEEE Std 802.1ASdn,
       IEEE Std 802.1ASdm,IEEE Std 802.1ASds, and IEEE Std 802.1ASed.";
  }


  feature fttm {
    description
      "This feature indicates that the device supports the Fault-tolerant
       timing module (FTTM) functionality.";
  }

  typedef fttm-output-trust-state {
    type enumeration {
      enum NOT-TRUSTED {
        value 0;
        description
          "Not trusted";
      }
      enum TIME-TRUSTED {
        value 1;
        description
          "Time trusted";
      }
      enum FREQ-TRUSTED {
        value 2;
        description
          "Frequency trusted";
      }
```

```
      enum NOT_VALID {
        value 3;
        description
          "The trust state is not valid";
      }
      enum TRUST-STATE-4 {
        value 4;
        description
          "Trust state 4, reserved";
      }
      enum TRUST-STATE-5 {
        value 5;
        description
          "Trust state 5, reserved";
      }
      enum TRUST-STATE-6 {
        value 6;
        description
          "Trust state 6, reserved";
      }
      enum TRUST-STATE-7 {
        value 7;
        description
          "Trust state 7, reserved";
      }
    }
    description
      "The fttmOutputTrustState type is an enumerated value that holds the output trust
state of a TSF or of the FTTM.";
    reference
      "6.4.3.11 of IEEE Std 802.1ASed";
  }

  typedef uint48 {
    type uint64 {
      range "0..281474976710655";
    }
    description
      "Unsigned 48-bit integer.";
  }

  grouping fault-tolerant-timing-module-group {
    description
      "Management of a single FTTM.";
    reference
      "14.23 of IEEE Std 802.1ASed";
    leaf dtsf-max-num-time-indexes {
      type uint8;
      config false;
      description
        "Implementation-specific. Gives the maximum number of input ClockTarget
Interfaces available on each of the DTSFs in the FTTM.";
      reference
        "IEEE Std 802.1ASed 14.23.2";
    }
    uses fttm-collected-tod-group;
    uses fttm-map-dtsf-to-itsf-group;
    uses fttm-map-index-to-tsf-group;
    uses fttm-map-ptp-instance-to-index-group;
    uses fttm-hyst-group;

    leaf fttm-invoke-status-avail {
      type boolean;
      default "false";
      config false;
      description
        "The fttmInvokeStatusAvail object indicates that the FTTM has updated the values
of the fttmCollectedTod status, after an invoke event from the ClockTarget";
      reference
        "14.23.8 of IEEE Std 802.1ASed";
    }
```

```
      leaf fttm-invoke-status-avail-clr {
        type boolean;
        default "false";
        description
          "The fttmInvokeStatusAvailClr object is used to clear the fttmInvokeStatusAvail
object";
        reference
          "IEEE 802.1ASed 14.23.9";
      }
      uses max-as-group;
      leaf fttm-max-num-dtsfs {
        type uint8 {
          range "0..126";
        }
        config false;
        description
          "Implementation-specific. Maximum number of DTSF instances available in the FTTM.
           The value is restricted to the range of 0 to 126.";
        reference
          "IEEE Std 802.1ASed 14.23.11";
      }
      leaf fttm-max-num-time-indexes {
        type uint8 {
          range "0..255";
        }
        config false;
        description
          "Implementation-specific. Maximum number of input ClockTarget Interfaces
available on the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.12";
      }
      leaf fttm-num-active-dtsfs {
        type uint8;
        config false;
        description
          "Number of active DTSF instances currently used in the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.13";
      }
      leaf fttm-num-active-time-indexes {
        type uint8;
        config false;
        description
          "The number of input ClockTarget Interfaces currently active on the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.14";
      }
      leaf fttm-sel-instance-index {
        type uint32;
        config false;
        description
          "Gives the instanceIndex value of the PTP Instance that is the source of the
ClockTarget interface that was selected by the FTTM as its trusted time.";
        reference
          "IEEE Std 802.1ASed 14.23.15";
      }
      leaf fttm-sel-time-index-change-cnt {
        type uint16;
        config false;
        description
          "The fttmItsfSelTimeIndexChangeCnt object gives the number of times the ITSF has
changed its time index selection.";
        reference
          "IEEE 802.1ASed 14.23.16";
      }
      leaf fttm-sel-time-rate-ratio-offset {
        type uint32;
        default "0";
        description
```

```
        "The fttmSelTimeRateRatioOffset object gives the maximum rate-ratio offset
magnitude, between the FTTM’s selected time clock rate and the FTTM’s local clock
(OSC_CLK) rate, that is deemed to be acceptable to go to or remain in the FREQ_TRUSTED
state (see 20.3.3.2.3).";
      reference
        "IEEE Std 802.1ASed 14.23.17";
    }
    leaf fttm-sel-time-std-dev-rate-ratio-offset {
      type uint32;
      default "0";
      description
        "The fttmSelTimeStdDevRateRatioOffset object gives the maximum standard deviation
of the rate-ratio offset, between the FTTM’s selected time clock rate and the FTTM’s
local clock (OSC_CLK) rate, that is deemed to be acceptable to go to or remain in the
FREQ_TRUSTED state (see 20.3.3.2.3).";
      reference
        "IEEE Std 802.1ASed 14.23.18";
    }
    leaf fttm-trust-state {
      type fttm-output-trust-state;
      default "NOT-TRUSTED";
      config false;
      description
        "The fttmTrustState object holds the output trust state of the FTTM.
         Valid values are NOT_TRUSTED, TIME_TRUSTED, FREQ_TRUSTED, and NOT_VALID.";
      reference
        "IEEE Std 802.1ASed 14.23.19";
    }
    uses fttm-tsf-algo-name-group;
    uses fttm-tsf-sel-time-index-group;
    leaf fttm-use-osc-clk {
      type boolean;
      config false;
      description
        "implementation-specific. Defines whether the OSC_CLK frequency is used as a
reference for time integrity.";
      reference
        "IEEE Std 802.1ASed 14.23.22";
    }
  }

  grouping instance-index-group {
    description
      "An index number that identifies a ptp-instance input to the FTTM from the 1588e
YANG module.";
    reference
      "IEEE Std IEEE Std 1588e-2024 15.3.3.2 Structure";
    leaf instance-index {
      type uint32;
      description
        "An index number that identifies a ptp-instance input to the FTTM from the 1588e
YANG module.";
      reference
        "IEEE Std IEEE Std 1588e-2024 15.3.3.2 Structure";
    }
  }

  grouping fttm-collected-tod-group {
    description
      "The fttmCollectedTod object is a vector of collected extended timestamps that
correspond to the latest ClockTarget invoke event. The vector member fttmCollectedTod[x]
holds the latest timeReceiverTimeCallback result for input ClockTarget interface x to the
FTTM.";
    reference
      "IEEE Std 802.1ASed 14.23.3";
    list fttm-collected-tod-list {
      key "fttm-input-index-number";
      description
        "The fttmCollectedTod object is a vector of collected extended timestamps that
correspond to the latest ClockTarget invoke event. The vector member fttmCollectedTod[x]
```

```
holds the latest timeReceiverTimeCallback result for input ClockTarget interface x to the
FTTM.";
      reference
        "IEEE Std 802.1ASed 14.23.3.";
      leaf fttm-input-index-number {
        type uint8;
        description
          "The FTTM input index number.";
        reference
          "IEEE Std 802.1ASed 14.23.3";
      }
      list extended-timestamp-list {
        key "seconds fractional-nanoseconds";
        config false;
        description
          "The ExtendedTimestamp type represents a positive time with respect to the
epoch.";
        reference
          "IEEE 802.1AS-2020 6.4.3.5";
        leaf seconds {
          type uint48;
          description
            "The integer portion of the timestamp in units of seconds.";
        }
        leaf fractional-nanoseconds {
          type uint48;
          description
            "The fractional portion of the timestamp in units of 2^-16 ns.";
        }
      }
    }
  }

  grouping fttm-map-dtsf-to-itsf-group {
    description
      "The fttmMapDtsfToItsf object provides the mapping for all the FTTM’s DTSF output
ClockTarget Interfaces to the ITSF and its input indexes. See K.7.";
    reference
      "IEEE Std 802.1ASed 14.23.4";
    list fttm-map-dtsf-to-itsf-list {
      key "tsf-instance-number";
      description
        "This grouping allows associations of all the DTSF output ClockTarget interfaces
         to ITSF input index numbers.";
      reference
        "IEEE Std 802.1ASed 14.23.4";
      leaf tsf-instance-number {
        type uint8;
        description
          "The DTSF instance number.";
        reference
          "IEEE Std 802.1ASed 14.23.4";
      }
      leaf itsf-input-index-number {
        type uint8;
        default "0";
        description
          "The ITSF’s input ClockTarget index number. ";
        reference
          "IEEE Std 802.1ASed 14.23.4";
      }
    }
  }

  grouping fttm-map-index-to-tsf-group {
    description
      "The fttmMapIndexToTsf object provides the mapping for all of the FTTM’s input
ClockTarget Interface index numbers to a Time Selection Function (TSF) instance number
and its input ClockTarget Interface index number. See K.6.";
    reference
      "IEEE Std 802.1ASed 14.23.5";
```

```
    list fttm-map-index-to-tsf-list {
      key "fttm-input-index-number";
      description
        "This grouping allows associations of the FTTM input index numbers to
         DTSF input index numbers and ITSF input index numbers.";
      reference
        "IEEE Std 802.1ASed 14.23.5";
      leaf fttm-input-index-number {
        type uint8;
        description
          "The FTTM input index number.";
        reference
          "IEEE Std 802.1ASed";
      }
      leaf tsf-instance-number {
        type uint8;
        description
          "The TSF instance number that the FTTM input index number is connected to. A
value of 0 represents the ITSF. Other values represent the DTSF instance numbers.";
        reference
          "IEEE Std 802.1ASed 14.23.5";
      }
      leaf tsf-input-index-number {
        type uint8;
        description
          "The input index number of the TSF (DTSF or ITSF)";
        reference
          "IEEE Std 802.1ASed 14.23.5 ";
      }
    }
  }

  grouping fttm-map-ptp-instance-to-index-group {
    description
      "The fttmMapPtpInstanceToIndex object provides the mapping of the instance index
number of the PTP Instance to the FTTM input index number.";
    reference
      "IEEE Std 802.1ASed 14.23.6";
    list fttm-map-ptp-instance-to-index-list {
      key "fttm-input-index-number";
      description
        "This grouping allows associations of the index numbers of PTP Instances
         to FTTM input index numbers.";
      reference
        "IEEE Std 802.1ASed 14.23.6";
      uses instance-index-group;
      leaf fttm-input-index-number {
        type uint8;
        description
          "The FTTM input index number for the FTTM input ClockTarget
           Interface associated with the PTP Instance with the
           corresponding index number.";
        reference
          "IEEE Std 802.1ASed 14.23.6";
      }
    }
  }

  grouping fttm-hyst-group {
    description
      "This fttmHyst object holds the hysteresis to be added to fttmMaxAS[x][y] (see
14.23.10) for the times of the two input FTTM input ClockTarget interfaces with index
numbers x and y.";
    reference
      "IEEE Std 802.1ASed 14.23.7";
    list fttm-hyst-lists {
      key "fttm-input-index-number";
      description
        "The x index.";
      reference
        "IEEE Std 802.1ASed 14.23.7";
```

```
        leaf fttm-input-index-number {
          type uint8;
          description
            "The first FTTM input index number of the two-dimensional array .";
          reference
            "IEEE Std 802.1ASed 14.23.7";
        }
        list fttm-hyst-list {
          key "fttm-input-index-number";
          description
            "The y index.";
          reference
            "IEEE Std 802.1ASed 14.23.7";
          leaf fttm-input-index-number {
            type uint8;
            description
              "The second FTTM input index number of the two-dimensional array .";
            reference
              "IEEE Std 802.1ASed 14.23.7";
          }
          leaf fttm-hyst {
            type uint32;
            units "2^-16 nanoseconds";
            default "0";
            description
              "The object fttmHyst[x][y] holds the hysteresis to be added to
                fttmMaxAS[x][y] (see 14.23.13) for the times of the two input FTTM input
                ClockTarget interfaces with index numbers x and y.";
            reference
              "IEEE Std 802.1ASed 14.23.7 ";
          }
        }
      }
    }

    grouping max-as-group {
      description
        "The fttmMaxAS[x][y] object gives the maximum magnitude of expected skew between
times provided by the FTTM input ClockTarget interfaces of index x and index y when those
times are not faulty. This value is used as the criteria to determine the trustworthiness
of the times being compared. See maxASxy in M.5";
      reference
        "IEEE Std 802.1ASed 14.23.10";
      list fttm-max-as-lists {
        key "fttm-input-index-number";
        description
          "The index x.";
        reference
          "IEEE Std 802.1ASed 14.23.10";
        leaf fttm-input-index-number {
          type uint8;
          description
            "The first FTTM input index number of the two-dimensional array.";
          reference
            "IEEE Std 802.1ASed 14.23.10";
        }
        list fttm-max-as-list {
          key "fttm-input-index-number";
          description
            "The index y.";
          reference
            "IEEE Std 802.1ASed 14.23.10";
          leaf fttm-input-index-number {
            type uint8;
            description
              "The second FTTM input index number of the two-dimensional array.";
            reference
              "IEEE Std 802.1ASed 14.23.10";
          }
          leaf fttm-max-as {
            type uint32;
```

```
            units "2^-16 nanoseconds";
            default "0";
            description
              "The maximum magnitude of expected skew between times provided by the input
               ClockTarget interfaces of index x and index y when those times are
               not faulty.";
            reference
              "IEEE Std 802.1ASed 14.23.10";
          }
        }
      }
    }

    grouping fttm-tsf-algo-name-group {
      description
        "The fttmTsfAlgoName object is a vector of strings, where each vector member
  provides the name of the algorithm used by each active TSF instance number (the ITSF and
  the active DTSFs).";
      reference
        "IEEE Std 802.1ASed 14.23.20";
      list fttm-tsf-algo-name-list {
        key "tsf-instance-number";
        description
          "The fttmTsfAlgoName object is a vector of strings, where each vector member
  provides the name of the algorithm used by each active TSF instance number (the ITSF and
  the active DTSFs)";
        reference
          "IEEE Std 802.1ASed 14.23.20";
        leaf tsf-instance-number {
          type uint8;
          description
            "The TSF instance number for which the algorithm is to be queried.";
          reference
            "IEEE Std 802.1ASed 20.3.2.2";
        }
        leaf fttm-tsf-algo-name {
          type string;
          config false;
          description
            "Each fttmTsfAlgoName[x] member, where x is a value from 0 to
  fttmNumActiveDtsfs (see 14.23.13) and represents the TSF instance number, is a 128 octet
  that contains a string with the name of the algorithm for the TSF instance number.";
          reference
            "IEEE Std 802.1ASed 14.23.20";
        }
      }
    }

    grouping fttm-tsf-sel-time-index-group {
      description
        "The fttmTsfSelTimeIndex object gives the input time index that is selected by each
  TSF instance.";
      reference
        "IEEE Std 802.1ASed 14.23.21";
      list fttm-tsf-sel-time-index-list {
        key "tsf-instance-number";
        description
          "The fttmTsfSelTimeIndex object gives the input time index that is selected by
  each TSF instance.";
        reference
          "IEEE Std 802.1ASed 14.23.21";
        leaf tsf-instance-number {
          type uint8;
          description
            "The TSF instance number for which the selected time-index is to be queried.";
          reference
            "IEEE Std 802.1ASed 14.23.8";
        }
        leaf fttm-tsf-sel-time-index {
          type uint16;
          default "511";
```

```
          config false;
          description
            "Gives the input time index that is selected to be the output of the
             TSF instance.";
          reference
            "IEEE Std 802.1ASed 14.23.21";
        }
      }
    }

    augment "/ptp-tt:ptp/ptp-tt:common-services" {
      description
        "Augment IEEE Std 1588 commonServices with fault-tolerant-timing-module service.";
      container fault-tolerant-timing-module-service {
        if-feature "fttm";
        description
          "The Fault-Tolerant Timing Module Service structure contains the
           fttmSystemList, which is a list of instances of the Fault Tolerant
           Timing Module Service.";
        reference
          "14.23 of IEEE Std 802.1AS";
        list fttm-system {
          key "fttm-system-index";
          description
            "Indexed list of FTTM systems in the FTTM Service";
          leaf fttm-system-index {
            type uint8;
            description
              "Index for the FTTM system.";
          }
          container fttm-system-ds {
            description
              "The fttmSystemDS describes the attributes of the
               respective instance of the Fault-Tolerant Timing Module Service.";
            reference
              "14.23 of IEEE Std 802.1AS";
            uses fault-tolerant-timing-module-group;
          }
          container fttm-system-description-ds {
            description
              "The fttmSystemDescriptionDS contains descriptive information for
               the respective instance of the Fault-Tolerant Timing Module Service.";
            reference
              "14.24 Fault Tolerant Timing Module System Description Parameter
               Data Set (fttmSystemDescriptionDS) of IEEE Std 802.1ASed";
            leaf user-description {
              type string {
                length "0..128";
              }
              description
                "Configuration description of the Fault-Tolerant Timing Module system.";
              reference
                "14.24.3 of IEEE Std 802.1ASed";
            }
          }
        }
      }
    }
  }
```

18. Hot Standby

19. Media-dependent-layer specification for IEEE 802.3 Clause 4 Media Access Control (MAC) operating in half-duplex

Insert the following clause.

20. Fault-tolerant timing with time integrity

20.1 General

It is important for some time-sensitive applications (e.g., aerospace networks, per IEEE P802.1DP) to consider fault tolerance, including availability and integrity of the synchronizing function, to provide reliable and trustworthy system behavior. Features of gPTP that can be used to support fault-tolerant time synchronization include its provisions for multiple time domains, multiple GMs, multiple time distribution paths, and multiple gPTP instances per port in Bridges and end stations, and the external port configuration mode (see 10.3.1.3) that allows static time distribution paths to be established. The use of these features must be carefully considered by a system designer to ensure that the application's requirements for assured systems are met. For example, an aerospace network is typically expected to tolerate multiple (typically 2) simultaneous arbitrary faults in Bridges, end stations, links, and GMs to maintain availability and integrity of clock synchronization.

To achieve fault-tolerant timing with time integrity, this standard defines a Fault-Tolerant Timing Module (FTTM) for use with the fault tolerance supporting gPTP features listed above. The concepts used by the FTTM are described in 20.2. Details on the FTTM and its default operations are given in 20.3. General information about fault-tolerant timing with time integrity can be found in Annex J.

20.2 Fault-tolerant time synchronization concepts

20.2.1 Availability of time

The continuous availability of time is enhanced by redundancy. For gPTP, this redundancy can be implemented by using multiple time domains, multiple time distribution paths, and multiple gPTP instances in Bridges and end stations.

20.2.2 Trust and Integrity of time

For this standard, a trusted time is one that passes a specified criterion that identifies it as being within a safe bound of a non-faulty time and is, thus, safe to use. This establishment of trust gives integrity to the time.

For gPTP, trust, and hence integrity, can be established through the comparison of the times coming from independent time sources and the observation that they match within the specified criterion. See M.5 for an example of such a criterion.

20.2.3 Time agreement generation and preservation

Time agreement generation and preservation is the process by which multiple time source nodes (GMs) come to an agreement on the time and maintain that agreement in the presence of both faults and oscillator drift. This process preserves both the collective accuracy and relative precision of the set of GMs.

Time agreement generation and preservation should be done a manner that is resilient to faults, including Byzantine faults. See [A], [B], [C], [D], and [E].

20.2.4 Time agreement distribution

Time agreement distribution is the process of distributing the time established by time agreement generation from time source nodes (GMs) to time destination nodes (gPTP End Instances). Time agreement distribution is performed using gPTP, per the models described in clause 7 and the mechanisms specified in clauses 8 to 16 of this standard.

20.2.5 Dependent gPTP times

Dependent gPTP times share one or more common time-influencing components. This could be a common GM, continuously synchronized GMs, GMs that share a common (continuously connected) ClockSource, or a common gPTP Relay Instance.

Because dependent gPTP times share one or more common influencers, they do not, on their own, enable end-to-end integrity checking of the time synchronization function. However, they can be used to improve the availability of a given time source and can provide partial integrity checks. For example, an application that receives timing from a single GM through more than one redundant synchronization trees has increased availability of that GM's time and can check the integrity of the synchronization trees by comparing the time received from them. However, because the time originates from a single GM, the integrity of that GM's time cannot be confirmed and, thus, end-to-end integrity of the time synchronization function is not achieved.

When a set of dependent gPTP times is used in combination with other gPTP times, which are independent (see 20.2.6), the set of dependent gPTP times can be reduced to a single independent gPTP time and used to enhance the ability to achieve end-to-end integrity of the time synchronization function. This operation is performed by the Fault-Tolerant Timing Module (see 20.3).

Dependent gPTP times can be identified by one of the following methods:

— They have the same gPTP domainNumber, majorSdoID, and minorSdoID. This indicates that gPTP messages from the same gPTP GM are received by two gPTP End Instances, on two distinct physical ports, that are serviced by the FTTM.

— They have different gPTP domainNumbers but the same gmtimeBaseIndicator. This indicates that the gPTP messages come from different gPTP GMs that share the same clockSource.

— They have gPTP domainNumbers that are defined by a management entity, which is out of scope of this standard, to be dependent.

NOTE—Faults that cause the masquerading of any of the above gPTP fields can be mitigated by the Fault-Tolerant Timing Module (see 20.3).

In a network that supports fault-tolerant timing with time integrity, the gmTimeBaseIndicator shall be made unique across all ClockSources in the network.

NOTE—A network management entity (outside the scope of this standard) could be used to ensure that gmtimeBaseIndicator is unique across all clock sources present.

20.2.6 Independent gPTP times

Independent gPTP times do not share any common time-influencing components with each other and, therefore, deliver independent time values. Independent gPTP times are, by definition, from different domains.

Because independent gPTP times do not share any common influencers, they can enable end-to-end integrity checking of the time synchronization function, provided their times track sufficiently closely. Independent gPTP domains need to be aligned to each other in a manner that is resilient to faults (i.e., achieve time agreement and preservation, see 20.2.3) For example, time agreement mechanisms can be used to align the clocks of two independent GMs.

Because the independent gPTP domains are synchronized to each other, they provide a redundant source of time to the end application and, thus, also improve the availability of the time synchronization function to the end application.

20.3 Fault-Tolerant Timing Module

To enable fault-tolerant timing with time integrity, a Fault-Tolerant Timing Module (FTTM) operating at the application layer, per Clause 9, is to be implemented in all time-aware bridges and end stations that receive multiple gPTP times. The FTTM manages the selection of a time source from amongst two or more gPTP times (and gPTP instances) to support increased availability (see 20.2.1) and integrity (see 20.2.2). The FTTM also supports single domain solutions but, in this scenario, it does not provide any enhancements for increased availability or integrity.

The availability and integrity scenarios supported by the FTTM are listed below.

- Enhanced availability and integrity scenario:

  This scenario operates with multiple times and multiple domains.

  In this mode, the FTTM supports increased availability and integrity.

- Enhanced availability and limited integrity scenario:

  This scenario operates with multiple times and a single domain.

  In this mode, the FTTM supports increased availability and potential time distribution integrity, but not GM integrity.

- Regular availability and no integrity scenario:

  This scenario operates with a single time and, hence, a single domain.

  In this mode, the FTTM does not support increased availability or integrity.

The application is expected to be aware of the availability and integrity scenario in which the FTTM is operating under and its effects on the integrity of the FTTM's output.

Figure 4 illustrates the FTTM operating with three gPTP instances. The FTTM can also use the local oscillator's clock (OSC_CLK) as an input to its selection state machine.

Because the FTTM resides between gPTP Instances and a ClockTarget, its effect is localized. This allows custom algorithms that can better service specific timing characteristics of the network to be used by the FTTM. A default algorithm that can be used by the Time Selection Function (TSF) instances (see 20.3.2) in the FTTM is defined in 20.3.5.

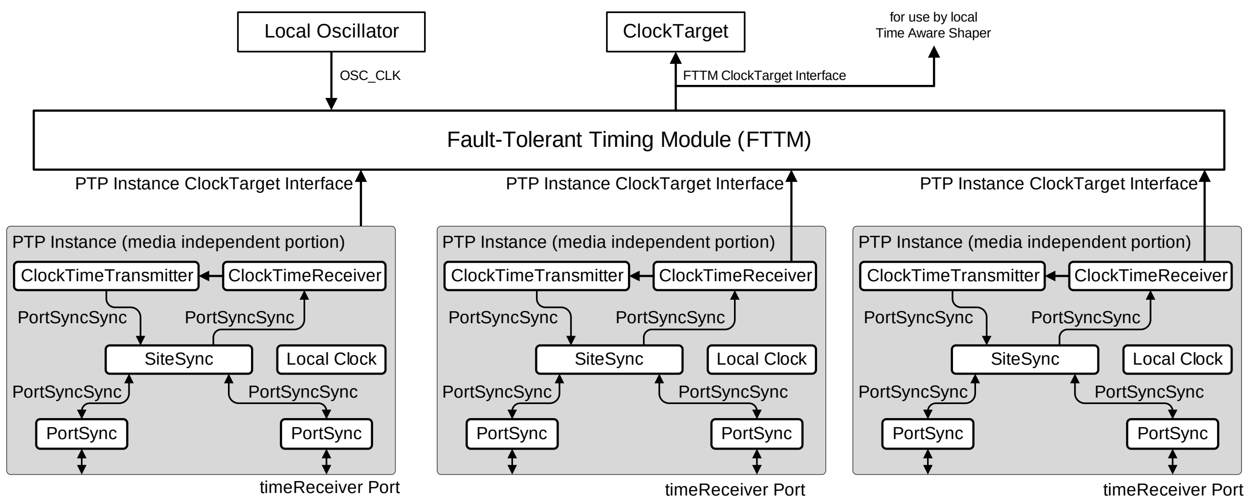


Figure 4— Fault-Tolerant Timing Module in operation

20.3.1 Scope and assumptions

The following list provides the detailed assumptions and goals for the FTTM:

— A fault-tolerant network (with time integrity) and its configuration are static during normal operation.
— All gPTP ports are configured using the external port configuration provision (i.e., the BTCA is not used).
— There is no administrative reconfiguration during run-time in the event of faults.
— While operation with one time and one domain is supported by the FTTM, operation with more than one time and more than one domain is required to enable fault tolerant timing with time integrity. To support interoperability, a minimum number of times and domains needs to be specified for an application.
— gPTP times are recognized as being dependent or independent as defined in clause 20.2.5 and 20.2.6, respectively.

20.3.2 Functional description

The FTTM consists of the following functions:

— A local oscillator clock (OSC_CLK). The health of OSC_CLK needs to be good in order for its associated functions to operate properly. The checking and the maintenance of the OSC_CLK health is outside the scope of this standard.
— Input ClockTargetEventCapture application interfaces (see Clause 9.3) providing time information to the FTTM, where gPTP End Instances serve as the ClockTimeReceiver entities and the Time Selection Functions (TSFs) within the FTTM serves as the ClockTarget entity. Time is passed to the TSFs via each ClockTarget application interface's timeReceiverTimeCallback parameter. The instanceIndex number associated with each gPTP End Instance is also passed to the TSFs.
— Zero or more instances of TSFs for servicing dependent times (Dependent Time Selection Functions, DTSFs).
— One instance of a TSF for servicing independent times (Independent Time Selection Function, ITSF).
— Zero or more instances of ClockTargetEventCapture application interface(s) (see clause 9.3) providing time information from DTSF(s) to the ITSF, where each DTSF serves as a ClockTimeReceiver entity and the ITSF serves as a ClockTarget entity. Time is passed to the ITSF via each ClockTargetEventCapture application interface's timeReceiverTimeCallback parameter. The instanceIndex number of the gPTP End Instance associated with the DTSF output ClockTargetEventCapture interface is also passed to the ITSF.
— Output ClockTargetEventCapture application interface (see clause 9.3) providing time information from the FTTM, where the FTTM's output (FTTM_OUTPUT) serves as the ClockTimeReceiver entity to the application's ClockTarget entity. Time is passed to the application's ClockTarget entity via the ClockTarget application interface's timeReceiverTimeCallback parameter. The instanceIndex number of the gPTP End Instance associated with the Output ClockTargetEventCapture interface is also provided by the FTTM.

A functional block diagram of the FTTM is shown in Figure 5.

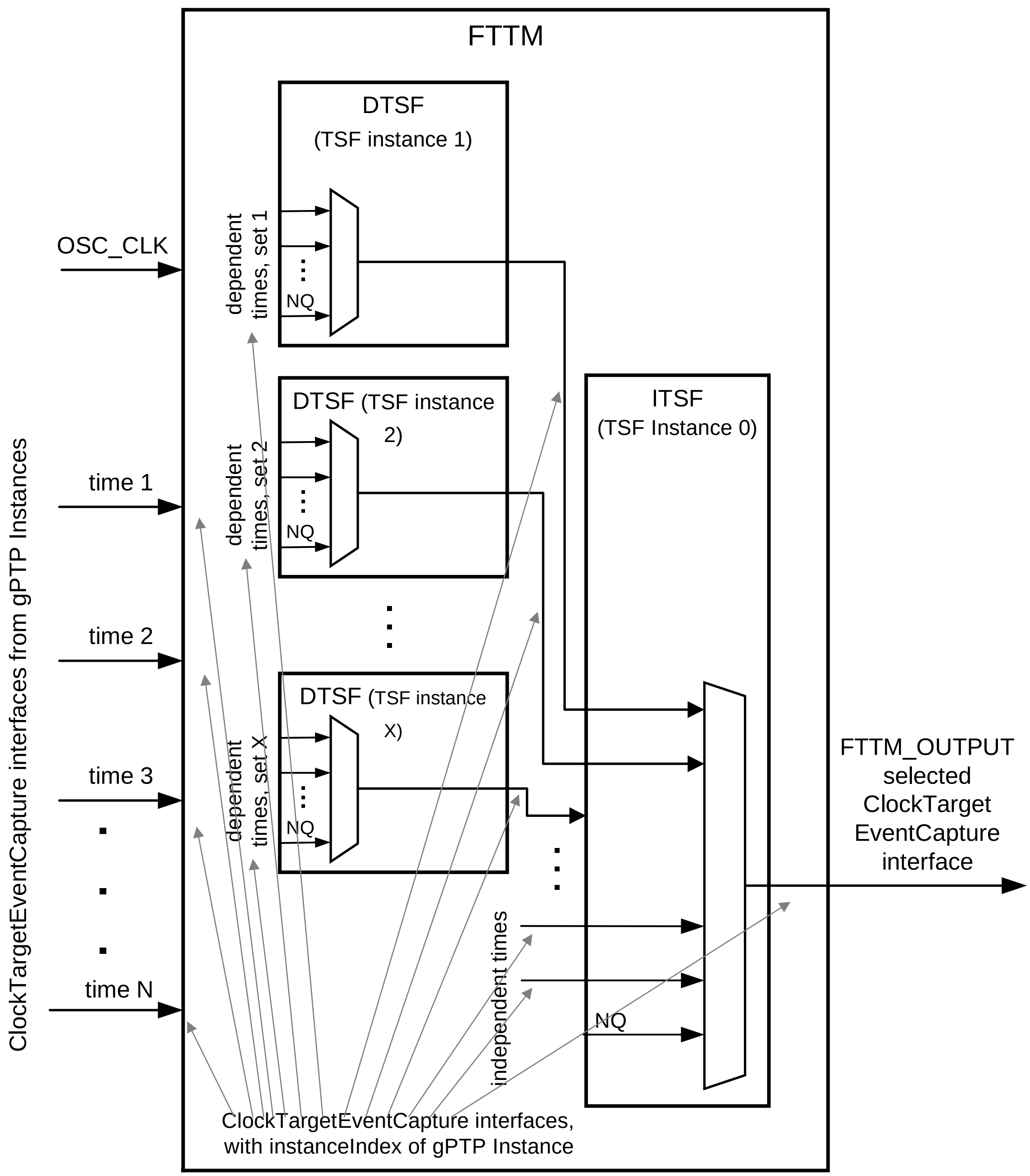


Figure 5—FTTM functional block diagram

When operating in the enhanced availability and integrity scenario (see 20.3):

— Each set of input times to the FTTM that share a common dependency shall be processed as a group by one DTSF instance. This grouping allows each DTSF instance to produce an output that is independent from the outputs of the other DTSFs and from the other input times to the FTTM.

— All the input times to the FTTM that share no dependency with any other time shall be processed by the ITSF.
— The output times of every DTSF instance shall be processed by the ITSF.
— The ITSF shall select one of its input times, or the NQ time if none of its input times can be determined to be trusted, as its result.

When operating in the enhanced availability and limited integrity scenario (see 20.3), it is expected that all the input times to the FTTM are connected to the ITSF and treated as if they are independent from each other (i.e., DTSFs are not used in this scenario). The ITSF shall select one of its input times, or the NQ time if none of its input times can be determined to be trusted, as its result.

When operating in the regular availability and no integrity scenario, (see 20.3), the FTTM shall always select the sole input time as its result.

The FTTM's local oscillator clock, OSC_CLK, may be used by the ITSF as a frequency reference to infer additional information about the qualities of the input times.

20.3.2.1 Input ClockTarget interfaces

The input ClockTarget interfaces to the FTTM are the output ClockTarget interface of the PTP Instances connected to the FTTM. They pass time information from the gPTP Instances to the TSFs of the FTTM. The FTTM accompanies each input ClockTarget interface with the instanceIndex number of the corresponding gPTP Instance.

The FTTM uses the ClockTargetEventCapture interface type (see clause 9.3). The use of other ClockTarget interface types (e.g., see clauses 9.4 and 9.5) with the FTTM is discussed in Annex N.

The FTTM's input ClockTarget interfaces can be for times that have a dependency with one or more times of other input ClockTarget interfaces or can be for times that have no dependency with the times of other input ClockTarget interfaces.

20.3.2.2 Time Selection Function (TSF)

Each Time Selection Function (TSF) instance has an embedded time-index selection algorithm that analyzes the TSF's set of input ClockTarget interfaces, determines which corresponding input time(s) can be trusted (see 20.2.2), and selects a trusted time and its corresponding gPTP Instance, if found, to be its output. If no trusted time is found, the algorithm selects the Not Qualified (NQ) time (see 20.3.2.3) for the TSF's output.

The state machine for the TSF is defined in 20.3.4.

One TSF instance in the FTTM is used for selecting a time from a set of independent times (see 20.2.6). This TSF is called the ITSF and is assigned the TSF instance number of zero.

If any of the input times to the FTTM are dependent (see 20.2.5), then one TSF instance is used to select a time for each set of dependent times. These TSFs are called DTSFs and are assigned TSF instance numbers from 1 to fttmNumActiveDtsfs (see 14.23.13).

The default time-index selection algorithm for the TSF is the mid-value time-index selection algorithm (MVTISA, see 20.3.5). Other algorithms, which are out of scope of this standard, can be used by any TSF instance. These other algorithms are not required to produce the same result as the default algorithm, MVTISA, or as each other to achieve time convergence.

Being trusted does not mean the time is non-faulty. However, as long as a time remains trusted (i.e., as long as its time continues to pass the specified criteria), it can be safely used by a DTSF or by the ITSF.

Any time that has an isSynced status (see 18.4.1.1) equal to FALSE or a gmPresent status (see 10.2.4.13) equal to FALSE is, without further consideration, declared to be untrusted.

20.3.2.3 Not Qualified (NQ) time

The Not Qualified (NQ) time is used to represent the condition where none of the input times to a TSF instance can be determined to be trusted. The NQ time contains the ClockTarget interface from any one of the input times (arbitrarily selected or implementation specific) being processed by the time-index selection algorithm but shall have the isSynced status (see 18.4.1.1) and the gmPresent status forced to FALSE, to indicate the untrusted condition.

NOTE—Because the NQ time is, by definition, not trusted, the values of its other parameters (aside from isSynced and gmPresent) have no functional impact.

All the possible parameters in the NQ time are listed below.

- domainNumber = the domainNumber from the arbitrarily selected time from the set of input times being processed by the algorithm
- timeReceiverTimeCallback = the timeReceiverTimeCallback value from the arbitrarily selected time
- isSynced = FALSE
- gmPresent = FALSE
- errorCondition = the errorCondition value from the arbitrarily selected time
- clockPeriod = the clockPeriod value from the arbitrarily selected time
- timeReceiverCallbackPhase = the timeReceiverCallbackPhase value invoked by the ClockTarget entity of the ClockTarget interface
- grandmasterIdentity = the grandmasterIdentity value from the arbitrarily selected time
- gmTimeBaseIndicator = the gmTimeBaseIndicator value from the arbitrarily selected time
- lastGmPhaseChange = the lastGmPhaseChange value from the arbitrarily selected time
- lastGmFreqChange = the lastGmFreqChange value from the arbitrarily selected time

20.3.2.4 Output ClockTarget Interfaces

The output ClockTarget interface from the FTTM is a reference to the ClockTarget interface of the PTP Instance selected by the TSFs in the FTTM.

The FTTM accompanies the output ClockTarget interface with the instanceIndex number of the selected gPTP Instance.

The FTTM uses the ClockTargetEventCapture interface type (see clause 9.3). The use of other ClockTarget interface types (e.g., see clauses 9.4 and 9.5) with the FTTM is discussed in Annex N.

20.3.3 FTTM state machine

20.3.3.1 General

The FTTM state machine described in this subclause interacts with all the TSF state machines (instantiated in the FTTM's DTSF instances and in the FTTM's ITSF) to find and select a trusted input ClockTarget interface, if one exists, to present as the FTTM's output.

The TSF state machine is defined in 20.3.4.

The default time-index selection algorithm used in the TSF state machine is the MVTISA, which is defined in 20.3.5.

The FTTM uses the ClockTargetEventCapture interface type (see clause 9.3). The use of other ClockTarget interface types with the FTTM is discussed in Annex N.

20.3.3.2 FTTM State machine variables

20.3.3.2.1 fttmTimeInterfaceRateRatioOff

The variable fttmTimeInterfaceRateRatioOff[x] is equal to the configured magnitude of the offset ratio between the rate of the time arriving on input ClockTarget interface x to the rate of the FTTM's local clock, OSC_CLK that is allowed to remain in the FREQ_TRUSTED state

The value is expressed as the fractional frequency offset multiplied by 2^{41}, i.e., the quantity (|rate of incoming clock / rate of OSC_CLK – 1.0|)(2^{41}),

The variable fttmTimeInterfaceRateRatioOff is a vector of size numTimeIndexes of Uint32 values and the data type is as follows:

— Typedef Uint32 [numTimeIndexes] fttmTimeInterfaceRateRatio;

The value for each fttmTimeInterfaceRateRatioOff vector member is selected from the fttmTimeInterfaceRatioRatioOff management object vector (see 14.23.17) based on the mapping of FTTM input ClockTarget interfaces to the default ITSF, as determined by the fttmMapIndexToTsf management object (see 14.23.5).

20.3.3.2.2 itsfInstanceIndex

The itsfInstanceIndex variable contains the UIntger32 instanceIndex value of the PTP Instance that originally generated the ClockTargetEventCapture.result for the selected input Clock Target of the ITSF.

20.3.3.2.3 itsfSelTImeIndex

The itsfSelTimeIndex variable is the index number of the selected input ClockTarget interface of the ITSF. This ITSF's selected input ClockTarget interface is presented as the output ClockTarget interface of the FTTM.

The range of values ranges from 1 to numTimeIndexes and the value of 511 represents the NQ time (see 20.3.2.3).

20.3.3.2.4 itsfTrustState (fttmOutputTrustState)

The variable itsfTrustState gives the determined trust state from the ITSF state machine (see 20.3.4). Valid values are NOT_TRUSTED and TIME_TRUSTED.

20.3.3.2.5 fttmTrustState (fttmOutputTrustState)

The variable fttmTrustState gives the determined trust state of the FTTM. Valid values are NOT_TRUSTED, TIME_TRUSTED, and FREQ_TRUSTED.

20.3.3.2.6 prevFttmTrustState (fttmOutputTrustState)

The variable prevFttmTrustState gives the previously determined trust state of the FTTM. Valid values are NOT_TRUSTED, TIME_TRUSTED, and FREQ_TRUSTED.

20.3.3.2.7 rRatioOff (Uinteger32)

The variable rRatioOff is the offset of the ratio between the rate of the time provided on the output of the ITSF to the rate of the FTTM's local clock, OSC_CLK. The value is expressed as a fractional frequency offset multiplied by 2^{41}, i.e., the quantity (rate of incoming clock / rate of OSC_CLK – 1.0)(2^{41}), The magnitude of this offset is one of the considerations used by FTTM state machine to determine whether it enters the FREQ_TRUSTED state or the NOT_TRUSTED state. See 20.3.3.3.

20.3.3.2.8 rRatioSdOff (Uinteger32)

The variable rRatioSdOff is the standard deviation of the offset of the ratio between the rate of the time provided on the output of the ITSF to the rate of the FTTM's local clock, OSC_CLK. It is one of the considerations used by the FTTM state machine to determine whether it enters the FREQ_TRUSTED state or the NOT_TRUSTED state. See 20.3.3.3.

20.3.3.3 FTTM State diagram

The FTTM state machine is shown in Figure 6.

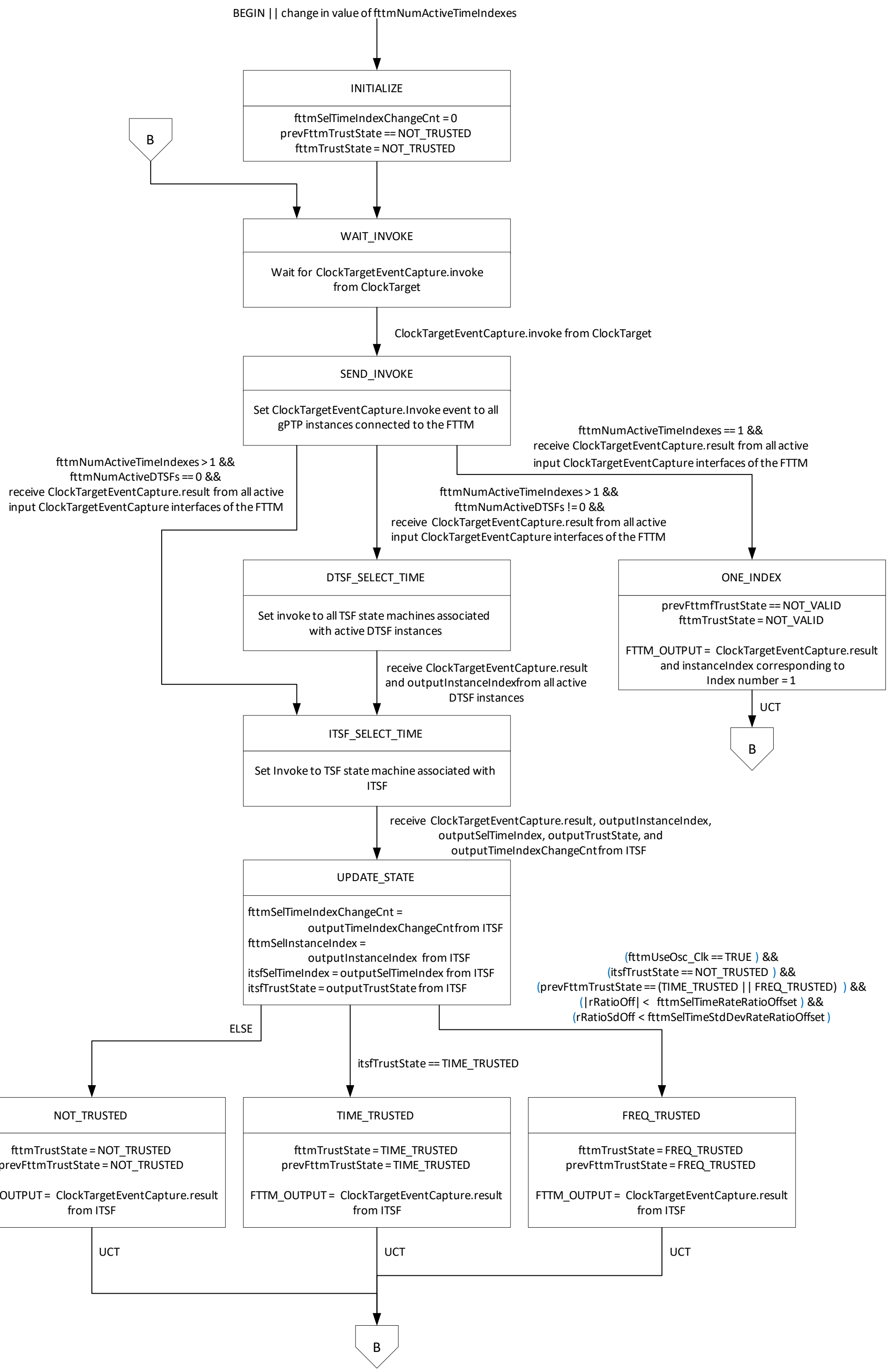


Figure 6—FTTM state machine

The INITIALIZE state of the FTTM state machine resets the dtsfEval and itsfEval variables to FALSE. This prepares them for future initiation of a default DTSF and default ITSF selection evaluation, respecdtively.

The WAIT_INVOKE state of the FTTM state machine waits for a ClockTargetEventCapture.invoke event from the ClockTarget. When the event is received, the state machine transfers to the SEND_INVOKE state.

The SEND_INVOKE state of the FTTM state machine simultaneously sends a ClockTargetEventCapture.invoke event to all the FTTM's input ClockTarget interfaces to request new timing information from the PTP Instances that are connected to the FTTM. Once all the input ClockTarget interfaces respond to the invoke event with ClockTargetEventCapture.result, this state transitions to one of the ONE_INDEX, DTSF_SELECT_TIME, or ITSF_SELECT_TIME states depending on the value of fttmNumActiveTimeIndexes and fttmNumActiveDTSFs.

If fttmNumActiveTimeIndexes is equal to 1, the state machine moves to the ONE_INDEX state. This state simply transfers the ClockTargetEventCapture.result from the input port of the FTTM to the output port of the FTTM. This state then transitions unconditionally back to the WAIT_INVOKE state.

If fttmNumActiveTimeIndexes is greater than 1 and fttmNumActiveDTSFs is not equal to 0, the state machine transitions to the DTSF_SELECT_TIME state of the FTTM state machine. This state sends a ClockTargetEventCapture.invoke to all the active DTSF instances in the FTTM. This causes the TSF state machine (see 20.3.4) in each of the active DTSF instances to perform another round of searching for and selecting, if available, a valid dependent time. Once every active DTSF instance responds to the invoke event with ClockTargetEventCapture.result and outputInstanceIndex, this state transitions to the ITSF_SELECT_TIME state.

If fttmNumActiveTimeIndexes is greater than 1 and fttmNumActiveDTSFs is equal to 0, the state machine transitions to the ITSF_SELECT_TIME state of the FTTM state machine. This state sends a ClockTargetEventCapture.invoke to the ITSF. This causes the TSF state machine (see 20.3.4) in the ITSF to perform another round of searching for and selecting, if available, a valid independent time. Once the ITSF responds to the invoke event with ClockTargetEventCapture.result, outputInstanceIndex, outputSelTimeIndex, outputTrustState, and outputTimeIndexChangeCnt, this state transitions to the UPDATE_STATE state.

The UPDATE_STATE state of the FTTM state machine updates is local variables with the outputs from the ITSF. Based on the outputTrustState from the ITSF, this state will transition to either the NOT_TRUSTED, the TIME_TRUSTED, or the FREQ_TRUSTED states.

If the ITSF produced a trust state of TIME_TRUSTED, the state machine transitions to the TIME_TRUSTED state. In this state, the FTTM state machine sets its trust state variables and management objects to TIME_TRUSTED and passes the ITSF's ClockTargetEventCapture.result to its output. The state machine then transitions back to the WAIT_INVOKE state.

If the ITSF produced a trust state of NOT_TRUSTED and the FTTM's previous trust state was either TIME_TRUSTED or FREQ_TRUSTED and the currently selected time has a rateRatio (relative to the FTTM's OSC_CLK) and a rateRatio standard deviation that is within configured thresholds and the fttmUseOscClk managed object is TRUE, then the state machine transitions to the FREQ_TRUSTED state. In this state, the FTTM state machine sets its trust state variables and management objects to FREQ_TRUSTED and passes the ITSF's ClockTargetEventCapture.result to its output. The state machine then transitions back to the WAIT_INVOKE state.

If neither of the conditions for the FTTM state machine to go to the TIME_TRUSTED or FREQ_TRUSTED states were met, then the state machine transitions to the NOT_TRUSTED state. In this state, the FTTM state machine sets its trust state variables and management objects to NOT_TRUSTED and passes the ITSF's ClockTargetEventCapture.result to its output. In this state, the

ClockTargetEventCapture.result will have its isSynced and gmPresent parameters set to FALSE. The state machine then transitions back to the WAIT_INVOKE state.

20.3.4 Time Selection Function (TSF) state machine

20.3.4.1 General

The TSF state machine described in 20.3.4 is individually instantiated in each DTSF instance and in the ITSF to perform the time index selection for the corresponding function.

The TSF state machine calls a time-index selection algorithm (see the SELECT_TIME_INDEX state in Figure 7). The default time-index selection algorithm for TSFs is the mid-value time-index selection algorithm (MVTISA), which is described in 20.3.5.

20.3.4.2 TSF state machine variables

The TSF variables described in this subclause are unique to each TSF instance in the FTTM and its corresponding time-index selection algorithm (e.g., MVTISA in 20.3.5). Some are local to the TSF instance and some, as indicated in the variables' descriptions, are passed down to the TSF from the FTTM state machine (see 20.3.3), derived from management objects (see 14.23), derived from ClockTargetEventCapture.result, and/or derived from ClockTargetEventCapture.result parameters (see 9.3.3) associated with the input ClockTargetEventCapture interfaces connected to the TSF.

20.3.4.2.1 gmIdentityStatus

The variable gmIdentityStatus is a vector of ClockIdentity. The vector member gmIdentityStatus[x] holds the grandmasterIdentity status from the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance, where x ranges from 1 to numTimeIndexes.

For an instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef ClockIdentity [numTimeIndexes] gmIdentityStatus;

20.3.4.2.2 hystVal

The variable hystVal[x][y] holds the hysteresis added to maxAsVal[x][y] (see 20.3.4.2.3) for the times of the two input ClockTargetEventCapture interfaces, with index numbers x and y, on the TSF instance. The hysteresis enables the use of one time skew level to set the trust status and another time skew level to clear the trust status.

The variable hystVal[x][y] is a two-dimensional array of UInteger32 values, each in units of 2^{-16} nanoseconds with a size of numTimeIndexes in each dimension. The range of x and of y is from 1 to numTimeIndexes.

The data type of hystVal is as follows:

— Typedef UInteger32 [numTimeIndexes][numTimeIndexes] hystVal;

The value for each hystVal array member is selected from an array member from the fttmHyst management object array (see 14.23.7) based on the mapping of FTTM input Clock Target interfaces to the instance of the TSF, as determined by the fttmMapIndexToTsf management object (see 14.23.5) and (for the ITSF only) the fttmMapDtsfToItsf management object (see 14.23.4).

20.3.4.2.3 maxAsVal

The value maxAsVal[x][y] is used as the criteria to determine the trustworthiness of the times from ClockTarget interfaces x and y when they are compared to each other. See $maxAS_{xy}$ in M.5.

The variable maxAsVal[x][y] is a two-dimensional array of UInteger32 values, each in units of 2^{-16} nanoseconds with a size of numTimeIndexes in each dimension. The data type is as follows:

— Typedef UInteger32 [numTimeIndexes][numTimeIndexes] maxAsVal;

The value for each maxAsVal array member is selected from an array member from the fttmMaxAS management object array (see 14.23.10) based on the mapping of FTTM input Clock Target interfaces to the TSF instance, as determined by the fttmMapIndexToTsf management object (see 14.23.5) and (for the ITSF only) the fttmMapDtsfToItsf management object (see 14.23.4).

20.3.4.2.4 numTimeIndexes (UInteger8)

The variable numTimeIndexes contains the number of input ClockTarget interfaces assigned to the TSF instance, where the assignment is done via the fttmMapIndexToTsf management object (see 14.23.5) and (for the ITSF only) the fttmMapDtsfToItsf management object (see 14.23.4).

20.3.4.2.5 outputInstanceIndex (Uinteger32)

The variable outputInstanceIndex gives the instanceIndex number of the PTP Instance of the ClockTarget interface selected by the TSF.

20.3.4.2.6 outputSelTimeIndex (fttmSelectedIndex)

The outputSelTimeIndex variable is the index number of the selected input ClockTarget interface of the TSF instance. This selected input ClockTarget interface is presented as the output ClockTarget interface of the instance of the TSF.

For any instance of the TSF, the range of values for the input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of 511 represents the NQ time (see 20.3.2.3).

20.3.4.2.7 outputTimeIndexChangeCnt (UInteger16)

The variable outputTimeIndexChangeCnt gives the number of times the TSF has changed its time index selection. The value increments every time the TSF changes its time index selection and rolls over to 0 if the value is incremented when its current value is 65535.

20.3.4.2.8 outputTrustState (fttmOutputTrustState)

The variable outputTrustState gives the determined trust state of the TSF instance’s state machine. Valid values are NOT_TRUSTED and TIME_TRUSTED.

20.3.4.2.9 prevSelTimeIndex (fttmSelectedIndex)

This prevSelTimeIndex variable is the index number of the previously selected input ClockTarget interface of the TSF instance. This previously selected input ClockTarget interface may be used as a condition in the time-index selection algorithm.

For any instance of the TSF, the range of values for the selected input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of 511 represents the NQ time (see 20.3.2.3).

20.3.4.2.10 prevTimeIndexStatus

The variable prevTimeIndexStatus is a vector that holds the previous trust status of a ClockTarget interface of the TSF instance. The variable prevTimeIndexStatus[x] holds the previous trust status of ClockTarget interface x. Valid values are NOT_TRUSTED and TRUSTED.

The data type is as follows:

— Typedef fttmInputTrustStatus [numTimeIndexes] prevTimeIndexStatus;

20.3.4.2.11 prevTimeIndexPairStatus

The variable prevTimeIndexPairStatus is a 2-dimensional array that holds the previous trust status between two ClockTarget interfaces of the instance of the TSF. The variable prevTimeIndexPairStatus[x][y] holds the previous trust status between ClockTarget interface x and ClockTarget interface y. Valid values are NOT_TRUSTED and TRUSTED.

The data type is as follows:

— Typedef fttmInputTrustStatus [numTimeIndexes][numTimeIndexes] prevTimeIndexPairStatus;

20.3.4.2.12 prevTrustState (fttmOutputTrustState)

The variable prevTrustState gives the previously determined trust state of the TSF instance’s state machine. Valid values are NOT_TRUSTED and TIME_TRUSTED for the default time-index selection algorithm, MVTISA (see 20.3.5).

20.3.4.2.13 savedDomainNumber

The variable savedDomainNumber is a vector of UInteger8. The vector member savedDomainNumber[x] holds the domainNumber status from the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance, where x ranges from 1 to numTimeIndexes.

For an instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef UInteger8 [numTimeIndexes] savedDomainNumber;

20.3.4.2.14 savedGmIdentity

The variable savedGmIdentity is a vector of ClockIdentity. The vector member savedGmIdentity[x] holds the clockIdentity of the Grandmaster PTP Instance from the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance, where x ranges from 1 to numTimeIndexes.

For an instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef ClockIdentity [numTimeIndexes] savedGmIdentity;

20.3.4.2.15 savedGmPresent

The variable savedGmPresent is a vector of Boolean. The vector member savedGmPresent[x] holds the gmPresent status from the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance, where x ranges from 1 to numTimeIndexes.

For an instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] savedGmPresent;

20.3.4.2.16 savedInstanceIndex

The variable savedInstanceIndex is a vector of UIntger32. The vector member savedInstanceIndex[x] holds the instanceIndex value of the PTP Instance that originally generated the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance, where x ranges from 1 to numTimeIndexes.

For an instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef UInteger32 [numTimeIndexes] savedInstanceIndex;

20.3.4.2.17 savedIsSynced

The variable savedIsSynced is a vector of Boolean. The vector member savedIsSynced[x] holds the isSynced status from the ClockTargetEventCapture.result for Clock Target interface x of the TSF instance. The range of x is from 1 to numTimeIndexes.

For each instance of the TSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] savedIsSynced;

20.3.4.2.18 savedTimeCallback

The variable savedTimeCallback is a vector of extended timestamps. The vector member savedTimeCallback[x] holds the latest timeReceiverTimeCallback result for input ClockTarget interface x.

The vector is of size numTimeIndexes and the data type is as follows:

— Typedef ExtendedTimestamp [numTimeIndexes] savedTimeCallback;

20.3.4.2.19 selTimeIndex (fttmSelectedIndex)

The selTimeIndex variable is the index number of the selected input ClockTarget interface of the TSF instance. This selected input ClockTarget interface is presented as the output ClockTarget interface of the instance of the TSF.

For any instance of the TSF, the range of values for the input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of 511 represents the NQ time (see 20.3.2.3).

20.3.4.2.20 selTimeIndexChangeCnt (UInteger16)

The variable selTimeIndexChangeCnt gives the number of times the TSF has changed its time index selection. The value increments every time the TSF changes its time index selection and rolls over to 0 if the value is incremented when its current value is 65535.

20.3.4.2.21 TimeIndexStatus

The variable TimeIndexStatus is a vector that holds the trust status of a ClockTarget interface of the TSF instance. The variable TimeIndexStatus[x] holds the trust status of ClockTarget interface x. Valid values are NOT_TRUSTED and TRUSTED.

The data type is as follows:

— Typedef fttmInputTrustStatus [numTimeIndexes] TimeIndexStatus;

20.3.4.2.22 TimeIndexPairStatus

The variable TimeIndexPairStatus is a 2-dimensional array that holds the trust status between two ClockTarget interfaces of the instance of the TSF. The variable TimeIndexPairStatus[x][y] holds the trust status between ClockTarget interface x and ClockTarget interface y. Valid values are NOT_TRUSTED and TRUSTED.

The data type is as follows:

— Typedef fttmInputTrustStatus [numTimeIndexes][numTimeIndexes] TimeIndexPairStatus;

20.3.4.2.23 trustState (fttmOutputTrustState)

The variable trustState gives the determined trust state of the TSF instance’s state machine. Valid values are NOT_TRUSTED and TIME_TRUSTED for the default time-index selection algorithm, MVTISA (see 20.3.5)..

20.3.4.2.24 x (UInteger8)

The variable x is a local variable used for looping functions in the TSF instance.

20.3.4.2.25 y (UInteger8)

The variable y is a local variable used for looping functions in the TSF instance.

20.3.4.3 TSF State diagram

The TSF state machine is shown in Figure 7. The TSF shall, by default, use the MVTISA of 20.3.5 as its time-index selection algorithm, which finds all the trusted input times and selects the index that corresponds to the ClockTarget interface that gives the trusted input time. The ClockTarget interface of the selected index shall be presented as the output of the TSF.

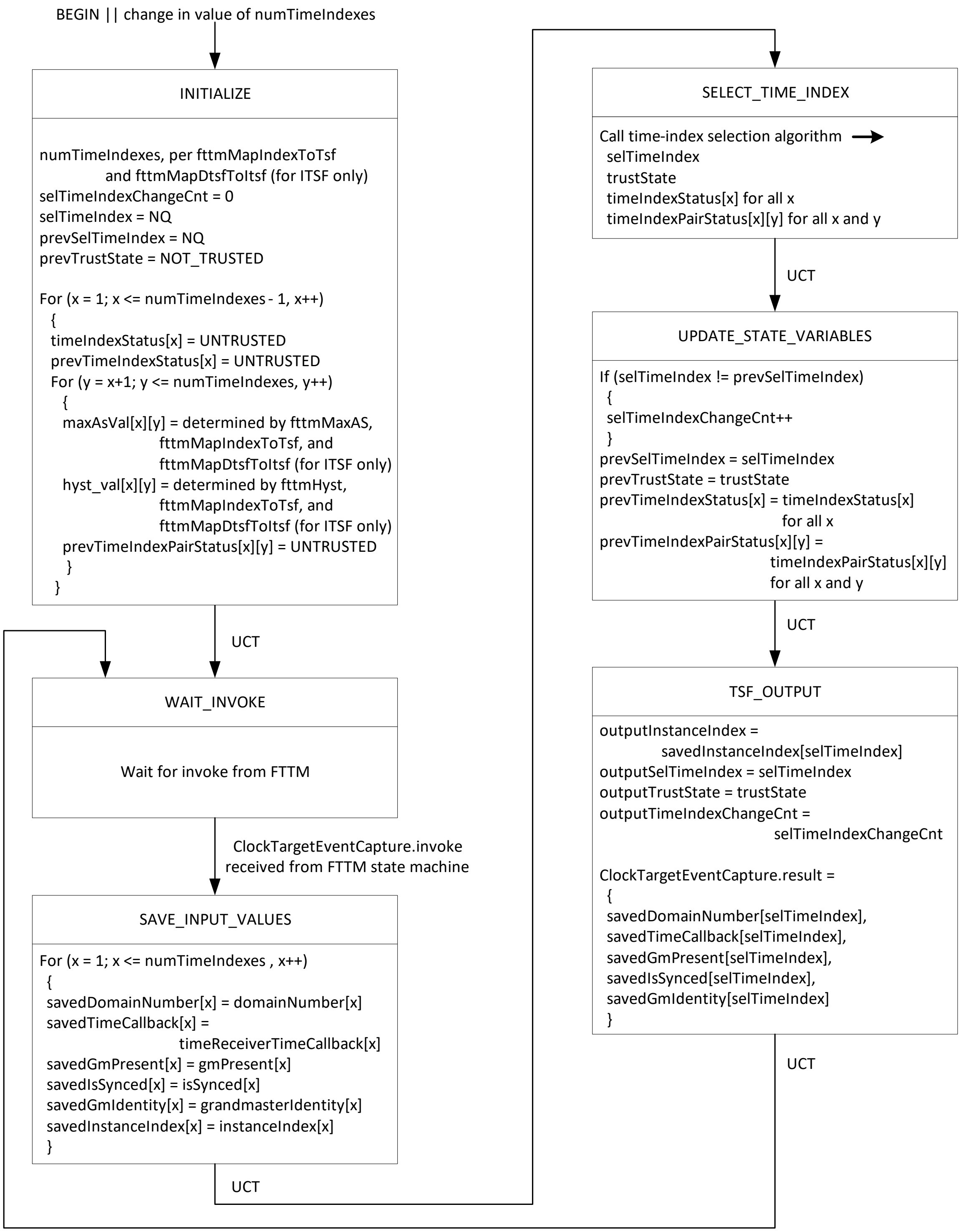


Figure 7— TSF state machine

The INITIALIZE state of the TSF state machine sets up the starting values of variables. Once the starting values are configured, this state machine unconditionally transfers to the WAIT_INVOKE state.

The WAIT_INVOKE state of the TSF state machine waits for an invoke event from the FTTM state machine. When this happens, the state machine transitions to the ASSIGN_INTF_RESULT_VARIABLES state.

The SAVE_INPUT_VALUES state of the TSF state machine assigns the timing result parameters (domainNumber, timeReceiverTimeCallback, gmPresent, isSynced, and grandmasterIdentity, see clause 9.3.3) from each of the input ClockTargetEventCapture interfaces to TSF variables. Once this assignment is finished, this state unconditionally transfers to the SELECT_TIME_INDEX state.

The SELECT_TIME_INDEX state of the TSF state machine calls a time-index selection algorithm (with the default algorithm being the MVTISA from 20.3.5) to find the trusted time, if any, from the input ClockTarget interfaces. If no trusted time is found, then the index number for the NQ time is identified. This state machine then unconditionally transfers to the UPDATE_STATE_VARIABLES state.

The UPDATE_STATE_VARIABLES state of the TSF state machine increments the selTimeIndexChangeCnt value, when appropriate, and updates the prevSelTimeIndex, prevTimeIndexStatus, prevTimeIndexPairStatus, and prevTrustState variables.

The TSF_OUTPUT state of the TSF state machine generates the ClockTargetEventCapture.result to the output of the TSF instance.

20.3.5 Mid-value time-index selection algorithm (MVTISA)

The mid-value trusted time-index selection algorithm (MVTISA) is the default time-index selection algorithm used for a TSF. Its name (see 14.23.20) is "MVTISA".

The MVTISA determines which time indexes have trusted times and then finds, amongst these time indexes with trusted times, the time index with the median time. The MVTISA shall be the default algorithm used by the TSF function.

The MVTISA looks at all possible combinations of input time index pairs to determine which pairs satisfy their specified maximum accepted skew magnitude threshold, $maxAs_{xy}$ (see M.5). All time indexes from any time index pair that satisfies its corresponding $maxAs_{xy}$ threshold is deemed to be trusted. The time index that has the median time amongst all the trusted time indexes is selected as the output of the MVTISA. If the number of trusted time indexes is even, the selected time index is the one with the smaller index value.

20.3.5.1 MVTISA variables

The variables local to the MVTISA are described in this subclause. Other variables used by the MVTISA are passed down to the MVTISA from the TSF that the MVTISA is embedded in or passed up from the MVTISA to the TSF that the MVTISA is embedded in. These variables are described in 20.3.4.2.

20.3.5.1.1 excludeTimeIndex

This variable excludeTimeIndex[x] is a vector, of size numTimeIndexes, of Boolean values that temporarily holds the exclusion status of the trusted input ClockTarget interfaces as they are sorted into ascending order. The data type is as follows:

— Typedef Boolean [numTimeIndexes] excludeTimeIndex;

When excludeTimeIndex[x] is TRUE, the input ClockTarget interface to the process with index x is excluded from the process.
When excludeTimeIndex[x] is FALSE, the input ClockTarget interface to the process with index x is not excluded from the process.

20.3.5.1.2 minValue (ExtendedTimestamp)

The variable minValue is a temporary value used by the MVTISA for sorting the times from the trusted ClockTarget interfaces, in time ascending order.

20.3.5.1.3 numSorted (UInteger8)

The variable numSorted holds a temporary count of the number of trusted time indexes that have been sorted in time ascending order.

20.3.5.1.4 orderedTimeIndex

The variable orderedTimeIndex is a vector of UInteger8 values, each of which correspond to an index number of a trusted input ClockTarget interface to the MVTISA. The vector member orderedTimeIndex[x] contains the Xth entry (starting from 1) of the trusted ClockTarget interface index numbers, ordered from lowest to highest ToD value.

The data type of is as follows:

— Typedef UInteger8 [numTimeIndexes] orderedTimeIndex;

20.3.5.1.5 TodDiff

The variable TodDiff is a 2-dimensional array, where the array member TodDiff[x][y] gives the magnitude of the time skew between the timeReceiverTimeCallback values of two ClockTarget interfaces, x and y, given as:

TodDiff[x][y] = | timeReceiverTimeCallback of ClockTarget Interface x –
timeReceiverTimeCallback of ClockTarget Interface y |.

For this process, each dimension of the array has a size of numTimeIndexes and the data type is as follows:

— Typedef ExtendedTimestamp [numTimeIndexes][numTimeIndexes] TodDiff;

20.3.5.1.6 x (UInteger8)

The variable x is a local variable used for looping functions.

20.3.5.1.7 y (UInteger8)

The variable y is a local variable used for looping functions.

20.3.5.2 MVTISA pseudo-code

Pseudo-code that represents the MVTISA is given below.

```
// ######################################################################
// Gather the current skews between the ToDs of all the time indexes.
// ######################################################################
For (x = 1; x <= numTimeIndexes - 1, x++) {
    For (y = x + 1, y <= numTimeIndexes, y++) {
        TodDiff[x][y] = |savedTimeCallback[x] - savedTimeCallback[y]|
        }
    }


// ######################################################################
// Clear status before starting a new round of time index comparisons.
// ######################################################################
trustState = NOT_TRUSTED
timeIndexPairStatus[x][y] = UNTRUSTED for all x and all y
timeIndexStatus[x] = UNTRUSTED for all x
numSorted = 1
excludeTimeIndex[x] = FALSE for all x


// ######################################################################
// Find all trusted time indexes, considering hysteresis.
// ######################################################################
For (x = 1, x <= numTimeIndexes - 1, x++) {
    For (y = x + 1, y <= numTimeIndexes, y++) {
        if ((TodDiff[x][y] <= maxAsVal[x][y] &&
             prevTimeIndexPairStatus[x][y] == UNTRUSTED) ||
             (TodDiff[x][y] <= (maxAsVal[x][y] + hystVal[x][y]) &&
              prevTimeIndexPairStatus[x][y] == TRUSTED)) &&
            (savedIsSynced[x] && savedGmPresent[x]) &&
            (savedIsSynced[y] && savedGmPresent[y]))
        {
             // trust found for the pair
             trustState = TIME_TRUSTED
             timeIndexPairStatus[x][y] = TRUSTED
             timeIndexStatus[x] = TRUSTED
             timeIndexStatus[y] = TRUSTED
        }
        else
        {
             // trust not found for the pair
        }
    }
}


// ######################################################################
// If trustState = TIME_TRUSTED, find time index with the mid-value ToD.
// ######################################################################
// Trusted times detected.
If {trustState == TIME_TRUSTED)
{
  // Sort all trusted time indexes in order of their ToD, from smallest
  // to largest, using two loops.
  // Outer loop iterates over all time indexes.
  For (x = 1, x <= numTimeIndexes, x++) {
    minValue = 2^48 seconds  // Start with ToD value that is larger
                             // than any possible gPTP ToD value.
```

```
        // Inner loop finds and records the time index with the minimum ToD
        // value and excludes it from further iterations of the outer loop.
        For (y = 1, y <= numTimeIndexes, y++) {
           if (timeIndexStatus[y] == TRUSTED &&
               excludeTimeIndex[y] == FALSE &&
               savedTimeCallback[y] <= minValue)
           {
             minValue = savedTimeCallback[y] // record latest min ToD value found
                                             // in the inner loop
             orderedTimeIndex[numSorted] = y  // record latest time index
                                              // with min ToD value
           }
        }
        // Exclude latest time index with the min ToD value and add to sort index.
        excludeTimeIndex[orderedTimeIndex[numSorted]] = TRUE
        numSorted = numSorted + 1
      }

      // Get median trusted time index.
      // The lower time index is selected if the number of trusted time
      // indexes is even.
          selTimeIndex = orderedTimeIndex[INT((numSorted)/2)]
    }

    // No time trust or frequency trust so set output time indexes to NQ
    // and clear previous trust status to NOT_TRUSTED.
    else
    {
      selTimeIndex = 511  // NQ
      trustState = NOT_TRUSTED
    }
```

Annex A (normative) Protocol Implementation Conformance Statement (PICS) proforma

Annex B (informative) Performance Requirements

Annex C (Informative) Timescales and epochs

Annex D Reserved for future use

Annex E Reserved for future use

Annex F (informative) PTP profile included in this standard

Annex G (informative) The asymmetry compensation measurement procedure based on line-swapping

Annex H (informative) Bibliography

Insert the following items alphabetically into the bibliography.

[A] L Lamport, PM Melliar-Smith, Synchronizing clocks in the presence of Faults, 1982.

[B] D Dolev, J Halpern, On the Possibility and Impossibility of Achieving Clock Synchronization, 1984.

[C] P Miner, A Geser, L Pike, Jeffery Maddalon, A Unified Fault-Tolerance Protocol, 2004.

[D] P Ramanathan, KG Shin, RW Butler, Fault-Tolerant Clock Synchronization in Distributed Systems, 1990.

[E] M Pease, R Shostak, L Lamport, Reaching Agreement in the Presence of Faults, 1980.

[F] IEEE Std 1588a^{TM}-2023, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Amendment 3: Precision Time Protocol (PTP) Enhancements for Best Master Clock Algorithm (BMCA) Mechanisms.

Insert the following annex.

Annex I (informative) Example Aerospace Configuration

Insert the following annex.

Annex J (informative) Time synchronization with Fault Tolerance and Integrity

J.1 Introduction

This Annex provides examples for time synchronization with fault tolerance and time integrity.

Time-sensitive applications that require fault tolerance are expected to tolerate multiple (typically 2) simultaneous arbitrary faults in end stations, Bridges, links, and GMs while maintaining availability and integrity of time synchronization.

Fault tolerance, or availability, and integrity address the reliable and accurate transmission of time values and the associated sync and follow-up messages in the presence of arbitrary faults in the network (link, Bridge, end station, and GM). Thus, under fault conditions, a correctly operating end station is expected to maintain a target maximum time error relative to the correctly operating GM. If unable to remain within the maximum time error, the correctly operating end station will detect an erroneous time synchronization state. To support this, it is expected that multiple clock domains are configured and managed in the network.

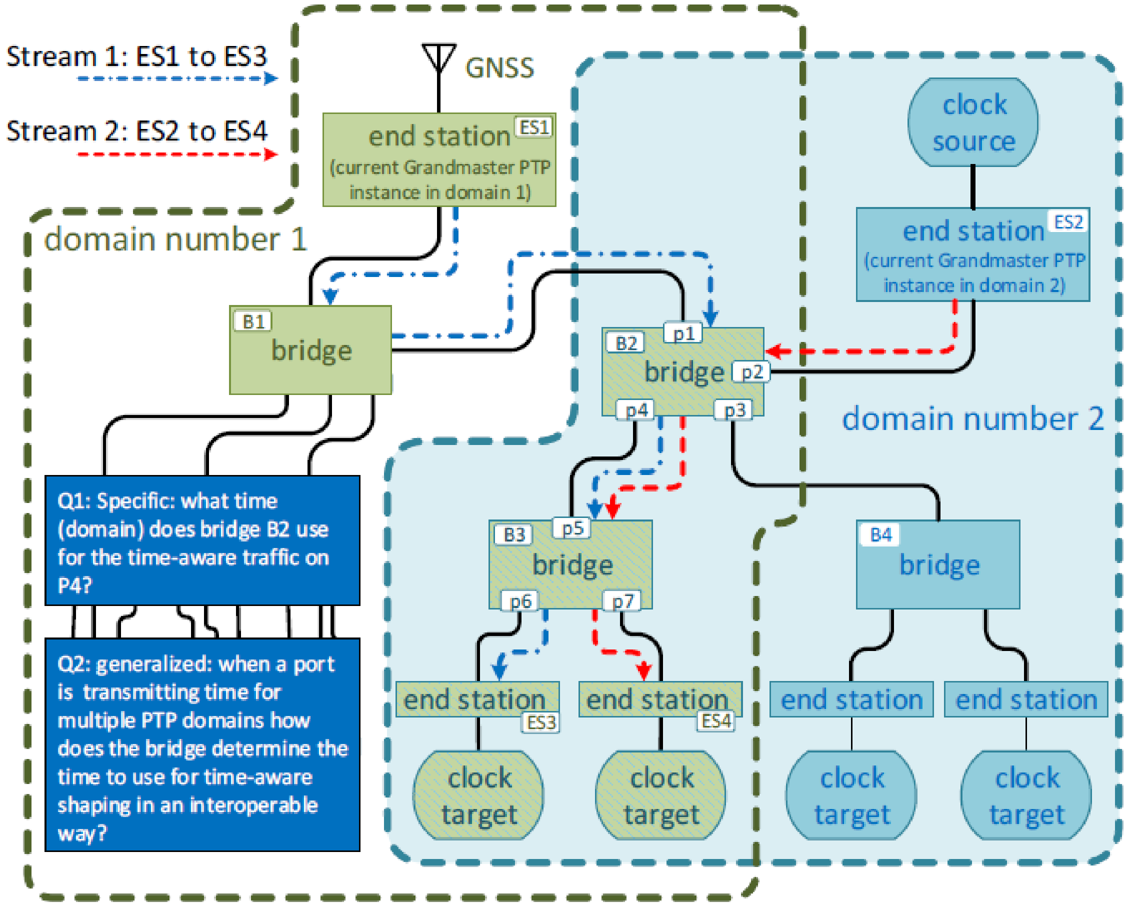
J.2 Clock domain management

As described in 20.2, clock domains can be considered dependent or independent. Independent clock domains, where ClockSources are independent, are expected to present problems for the enhancements for scheduled traffic function (8.6.8.4 of IEEE Std 802.1Q-2022) and for the PSFP stream gating function (8.6.5.4 of IEEE Std 802.1Q-2022), assuming both functions are referenced to gPTP time, because the transmission gates and the stream gates on a port only listen to one timer. This makes it problematic to bridge synchronized traffic between unsynchronized domains.

In Figure J-1, two clock domains D1 and D2 are shown overlapping at Bridges B2 and B3 with streams S1 and S2 sharing a common output port, P4 on B2, and a common input port, P5 on B3. If the two clock domains are synchronized, Bridges B2 and B3 will synchronize to the common time and will be able to forward both streams to the downstream end stations. However, if the two clock domains are not synchronized, then the transmission gate and stream gate opening and closing times for a given traffic class do not agree for both clock domains on the shared ports, causing violations on at least one clock domain.

It is not possible to support multiple unsynchronized transmission gate schedules or multiple stream gating schedules on a single port so networks using multiple PTP domains should ensure that the clock domains are either dependent on a common ClockSource or are synchronized to each other by some other means (see 20.2).

Management of multiple PTP instances using a fault-tolerant timing module (FTTM) is discussed in 20.3.



<Insert original Figure D-1 from P802.1DP/D1.1)

Figure J-1 – Multiple gPTP domains with shared port

J.3 Time agreement generation examples

PPS-based implementations.

TTE-based implementations.

J.4 Balancing availability and integrity

A common time source is used for all GMs.

Only 2 domains are used, instead of 3 or more.

An insufficient number of independent paths are available in the network for all PTP domains.

J.5 FTTM operation in example network topologies

A.1.1 N x point-to-point network (one for each gPTP domain)

A.1.2 Dual-homed network

A.1.3 Multi-homed network

A.1.4 Dual-star network

A.1.5 Bidirectional ring network

A.1.6 Mesh network

Insert the following annex.

Annex K

(informative)

FTTM Configuration Examples

This annex offers a detailed guide through a specific configuration example that utilizes simple YANG instance data that has been validated in accordance with the ieee802-dot1as-fttm module. The explanatory text provided between segments of XML instance data clarifies its alignment and connection with the FTTM service described in Clause 20.

K.1 Initial Setup

The example begins with a foundational XML structure that incorporates Precision Time Protocol (PTP) instance data, defined within the ptp element. This element is identified by its namespace urn:ieee:std:1588:yang:ieee1588-ptp-tt, indicating its adherence to the IEEE 1588 PTP standard tailored for telecommunications.

```
<ptp xmlns="urn:ieee:std:1588:yang:ieee1588-ptp-tt">
  <common-services>
    <fault-tolerant-timing-module-service
xmlns="urn:ieee:std:802.1AS:yang:ieee802-dot1as-fttm">
```

K.2 Fault-Tolerant Timing Module Management

Within this structure, we delve into the configuration of a Fault-Tolerant Timing Management (FTTM) system. The XML snippet introduces a fttm-system element, which houses a unique fault-tolerant-timing-function-index. This example illustrates a scenario with a single FTTM instance, identified by index 1.

```
      <fttm-system>
        <fttm-system-index>1</fttm-system-index>
```

K.3 FTTM Configuration Details

Further detailing the FTTM configuration, the fttm-system-ds element specifies the system's capabilities, including the maximum number of available input ClockTarget Interfaces (fttm-max-num-time-indexes), the maximum number of available DTSF instances (fttm-max-num-dtsfs), the maximum number of available input ClockTarget Interfaces on each DTSF instance (dtsf-max-num-time-indexes), and the ability of the FTTM to use a local oscillator for its time selection operations (fttm-use-osc-clk).

```
        <fttm-system-ds>
          <fttm-max-num-time-indexes>3</fttm-max-num-time-indexes>
          <fttm-max-num-dtsfs>1</fttm-max-num-dtsfs>
          <dtsf-max-num-time-indexes>2</dtsf-max-num-time-indexes>
          <fttm-use-osc-clk>true</fttm-use-osc-clk>
```

K.4 TSF Configuration

The configuration continues with the definition of TSFs within the FTTM, where the TSF instance that serves as the ITSF is identified as tsf-instance-number = 0 and where each TSF instance that serves as a DTSF is identified with a tsf-instance-number that ranges from 1 to fttm-num-active-dtsfs.

The time-index selection algorithm that is used by the two TSFs in this example are read, as shown below.

```
<fttm-tsf-algo-name-list>
  <tsf-instance-number>0</tsf-instance-number>
  <fttm-tsf-algo-name>MVTISA</fttm-tsf-algo-name>
</fttm-tsf-algo-name-list>
<fttm-tsf-algo-name-list>
  <tsf-instance-number>1</tsf-instance-number>
  <fttm-tsf-algo-name>MVTISA</fttm-tsf-algo-name>
</fttm-tsf-algo-name-list>
```

K.5 PTP Instance association to FTTM input indexes

The example then outlines the association of PTP Instances with the FTTM input ClockTarget interfaces using fttm-map-ptp-instance-to-index, emphasizing the unique indexing of each PTP Instance within the full ieee1588-ptp-tt module. The configuration of three PTP Instances, with instance indexes 123, 456, and 789 associated with FTTM input ClockTarget interfaces index numbers 1, 2, and 3, respectively, is illustrated in the code below and in Figure K.1.

```
<fttm-map-ptp-instance-to-index-list>
  <fttm-input-index-number>1</fttm-input-index-number>
  <instance-index>123</instance-index>
</fttm-map-ptp-instance-to-index-list>
<fttm-map-ptp-instance-to-index-list>
  <fttm-input-index-number>2</fttm-input-index-number>
  <instance-index>456</instance-index>
</fttm-map-ptp-instance-to-index-list>
<fttm-map-ptp-instance-to-index-list>
  <fttm-input-index-number>3</fttm-input-index-number>
  <instance-index>789</instance-index>
</fttm-map-ptp-instance-to-index-list>
```

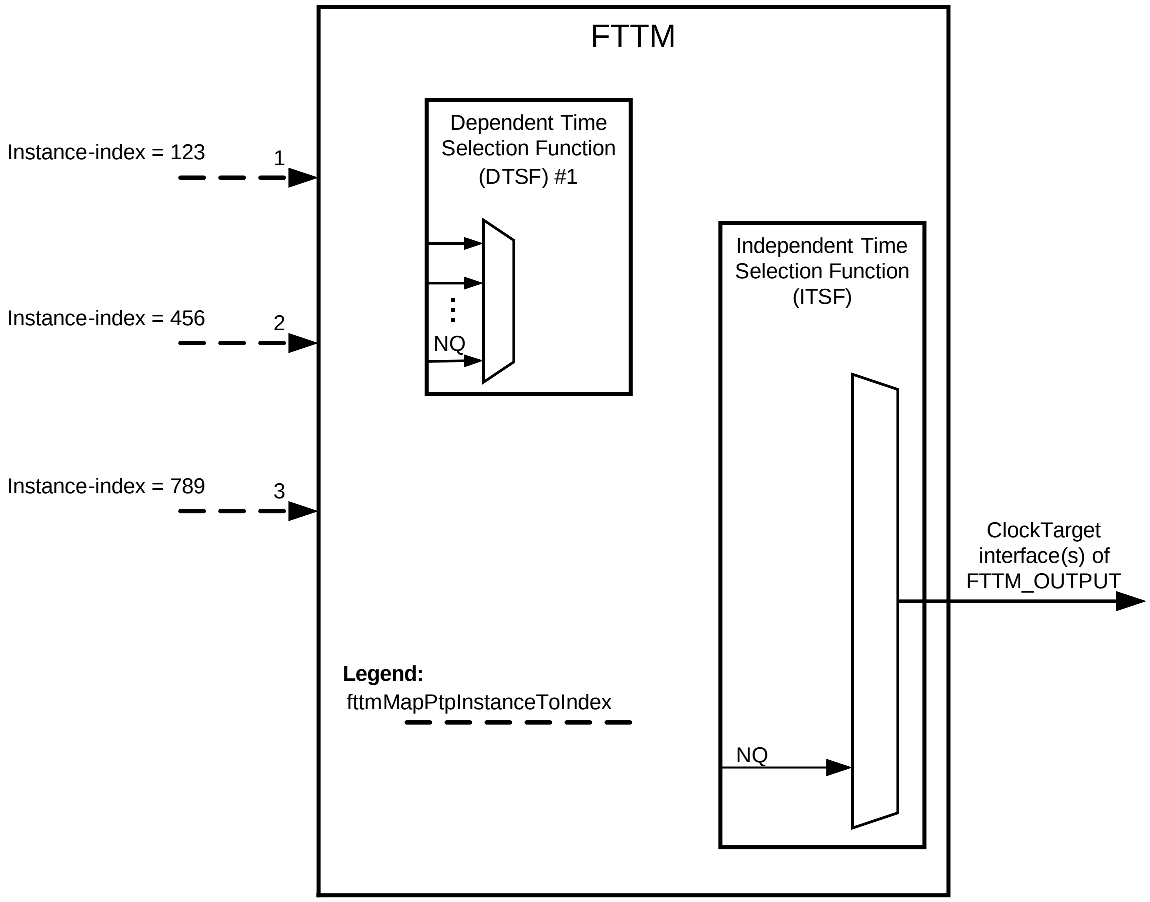


Figure K.1— fttmMapPtpInstanceToIndex example

K.6 FTTM input index connection to TSF input indexes

The example then outlines the connection of FTTM input indexes to TSF input indexes, using fttm-map-index-to-tsf, allowing the FTTM input ClockTarget Interfaces associated with the FTTM input indexes to be processed by an appropriate TSF; a DTSF (as a dependent time) or the ITSF (as an independent time).

The FTTM input index 1 is connected to input index 1 of TSF instance number 1 (a DTSF). The FTTM input index 2 is connected to index 2 of the same TSF. FTTM input index 3 is connected to input index 2 of TSF instance number 0 (the ITSF). This is illustrated in the following code and in Figure K.2.

The above connections insinuate that FTTM input indexes 1 and 2 share a dependence and that DTSF #1 will select between the two of them.

```
<fttm-map-index-to-tsf-list>
  <fttm-input-index-number>1</fttm-input-index-number>
  <tsf-instance-number>1</tsf-instance-number>
  <tsf-input-index-number>1</tsf-input-index-number>
</fttm-map-index-to-tsf-list>
<fttm-map-index-to-tsf-list>
  <fttm-input-index-number>2</fttm-input-index-number>
  <tsf-instance-number>1</tsf-instance-number>
  <tsf-input-index-number>2</tsf-input-index-number>
</fttm-map-index-to-tsf-list>
```

```
<fttm-map-index-to-tsf-list>
  <fttm-input-index-number>3</fttm-input-index-number>
  <tsf-instance-number>0</tsf-instance-number>
  <tsf-input-index-number>2</tsf-input-index-number>
</fttm-map-index-to-tsf-list>
```

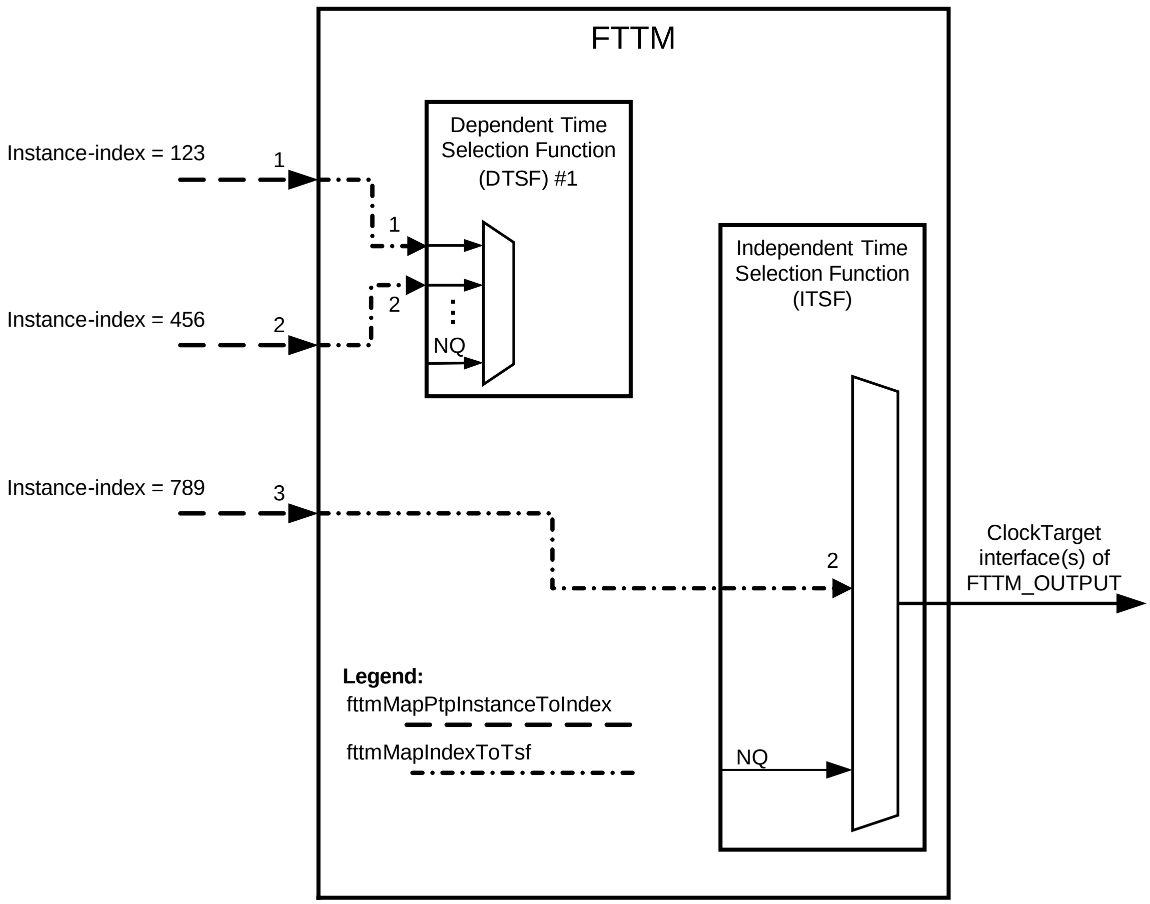


Figure K.2—fttmMapIndexToTsf example

K.7 DTSF outputs to ITSF input indexes

The example outlines the connection of DTSF output interfaces to ITSF input indexes, using fttm-map-dtsf-to-itsf, allowing the interface selected by the DTSF to undergo ITSF selection. Following the example from K.6, the output of DTSF #1 is to be connected to ITSF input index 1. This is illustrated in the code below and in Figure K.3.

With the addition of this connection, each of the 3 input ClockTarget Interfaces of the FTTM have a selectable path, via DTSF and ITSF state machines and selection algorithms, to the FTTM output ClockTarget Interface.

```
<fttm-map-dtsf-to-itsf-list>
   <tsf-instance-number>1</tsf-instance-number>
   <itsf-input-index-number>1</itsf-input-index-number>
</fttm-map-dtsf-to-itsf-list>
```

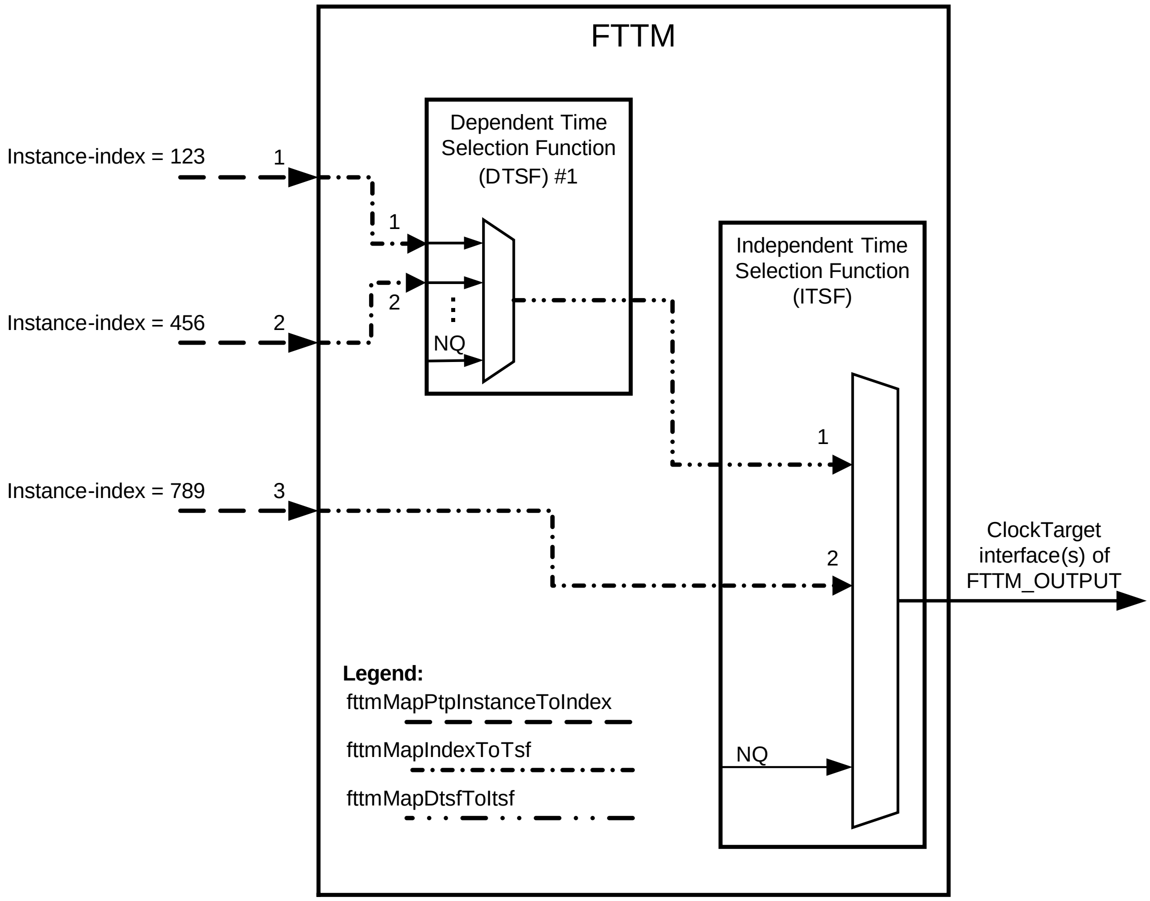


Figure K.3— fttmMapDtsfToItsf example

K.8 Inter-PTP Instance skew

The following XML instance data represents a two dimensional array holding the maximum magnitude of expected skew magnitude between times provided by the FTTM input ClockTarget interfaces of index x and index y when the times given by those interfaces are not faulty.

In this example, we have:

- $maxAS_{12}$= 11111 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 2 when the times given by those interfaces are not faulty.

- $maxAS_{13}$= 22222 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 3 when the times given by those interfaces are not faulty.

- $maxAS_{23}$= 33333 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 3 when the times given by those interfaces are not faulty.


```
<fttm-max-as-lists>
  <fttm-input-index-number>1</fttm-input-index-number>
  <fttm-max-as-list>
```

```
    <fttm-input-index-number>2</fttm-input-index-number>
    <fttm-max-as>11111</fttm-max-as>
  </fttm-max-as-list>
  <fttm-max-as-list>
    <fttm-input-index-number>3</fttm-input-index-number>
    <fttm-max-as>22222</fttm-max-as>
  </fttm-max-as-list>
</fttm-max-as-lists>
<fttm-max-as-lists>
  <fttm-input-index-number>2</fttm-input-index-number>
  <fttm-max-as-list>
    <fttm-input-index-number>3</fttm-input-index-number>
    <fttm-max-as>33333</fttm-max-as>
  </fttm-max-as-list>
</fttm-max-as-lists>
```


K.9 Hysteresis

The following XML instance data represents a two dimensional array holding the assigned hysteresis values, used on the skews detected on the FTTM input ClockTarget interfaces of index x and index y.

In this example, we have:

- $Hyst_{12}$= 999 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 2 when the times given by those interfaces are not faulty.

- $Hyst_{13}$= 888 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 3 when the times given by those interfaces are not faulty.

- $Hyst_{23}$= 777 x 2^{-16} ns, between times provided by the FTTM input ClockTarget interfaces of index x = 1 and index y = 3 when the times given by those interfaces are not faulty.


```
<fttm-hyst-lists>
  <fttm-input-index-number>1</fttm-input-index-number>
  <fttm-hyst-list>
    <fttm-input-index-number>2</fttm-input-index-number>
    <fttm-hyst>9999</fttm-hyst>
  </fttm-hyst-list>
  <fttm-hyst-list>
    <fttm-input-index-number>3</fttm-input-index-number>
    <fttm-hyst>8888</fttm-hyst>
  </fttm-hyst-list>
</fttm-hyst-lists>
<fttm-hyst-lists>
  <fttm-input-index-number>2</fttm-input-index-number>
  <fttm-hyst-list>
    <fttm-input-index-number>3</fttm-input-index-number>
    <fttm-hyst>7777</fttm-hyst>
  </fttm-hyst-list>
</fttm-hyst-lists>
```


K.10 FTTM RateRatio offsets

The following XML instance data represents the fttm-sel-time-rate-ratio-offset value and the fttm-sel-time-std-dev-rate-ratio-offset value. The former is equal to the maximum measured rate ratio offset of the normalized clock rate of the time arriving on the input interfaces to the normalized clock rate of the FTTM's local clock, OSC_CLK, that allows the FTTM to remain in the FREQ_TRUSTED state when the TIME_TRUSTED state cannot be satisfied. The latter is equal to the maximum standard deviation of the measured rate ratio offset of the normalized clock rate of the time arriving on the input interfaces to the normalized clock rate of the FTTM's local clock, OSC_CLK, that allows the FTTM to remain in the FREQ_TRUSTED state when the TIME_TRUSTED state cannot be satisfied.

In this example, the fttm-sel-time-rate-ratio-offset value is 0.0001 (i.e., 100ppm) and the fttm-sel-time-std-dev-rate-ratio-offset value is 0.00002. Per 14.23.17 and 14.23.18, both values are multiplied by 2^{41} before into the managed object.

This example assumes that the fttm-use-osc-clk object is TRUE, enabling OSC_CLK to be used by the FTTM.

```
<fttm-sel-time-rate-ratio-offset>219902326<fttm-sel-time-rate-ratio-offset>
<fttm-sel-time-std-dev-rate-ratio-offset>43980465<fttm-sel-time-std-dev-rate-ratio-
offset>
```

K.11 FTTM status and statistics

The following XML instance data represents a reading of the status and statistics objects from the FTTM.

The status is given for the scenario in which the PTP Instance with instance index 456 is selected by the FTTM as the source for its trusted time. The selected time-index path is shown in Figure K.4.

The ToD returned by the three PTP Instances (i.e., the timeReceiverTimeCallback result) at the last ClockTargetEventCapure.invoke event sent to the PTP Instances is read from fttm-collected-tod. The validity of fttm-collected-tod is first checked by confirming fttm-invoke-status-avail is TRUE. After the ToD values are read for all PTP Instances, the state of fttm-invoke-status-avail is cleared to FALSE by setting fttm-invoke-status-avail-clr to TRUE and then back to FALSE.

The statistic on fttm-sel-time-index-change-cnt shows that the FTTM's counter value for the number of times it has changed its selected time-index is 2. The number of times the selected time-index has changed since the last time this counter value was read is given by the difference between the current value of the counter and the previous value of the counter (with a counter wrap-around to 0 after 65535).

```
          <fttm-num-active-dtsfs>1</fttm-num-active-dtsfs>
          <fttm-num-active-time-indexes>3</fttm-num-active-time-indexes>
          <fttm-tsf-sel-time-index-list>
            <tsf-instance-number>0</tsf-instance-number>
            <fttm-tsf-sel-time-index>1</fttm-tsf-sel-time-index>
          </fttm-tsf-sel-time-index-list>
          <fttm-tsf-sel-time-index-list>
            <tsf-instance-number>1</tsf-instance-number>
            <fttm-tsf-sel-time-index>2</fttm-tsf-sel-time-index>
          </fttm-tsf-sel-time-index-list>
          <fttm-sel-instance-index>456</fttm-sel-instance-index>
          <fttm-trust-state>TIME-TRUSTED</fttm-trust-state>

          <fttm-invoke-status-avail>TRUE</fttm-invoke-status-avail>
          <fttm-collected-tod-list>
            <fttm-input-index-number>1</fttm-input-index-number>
            <extended-timestamp-list>
              <seconds>0</seconds>
              <fractional-nanoseconds>1000</fractional-nanoseconds>
            </extended-timestamp-list>
          </fttm-collected-tod-list>
          <fttm-collected-tod-list>
            <fttm-input-index-number>2</fttm-input-index-number>
            <extended-timestamp-list>
              <seconds>0</seconds>
```

```
        <fractional-nanoseconds>1006</fractional-nanoseconds>
      </extended-timestamp-list>
    </fttm-collected-tod-list>
    <fttm-collected-tod-list>
      <fttm-input-index-number>3</fttm-input-index-number>
      <extended-timestamp-list>
        <seconds>0</seconds>
        <fractional-nanoseconds>1008</fractional-nanoseconds>
      </extended-timestamp-list>
    </fttm-collected-tod-list>
    <fttm-invoke-status-avail-clr>TRUE</fttm-invoke-status-avail-clr>
    <fttm-invoke-status-avail-clr>FALSE</fttm-invoke-status-avail-clr>

    <fttm-sel-time-index-change-cnt>2</fttm-sel-time-index-change-cnt>
```

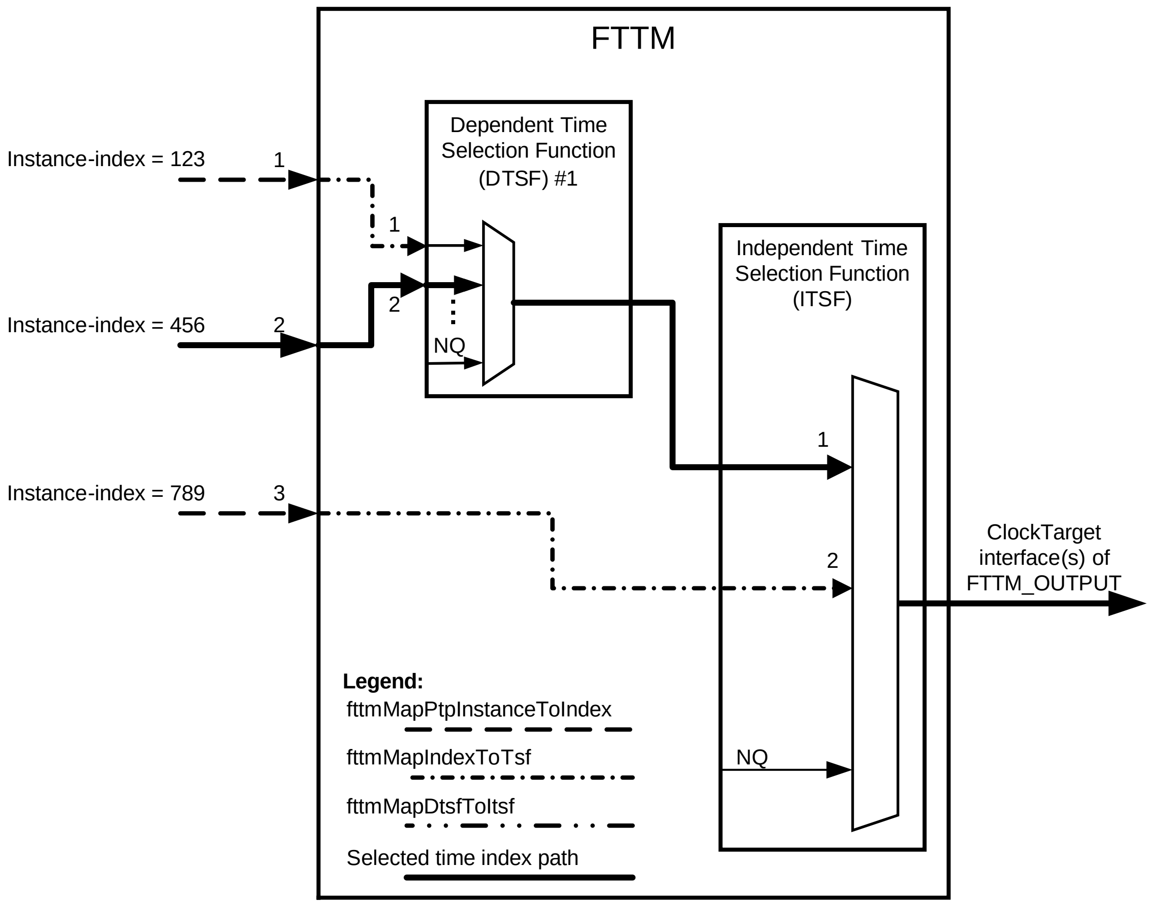


Figure K.4— FTTM status example

K.12 Conclusion

This example concludes with the proper closing tags, encapsulating the detailed configuration of an FTTM system within the common services of the ptp element. This walkthrough serves as a practical guide to understanding the XML configuration necessary for implementing the FTTM service.

```
</fttm-system-ds>
<fttm-system-description-ds>
```

```
          <user-description>This is a test instance.</user-description>
        </fttm-system-description-ds>
      </fttm-system>
    </fault-tolerant-timing-module-service>
  </common-services>
</ptp>
```

Annex L

(informative)

Fault-tolerance claims for FTTM

Annex M

(informative)

Time error accumulation examples

M.1 General

As gPTP time is distributed through a network from a GM to a gPTP Instance, time error accumulates. Some of the reasons for this time error accumulation are listed below:

- — Timing errors at the GM (TEgm)
- — Timing errors at intermediate gPTP Relay Instances (TErly)
- — Timing errors at the gPTP End Instance (TEend)
- — Link asymmetry between gPTP Instances (TElnk)

The above time errors are illustrated in Figure M.1.

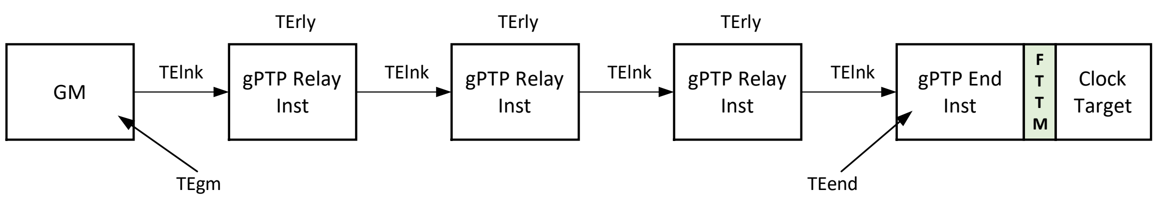


Figure M.1— Time error accumulation across a network

It is possible to determine the potential maximum absolute value of each of the above time errors and, thus, the maximum potential time error at the gPTP End Instance. This result, maxAccumTE (see M.2), if available, can be used by the FTTM in its selection process for a trusted gPTP time to present to the ClockTarget.

NOTE— The potential maximum absolute values of TEgm, TErly, TEend, and TElnk are expected to be calculated or measured prior to operational usage. This could be done at the design, characterization, or certification phase. The specific methods and procedures to do this are not defined in this standard.

The ENHANCED_ACCURACY_METRICS TLV from IEEE Std 1588a-2023 can be used to accumulate the maximum constant and dynamic time errors of each gPTP instance and the connecting links, on a hop-by-hop basis, in the path from the GM to, but not including, the final gPTP End Instance. This TLV is carried in gPTP Announce messages.

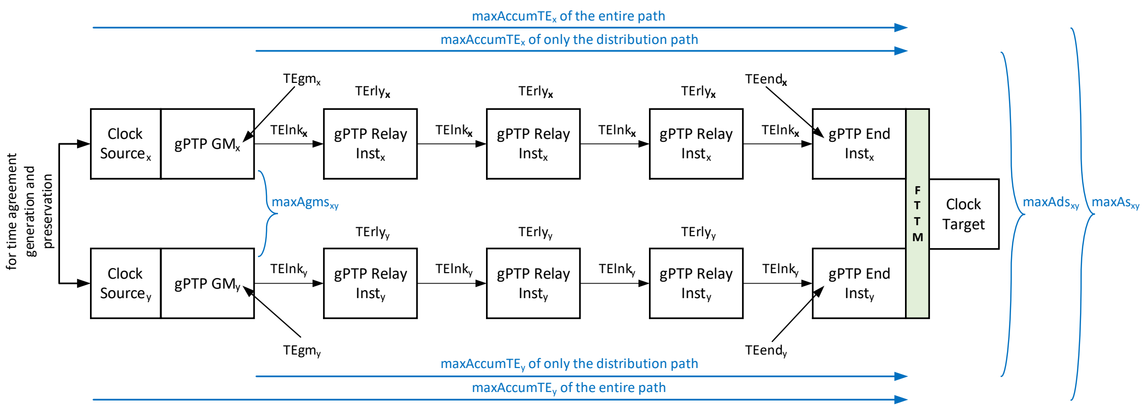


Figure M.2— Time skew across two time distribution paths

Time skew between two time distribution paths is shown in Figure M.2. The time error contributors across each path are the same as shown in Figure M.1, but the contributors on each path are marked with the path's identifying suffix, x or y.

The various time skew results are described in M.2, M.3, M.4, and M.5.

M.2 $maxAccumTE_x$

For the list of reasons for time error accumulation given in M.1, the parameter $maxAccumTE_x$ is the maximum non-faulty accumulated time error magnitude for the time distributed on path x, from its GM (TEgm), through all intermediate gPTP Relay Instances (TErly) and the corresponding links (TElnk), to the gPTP End Instance (TEend) that is connected to the FTTM.

$$maxAccumTE_x = \max(|TEgm_x|) + \sum\max(|TErly_x|) + \sum\max(|TElnk_x|) + \max(|TEend_x|)$$

NOTE—∑ is the summation symbol and represents a summation of all instances of the term adjacent to it.

M.3 $maxAgms_{xy}$

The parameter $maxAgms_{xy}$ is the maximum accepted time skew magnitude between two non-faulty gPTP GMs, GM_x and GM_y. This value is equal to the worst-case time error magnitude between the two GMs when they are not faulty.

$$maxAgms_{xy} = \max(|TEgm_x|) + \max(|TEgm_y|)$$

M.4 $maxAds_{xy}$

The parameter $maxAds_{xy}$ is the maximum accepted distribution skew magnitude between the time of two non-faulty times, distributed on path x and path y. This value is equal to the worst-case time error magnitude between the two times, from the perspective of the FTTM, resulting from their distribution paths when they are not faulty.

$$maxAds_{xy} = \sum\max(|TErly_x|) + \sum\max(|TElnk_x|) + \max(|TEend_x|) + \sum\max(|TErly_y|) + \sum\max(|TElnk_y|) + \max(|TEend_y|)$$

M.5 $maxAs_{xy}$

The parameter $maxAs_{xy}$ is the maximum accepted skew magnitude between two non-faulty times, distributed on path x and path y. This value is equal to the worst-case time error magnitude between two synchronized times, from the perspective of the FTTM, when they are not faulty. This value can be used as a criterion to determine the trustworthiness of the times being compared.

$$maxAs_{xy} = maxAgms_{xy} + maxAds_{xy}$$
$$= maxAccumTE_x + maxAccumTE_y$$

System integrators should design their TSN network, which supports fault-tolerant timing and time integrity, such that synchronization-dependent nodes can withstand a drift equal to the magnitude of this maximum accepted skew.

Annex N

(informative)

Bridging alternate ClockTarget interface types to the FTTM

Show how ClockTargetTriggerGenerate and ClockTargetClockGenerator interfaces can be used with the FTTM, employing proxy timers that can bridge to the ClockTargetEventCapture interface.