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

Insert the following normative reference:

IEEE Std 1588a™-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.

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

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 fttmSelectedIndex (UInteger9)

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.

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 fttmTrustState

The fttmTrustState type is an Enumerated value that holds the trust state of default DTSF or default ITSF processes. The data type is as follows:

- Typedef Enumeration3 fttmTrustState;
  - 000: NO_TRUST
  - 001: TIME_TRUST
  - 010: FREQ_TRUST
  - 011: TRUST_STATE_3
  - 100: TRUST_STATE_4
  - 101: TRUST_STATE_5
  - 110: TRUST_STATE_6
  - 111: TRUST_STATE_7

6.4.3.12 fttmTrustStatus

The fttmTrustStatus type is an Enumerated value that holds a trust status of an item (e.g., the time of a ClockTarget Interface). The data type is as follows:

- Typedef Enumeration1 fttmTrustStatus;
  - 0: TRUSTED
  - 1: NOT_TRUSTED

6.4.3.13 fttmUint9NumActiveDtsfs

The fttmUint9NumActiveDtsfs type is a vector of UInteger9 values with NumActiveDtsfs members. The data type is as follows:

- Typedef UInteger9 [fttmNumActiveDtsfs];

6.4.3.14 fttmUint8NumActiveDtsfs

The fttmUint8NumActiveDtsfs type is a vector of UInteger8 values with NumActiveDtsfs members. The data type is as follows:

— Typedef UInteger8 [fttmNumActiveDtsfs];

6.4.3.15 fttmUint32Uint8fttmNumActiveTimeIndexes

The fttmUint32Uint8fttmNumActiveTimeIndexes type is a vector of a set of UInteger32 and UInteger8 values with fttmNumActiveTimeIndexes members. The data type is as follows:

— Typedef UInteger32 UInteger8 [fttmNumActiveTimeIndexes];

6.4.3.16 fttmExtsfttmNumActiveTimeIndexes

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

— Typedef ExtendedTimestamp [fttmNumActiveTimeIndexes];

6.4.3.17 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 fttmFl64fttmNumActiveTimeIndexes

The fttmFl64fttmNumActiveTimeIndexes type is a vector of Float64 values with fttmNumActiveTimeIndexes members. The data type is as follows:

— Typedef Float64 [fttmNumActiveTimeIndexes];

7. Time-synchronization model for a packet network

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 Description Parameter Data Set (fttmSystemDescriptionDS)

14.23.1 General

The fttmSystemDescriptionDS contains descriptive information for 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 1 to 127. The default value is implementation-specific.

14.23.3 dtsfSelTimeIndex (fttmUint9NumActiveDtsfs)

The dtsfSelTimeIndex object is a vector that gives the input time index that is selected to be the output of each of the active DTSF instances in the FTTM. If the DTSF is using the default MVTISA, its dtsfSelTimeIndex is equal to medTimeIndex (see 19.3.3.3.2.6).

The dtsfSelTimeIndex object is read-only and has a default value of 511 (i.e., the NQ index) for each of its vector members.

14.23.4 fttmClrInvokeStatusAvail (Boolean)

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

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

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

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

14.23.6 fttmDtsfToItsfMap (fttmUint8NumActiveDtsfs)

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

Each fttmDtsfToItsfMap[x] object, where x is a value from 1 to fttmNumActiveDtsfs (see 14.23.16), 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 14.23.17).
  - A value of 0 means the DTSF's output ClockTarget Interface is not connected.

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

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 fttmDtsfToItsfMap object would be as follows:

- fttmDtsfToItsfMap[1] = 3 // DTSF #1's output to ITSF input index #
- fttmDtsfToItsfMap[2] = 2 // DTSF #2's output to ITSF input index #22

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 14.23.13) 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.

14.23.8 fttmIndexToDtsfItsfMap (fttmUint8Uint8fttmNumActiveTimeIndexes)

The fttmIndexToDtsfItsfMap object provides the mapping for all of the FTTM's input ClockTarget Interface index numbers to a DTSF instance and its input ClockTarget Interface index number or to the ITSF instance and its input ClockTarget Interface index number.

Each fttmIndexToDtsfItsfMap[x] object, where x is a value from 1 to fttmNumActiveTimeIndexes (see 14.23.17 and is equal to the assigned index number of the FTTM input ClockTarget Interface (see 14.23.18), consists of two items:

- — The DTSF or ITSF 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 14.23.14).
- — 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 14.23.17).
  - — A value of 0 means that the associated DTSF's/ITSF's input ClockTarget Interface is not active.

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

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 fttmIndexToDtsfItsfMap object would be as follows:

- fttmIndexToDtsfItsfMap[1] = {1,1} //FTTM input index #1 to DTSF instance #1's input index #1
- fttmIndexToDtsfItsfMap[2] = {1,2} //FTTM input index #2 to DTSF instance #1's input index #2
- fttmIndexToDtsfItsfMap[3] = {2,1} //FTTM input index #3 to DTSF instance #2's input index #1
- fttmIndexToDtsfItsfMap[4] = {2,2} //FTTM input index #4 to DTSF instance #2's input index #2
- fttmIndexToDtsfItsfMap[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 fttmDtsfToItsfMap (see 14.23.6).

14.23.9 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 cleared to FALSE by assertion of fttmClrInvokeStatusAvail to TRUE.

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

14.23.10 fttmItsfSelTimeIndex (fttmSelectedIndex)

The fttmItsfSelTimeIndex object gives the input time index that is selected by the ITSF. If the ITSF is using the default MVTISA, fttmItsfSelTimeIndex is equal to medTimeIndex (see 19.3.3.4.2.7).

The fttmItsfSelTimeIndex object is read-only and has a default value of 511 (i.e., the NQ index).

14.23.11 fttmItsfSelTimeIndexChangeCnt (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.

14.23.12 fttmItsfTrustState (fttmTrustState)

The fttmItsfTrustState object gives the current trust status of the ITSF. If the default ITSF is used, then fttmItsfTrustState is equal to prevTrustState (see 19.3.3.4.2.10).

The fttmItsfTrustState object is read-only and has a default value of NO_TRUST.

14.23.13 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 19.2.7.4.

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

14.23.14 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 127.

14.23.15 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.

14.23.16 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.

14.23.17 fttmNumActiveTimeIndexes (UInteger8)

The fttmNumActiveTimeIndexes object gives the number of input ClockTarget Interfaces currently active on the FTTM.

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

14.23.18 fttmPtpInstanceToIndexMap (fttmUint32Uint8fttmNumActiveTimeIndexes)

The fttmPtpInstanceToIndexMap object provides the mapping from the Instance Number of each of the PTP Instances connected to the FTTM to an index number.

The fttmPtpInstanceToIndexMap object is a vector of size fttmNumActiveTimeIndexes, in which each vector member contains the following two items:

— The instance-number of the PTP Instance (see clauses 2 and 3 of IETF RFC8575) that is associated to the input ClockTarget interface of the FTTM.
— The assigned index number for the input ClockTarget interface of the FTTM.

14.23.19 fttmTimeInterfaceRateRatioOff (fttmFl64fttmNumActiveTimeIndexes)

The fttmTimeInterfaceRateRatioOff object gives, for each and every index number of active input ClockTarget Interfaces of the FTTM, the maximum rate-ratio offset magnitude of its clock frequency (relative to the frequency of the FTTM's local clock, OSC_CLK) that is deemed to be acceptable to go to or remain in the FREQ_TRUST state (see clause 19.3.3.4).

For example, a rateRatio offset magnitude of 1ppm is represented by the value 0.000001.

The fttmTimeInterfaceRateRatioOff object is read/write and has a default rate-ratio value of 0.0 for all object members. A

14.23.20 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_TRUST state in the default ITSF state machine, per 19.3.3.4).

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.

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, the ieee802-dot1as-hs.yang and ieee802dot1as-fttm.yang modules import~~s~~ 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, 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 FaultTolerantTimingModule.

Replace Figure 17-1 with the following:

Update Figure 17-1 by adding 802.1ASed objects to the YANG tree overview, just like 802.1ASdm did for its objects.

Replace Figure 17-4 with the following:

Update Figure 17-4 by adding 802.1ASed objects to the common services, just like 802.1ASdm did for its objects.

Insert the following new subclause:

17.2.1 FTTM YANG module UML diagram

The UML diagram in Figure 1 illustrates the interconnections and relationships among the components of the YANG module. The ieee802-dot1as-fttm YANG module is intricately designed to facilitate the management of Fault-Tolerant Timing Module (FTTM) systems, building upon the foundational elements specified in IEEE Std 1588. This module is characterized by its unique namespace and prefix, and it incorporates elements from both the ieee1588-ptp-tt, ieee802-dot1as-ptp, and ieee802-dot1as-hs modules through import statements, indicating a layered approach to extending these base models to suit the specific requirements of IEEE Std 802.1AS.

Central to the FTTM YANG module's structure are various groupings that organize related management objects into coherent clusters. These groupings address different facets of FTTM management, ranging from the configuration of individual FTTM instances to the referencing of PTP instances, and the management of timing accuracy and skew. For instance, the fault-tolerant-timing-module-group is pivotal for managing a single FTTM instance and encapsulates other groupings to cover diverse aspects of FTTM configuration such as algorithms, time selection functions, and error management. Groupings in YANG modules offer significant advantages by promoting reusability, modularity, extensibility, and organization within network configuration models. They enable developers to define a common set of configuration elements that can be reused across different parts of a YANG model or even across multiple models, reducing duplication and ensuring consistency. This modularity simplifies the model, making it easier to understand, maintain, and extend. Groupings also allow for the abstraction of complex configurations into simpler constructs, improving the model's usability and readability. Furthermore, they provide flexibility in usage through the "uses" statement, allowing tailored inclusion of configuration blocks as needed. Overall, groupings enhance the efficiency, maintainability, and scalability of YANG models, especially in complex network environments and as technology standards evolve.

Moreover, the module extends the ieee1588-ptp-tt module's structures through augmentations, seamlessly integrating FTTM functionality into the broader PTP management framework. This includes enhancements to instance lists, synchronization status descriptions, and port statistics, among others, thereby providing a holistic and extensible framework for the management of FTTM systems. Through these augmentations and groupings, the ieee802-dot1as-fttm YANG module establishes a comprehensive model for the configuration and management of FTTM systems, leveraging existing standards to ensure robust and precise time-sensitive networking.

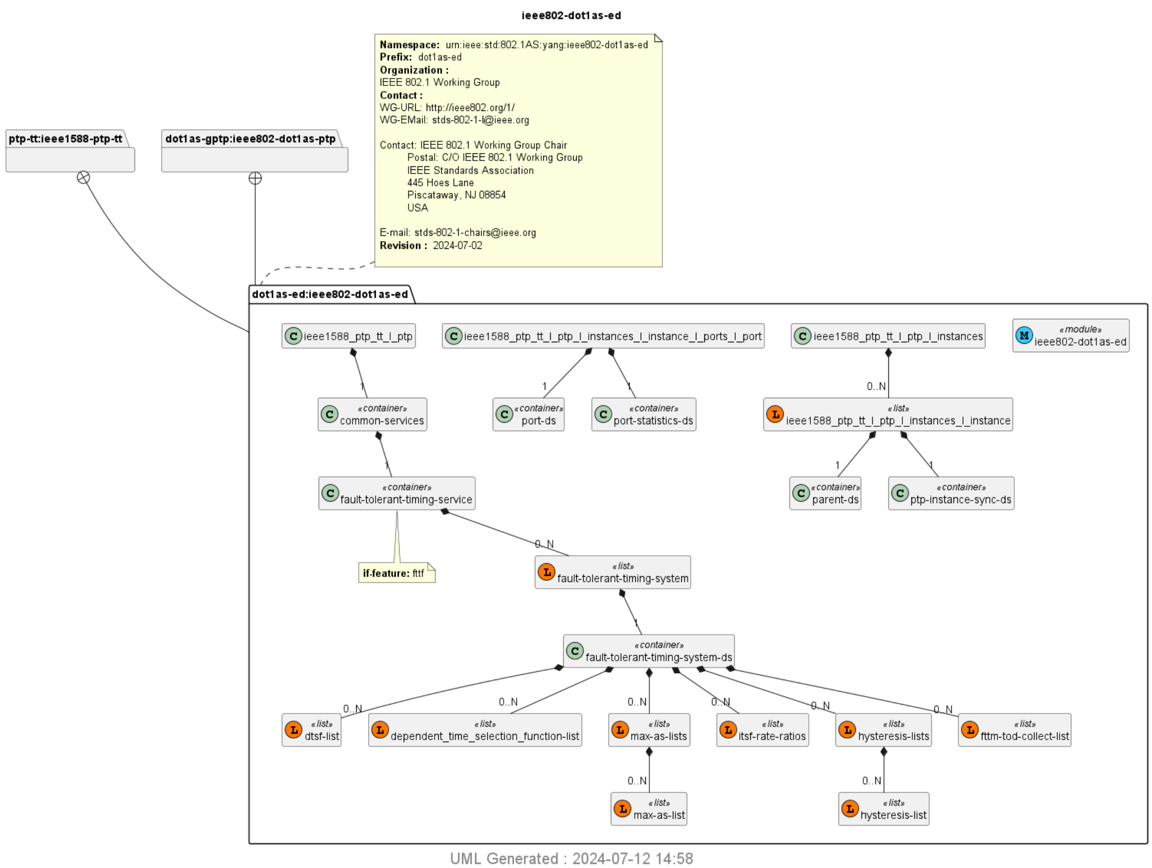


Figure 1—FTTM YANG module UML diagram

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-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

Insert the following subclause:

17.5.3 Tree diagram for ieee802-dot1as-fttm.yang

```
module: ieee802-dot1as-fttm

  augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:parent-ds:
    +--ro gm-present?   boolean
  augment /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance:
    +--rw ptp-instance-sync-ds
       +--ro is-synced?   boolean
  augment      /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port/ptp-
tt:port-ds:
    +--rw gptp-capable-state-machines-enabled?   boolean
  augment    /ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance/ptp-tt:ports/ptp-tt:port/dot1as-
gptp:port-statistics-ds:
    +--ro rx-sync-count-time-receiver-p?   uint32
```

```
augment /ptp-tt:ptp/ptp-tt:common-services:
  +--rw fault-tolerant-timing-module-service {fttm}?
     +--rw fault-tolerant-timing-module* [fault-tolerant-timing-module-index]
        +--rw fault-tolerant-timing-module-index           uint8
        +--rw fault-tolerant-timing-module-ds
           +--ro dtsf-max-num-time-indexes?                uint8
           +--ro fttm-max-num-dtsfs?                       uint8
           +--ro fttm-num-active-dtsfs?                    uint8
           +--ro fttm-max-num-time-indexes?                uint8
           +--ro fttm-num-active-time-indexes?             uint8
           +--ro fttm-use-osc-clk?                         boolean
           +--ro algorithm-used?                           string
           +--rw fttm-ptp-instance-to-index-map-list* [instance-number]
           |  +--rw instance-number                        uint32
           |  +--rw fttm-input-index-number?               unit8
           +--rw fttm-index-to-dtsf-itsf-map-list* [fttm-input-index-number]
           |  +--rw fttm-input-index-number                uint8
           |  +--rw itsf-instance-number?                  uint8
           |  +--rw dtsf-instance-number?                  uint8
           |  +--rw itsf-input-index-number?               uint8
           |  +--rw dtsf-input-index-number?               uint8
           +--rw fttm-dtsf-to-itsf-map-list* [dtsf-instance-number]
           |  +--rw dtsf-instance-number                   uint8
           |  +--rw itsf-input-index-number?               uint8
           +--rw dtsf-instance-number-list* [dtsf-instance-number]
           |  +--rw dtsf-instance-number                   uint8
           |  +--ro dtsf-algorithm?                        string
           +--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
           +--rw fttm-hyst-lists* [fttm-input-index-number]
           |  +--rw fttm-input-index-number                uint8
           |  +--rw hysteresis-list* [fttm-input-index-number]
           |     +--rw fttm-input-index-number             uint8
           |     +--rw fttm-hyst?                          uint32
           +--rw fttm-time-interface-rate-ratio-lists* [fttm-input-index-number]
           |  +--rw fttm-input-index-number                uint8
           |  +--rw fttm-time-interface-rate-ratio?        decimal64
           +--rw dtsf-sel-time-index-list* [dtsf-instance-number]
           |  +--rw dtsf-instance-number                   uint8
           |  +--ro dtsf-sel-time-index?                   fttmSelectedIndex
           +--ro fttm-itsf-sel-time-index?                 fttmSelectedIndex
           +--ro fttm-itsf-sel-time-index-change-cnt?      uint16
           +--ro fttm-itsf-trust-state?                    fttmTrustState
           +--rw fttm-collected-tod-list* [fttm-input-index-number]
           |  +--rw fttm-input-index-number                uint8
           |  +--ro fttm-collected-tod?                    extendedTimeStamp
           +--ro fttm-invoke-status-avail?                 boolean
           +--rw fttm-clr-invoke-status-avail?             boolean
```

17.6 YANG modules

17.6.1 Module ieee802-dot1as-gptp-yang

17.6.2 Module ieee802-dot1as-hs-yang

Insert the following subclause:

17.6.3 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;
  }
  import ieee802-dot1as-ptp {
    prefix dot1as-gptp;
  }

  organization
    "IEEE 802.1 Working Group";
  contact
    "WG-URL: http://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
    "Management objects that control the FTTM as
     specified in IEEE Std 802.1ASed-2024.

     References in this YANG module to IEEE Std 802.1AS are to
     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, and
     IEEE Std 802.1ASdm.

     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 2024-07-15 {
    description
      "Published as part of IEEE Std 802.1ASed-2024.
       Initial version.";
    reference
      "IEEE Std 802.1AS - YANG Data Model";
  }


// typedef statements
  typedef fttm-selected-index
    type uint9;
    uses fttm-ptp-instance-to-index-map-group;
    description
      "The fttmSelectedIndex type gives the selected ClockTarget interface index
       for a DTSF or for the ITSF, with the following meanings:
         00 to 255:  The index number of the selected input ClockTarget interface.
         256 to 510: Reserved.
         511:        The NQ index.";
    reference
      "6.4.3.9 of IEEE Std 802.1ASed";

  typedef fttm-interface-value-array
    description
      "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;”;
    reference
      "6.4.3.10 of IEEE Std 802.1ASed";

  typedef fttm-trust-state
    type enumeration3;
    description
      “The fttmTrustState type is an Enumerated value that holds the trust state of
       default DTSF or default ITSF processes. The data type is as follows:
         Typedef Enumeration3 fttmTrustState;
           000:  NO_TRUST
           001:  TIME_TRUST
           010:  FREQ_TRUST
           011:  TRUST_STATE_3
           100:  TRUST_STATE_4
           101:  TRUST_STATE_5
           110:  TRUST_STATE_6
           111:  TRUST_STATE_7”;
    reference
      "6.4.3.11 of IEEE Std 802.1ASed";

  typedef fttm-trust-status
    type enumeration1;
    description
      “The fttmTrustStatus type is an Enumerated value that holds a trust status of
       an item (e.g., the time of a ClockTarget Interface). The data type is as follows:
         Typedef Enumeration1 fttmTrustStatus;
           0:  TRUSTED
           1:  NOT_TRUSTED”;
    reference
      "6.4.3.12 of IEEE Std 802.1ASed";

  typedef fttm-uint9-num-active-dtsfs
    description
      “The fttmUint9NumActiveDtsfs type is a vector of UInteger9 values
       with NumActiveDtsfs members. The data type is as follows:
         Typedef UInteger9 [fttmNumActiveDtsfs];”;
    reference
      "6.4.3.13 of IEEE Std 802.1ASed";

  typedef fttm-uint8-num-active-dtsfs
    description
      “The fttmUint8NumActiveDtsfs type is a vector of UInteger8
       values with NumActiveDtsfs members. The data type is as follows:
         Typedef UInteger8 [fttmNumActiveDtsfs];”;
    reference
      "6.4.3.14 of IEEE Std 802.1ASed";

  typedef fttm-uint32-uint8-fttm-num-active-time-indexes
    description
      “The fttmUint32Uint8fttmNumActiveTimeIndexes type is a vector
       of a set of UInteger32 and UInteger8 values with
       fttmNumActiveTimeIndexes members. The data type is as follows:
         Typedef UInteger32 UInteger8 [fttmNumActiveTimeIndexes];”;
    reference
      "6.4.3.15 of IEEE Std 802.1ASed";

  typedef fttm-exts-fttm-num-active-time-indexes
    description
      “The fttmExtsfttmNumActiveTimeIndexes type is a vector of
       ExtendedTimestamp values with fttmNumActiveTimeIndexes members.
       The data type is as follows:
         Typedef ExtendedTimestamp [fttmNumActiveTimeIndexes];”;
    reference
      "6.4.3.16 of IEEE Std 802.1ASed";

  typedef fttm-uint8-uint8-fttm-num-active-time-indexes
    description
      “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];”;
      reference
        "6.4.3.17 of IEEE Std 802.1ASed";

    typedef fttm-fl64-fttm-num-active-time-indexes
      description
        “The fttmFl64fttmNumActiveTimeIndexes type is a vector of Float64
         values with fttmNumActiveTimeIndexes members.
         The data type is as follows:
           Typedef Float64 [fttmNumActiveTimeIndexes];”;
      reference
        "6.4.3.18 of IEEE Std 802.1ASed";


  // FTTM
    feature fttm {
      description
        "This feature indicates that the device supports fttm functionality.";
    }

    grouping fault-tolerant-timing-module-group {
      description
        "Management of a single FTTM.";
      reference
        "IEEE Std 802.1ASed 14.23.14";

      leaf dtsf-max-num-time-indexes {
        type uint8;
        config false;
        default “implementation-specific”
        description
          "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 dtsfMaxNumTimeIndexes(UInteger8)";
      }
      leaf fttm-max-num-dtsfs {
        type uint7;
        config false;
        default “implementation-specific”
        description
          "Maximum number of DTSF instances available in the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.14 fttmMaxNumDtfs (UInteger8)";
      }
      leaf fttm-num-active-dtsfs {
        type uint8;
        config false;
        uses fttm-index-to-dtsf-itsf-map-group;
        description
          "Number of active DTSF instances currently used in the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.16 fttmMaxNumDtfs (UInteger8)";
      }
      leaf fttm-max-num-time-indexes {
        type uint8;
        config false;
        default “implementation-specific”
        description
          "Maximum number of input ClockTarget Interfaces available on the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.15 fttmMaxNumTimeIndexes (UInteger8)";
      }
      leaf fttm-num-active-time-indexes {
        type uint8;
        config false;
        uses fttm-ptp-instance-to-index-map-group;
        description
          "The number of input ClockTarget Interfaces currently active on the FTTM.";
        reference
          "IEEE Std 802.1ASed 14.23.17 fttmMaxNumTimeIndexes (UInteger8)";
```

```
        }
        leaf fttm-use-osc-clk {
          type boolean;
          config false;
          default "implementation-specific"
          description
            "Defines whether the OSC_CLK frequency is used as a reference
             for time integrity";
          reference
            "IEEE Std 802.1ASed 14.23.20 fttmUseOscClk (Boolean)";
        }
        leaf fttm-itsf-sel-time-index {
          type fttm-selected-index;
          config false;
          default "511"
          uses fttm-index-to-dtsf-itsf-map-group;
          description
            "Gives the input time index that is selected by the ITSF.";
          reference
            "IEEE 802.1ASed 14.23.10 fttmItsfSelTimeIndex (fttmSelectedIndex)";
        }
        leaf fttm-itsf-trust-state {
          type fttm-trust-state;
          config false;
          default "NO-TRUST"
          description
            "The fttmItsfTrustState object gives the current trust status of the ITSF.";
          reference
            "IEEE Std 802.1ASed 14.23.12 fttmItsfTrustState (fttmTrustState)";
        }
        leaf fttm-itsf-sel-time-index-change-cnt {
          type uint16;
          config false;
          default "0"
          description
            "Gives the number of times the ITSF has changed its time index selection.
             The count rolls over to 0 if the count is incremented when its current
             value is 65535";
          reference
            "IEEE 802.1ASed 14.23.11 fttmItsfSelTimeIndexChangeCnt (UInteger16)";
        }
        leaf fttm-invoke-status-avail {
          type boolean;
          config false;
          default "false"
          description
            "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 cleared to FALSE by assertion of
             fttmClrInvokeStatusAvail to TRUE.";
          reference
            "IEEE 802.1ASed 14.23.9 fttmInvokeStatusAvail (Boolean)";
        }
        leaf fttm-clr-invoke-status-avail {
          type boolean;
          default "false"
          description
            "Used to clear the fttmInvokeStatusAvail object to FALSE.
             When fttmClrInvokeStatusAvail is changed from FALSE
             to TRUE, fttmInvokeStatusAvail is cleared to FALSE.
             The fttmClrInvokeStatusAvail object must be written to FALSE before it can be
             used again.
             The fttmClrInvokeStatusAvail object is read/write, with a default value
             of FALSE.";
          reference
            "IEEE 802.1ASed 14.23.4 fttmClrInvokeStatusAvail (Boolean)";
```

```
    }

    uses fttm-ptp-instance-to-index-map-group;
    uses max-as-group;
    uses fttm-hyst-group;
    uses fttm-time-interface-rate-ratio-off-group;
    uses fttm-index-to-dtsf-itsf-map-group;
    uses fttm-dtsf-to-itsf-map-group;
    uses dtsf-sel-time-index-group;
    uses fttm-collected-tod-group;
  }

  grouping instance-number-group {
    description
      "See IEEE Std 802.1ASed Figure 4—FTTM functional block diagram 'Clock target from
       PTP instances' indicating that we will have access to the ptp-tt:ptp/ptp-instances
       list on the device. ptp-tt:/common-services provides management access to the
       common services where common services operate on all PTP Instances of the PTP
       Node. Thus, we map an entire ptp-instance when working with FTTM indices and all
       its related information comes along for free where the index is ptp-
       tt:ptp/instances/instance/instance-index. ptp-tt:ptp/instances/default-ds/domain-
       number is already available.";
    reference
      "Clause 19.3.3.2.1.8 of IEEE Std 802.1ASed";
    leaf instance-number {
      type uint32;
      default "3";
      description
        "An index number that identifies a ptp-instance input to the FTTM.";
      reference
        "Clause 19.3.3.2.1.8 ordered_time_index of IEEE Std 802.1ASed";
    }
  }

  grouping fttm-time-interface-rate-ratio-off-group {
    list fttm-time-interface-rate-ratio-off-list {
      key "fttm-input-index-number";
      uses fttm-ptp-instance-to-index-map-group;
      leaf fttm-time-interface-rate-ratio-off {
        type decimal64 {
          fraction-digits 18;
        }
        default "0.0";
        config true;
        description
          "The fttmTimeInterfaceRateRatioOff object gives, for each and every index
           number of active input ClockTarget Interfaces of the FTTM, the maximum rate-
           ratio offset of its clock frequency (relative to the frequency of the FTTM’s
           local clock, OSC_CLK) that is deemed to be acceptable to go to or remain in
           the FREQ_TRUST state (see clause 19.3.3.4).";
        reference
          "IEEE Std 802.1ASed 14.23.19 fttmTimeInterfaceRateRatioOff
          (fttmFl64fttmNumActiveTimeIndexes)";
      }
    }
  }

  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 19.2.7.4.";
    reference
      "IEEE Std 802.1ASed 14.23.13 fttmMaxAs (fttmInterfaceValueArray)";
    list fttm-max-as-lists {
      key "fttm-input-index-number";
      uses fttm-ptp-instance-to-index-map-group;
      description
        "The index x.";
      reference
```

```
          "IEEE Std 802.1ASed 14.23.13 fttmMaxAs (fttmInterfaceValueArray)";
        list fttm-max-as-list {
          key "fttm-input-index-number";
          uses fttm-ptp-instance-to-index-map-group;
          description
            "The index y.";
          reference
            "IEEE Std 802.1ASed 14.23.13 fttmMaxAs (fttmInterfaceValueArray)";
          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 14.23.13 fttmMaxAs (fttmInterfaceValueArray)";
          }
        }
      }
    }

    grouping fttm-hyst-group {
      description
        "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 object fttmHyst
         is a read/write object with a default value of 0 for all array members.";
      reference
        "IEEE Std 802.1ASed 14.23.7 fttm-hyst (fttmInterfaceValueArray)";
      list fttm-hyst-lists {
        key "fttm-input-index-number";
        uses fttm-ptp-instance-to-index-map-group;
        description
          "The x index.";
        reference
          "IEEE Std 802.1ASed 14.23.7 fttm-hyst (fttmInterfaceValueArray)";
        list fttm-hyst-list {
          key "fttm-input-index-number ";
          uses fttm-ptp-instance-to-index-map-group;
          description
            "The y index.";
          reference
            "IEEE Std 802.1ASed 14.23.7 fttm-hyst";
          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 fttmHyst";
          }
        }
      }
    }

    grouping fttm-ptp-instance-to-index-map-group {
      description
        "The fttmPtpInstanceToIndexMap object provides the mapping from the Instance Number
         of each of the PTP Instances connected to the FTTM to an index number ";
      reference
        "IEEE Std 802.1ASed 14.23.18 fttmPtpInstanceToIndexMap
         (fttmUint32Uint8fttmNumActiveTimeIndexes)";
      list fttm-ptp-instance-to-index-map-lists {
        key "instance-number";
        description
          "This grouping allows associations of the instance numbers of PTP Instances
           to FTTM input index numbers.";
```

```
        reference
          "IEEE Std 802.1ASed 14.23.18 fttmPTPInstancetoIndexMap
                                    (fttmUint32Uint8fttmNumActiveTimeIndexes)";
        uses instance-number-group;
        leaf fttm-input-index-number {
          type uint8;
          description
            "The FTTM index number for the FTTM input ClockTarget Interface associated
             with the PTP Instance with the corresponding instance number.";
        reference
          "IEEE Std 802.1ASed 14.23.18 fttmPTPInstancetoIndexMap
                                    (fttmUint32Uint8fttmNumActiveTimeIndexes)";
        }
      }
    }

    grouping fttm-index-to-dtsf-itsf-map-group {
      description
        "The fttmIndexToDtsfItsfMap object provides the mapping for all of the FTTM's input
         ClockTarget Interface index numbers to a DTSF instance and its input ClockTarget
         Interface index number or to the ITSF instance and its input ClockTarget Interface
         index number.
         Each fttmIndexToDtsfItsfMap[x] object, where x is a value from 1 to
         fttmNumActiveTimeIndexes (see 14.23.17 and is equal to the assigned index number
         of the FTTM input ClockTarget Interface (see 14.23.18), consists of two items:
           -The DTSF or ITSF 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 14.23.14).
           -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 14.23.17).
              -A value of 0 means that the associated DTSF's/ITSF's input ClockTarget
               Interface is not active.";
      reference
        "IEEE Std 802.1ASed 14.23.8 fttmIndexToDtsfItsfMap
         (fttmUint8Uint8fttmNumActiveTimeIndexes)";
      list fttm-index-to-dtsf-itsf-map-lists {
        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 14.23.8 fttmIndexToDtsfItsfMap
                                   (fttmUint8Uint8fttmNumActiveTimeIndexes)";
        uses fttm-ptp-instance-to-index-map-group;
        leaf dtsf-instance-number {
          type uint8;
          description
            "This grouping allows associations of the FTTM input index numbers to
             DTSF input index numbers and ITSF input index numbers. A value of 0
             Represents the ITSF. Other values represent the DTSF instance number.";
        reference
          "IEEE Std 802.1ASed 14.23.8 fttmIndexToDtsfItsfMap
                                    (fttmUint8Uint8fttmNumActiveTimeIndexes)";
        leaf fttm-input-index-number {
          type uint8;
          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 14.23.8 fttmIndexToDtsfItsfMap
                                    (fttmUint8Uint8fttmNumActiveTimeIndexes)";
        }
      }
    }

    grouping fttm-dtsf-to-itsf-map-group {
```

```
      description
        " The fttmDtsfToItsfMap object provides the mapping for all the FTTM’s DTSF output
          ClockTarget Interfaces to the ITSF and its input indexes.
          Each fttmDtsfToItsfMap[x] object, where x is a value from 1 to fttmNumActiveDtsfs
          (see 14.23.16), 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 14.23.17).
            -A value of 0 means the DTSF’s output ClockTarget Interface is not connected.";
      reference
        "IEEE Std 802.1ASed 14.23.6 fttmDtsfToItsfMap (fttmUint8NumActiveDtsfs)";
      list fttm-dtsf-to-itsf-map-lists {
        key "dtsf-instance-number";
        uses fttm-index-to-dtsf-itsf-map-group;
        description
          "This grouping allows associations of all the DTSF output ClockTarget interfaces
           to ITSF input index numbers.";
        reference
          "IEEE Std 802.1ASed 14.23.6 fttmDtsfToItsfMap (fttmUint8NumActiveDtsfs)";
        leaf itsf-input-index-number {
          type uint8;
          description
            "This grouping allows associations of all the DTSF output ClockTarget
             Interfaces to ITSF input index numbers.";
        reference
          " IEEE Std 802.1ASed 14.23.6 fttmDtsfToItsfMap (fttmUint8NumActiveDtsfs)";";
        }
      }
    }

  grouping dtsf-sel-time-index-group {
      list dtsf-sel-time-index-list {
        key "dtsf-instance-number";
        uses fttm-index-to-dtsf-itsf-map-group;
        leaf dtsf-sel-time-index {
          type uint9;
          config false;
          default “511”
          description
            "Gives the input time index that is selected to be the output of each of the
             active DTSF instances in the FTTM.";
          reference
            "IEEE 802.1ASed 14.23.3 dtsfSelTImeIndex";
        }
      }
    }

  grouping fttm-collected-tod-group {
      list fttm-collected-tod-list {
        key "fttm-input-index-number";
        uses fttm-ptp-instance-to-index-map-group;
        leaf fttm-collected-tod {
          type extended-timestamp;
          config false;
          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 14.23.5 fttmCollectedToD
                                    (fttmExtsfttmNumActiveTimeIndexes)";
        }
      }
    }


    /* From dot1as-hs */

    augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
```

```
        + "/ptp-tt:parent-ds" {
    description
      "Augment IEEE Std 1588 parentDS.";
    leaf gm-present {
      type boolean;
      config false;
      description
        "The value of gmPresent is set equal to the value of the
         global variable gmPresent. This parameter indicates to the
         ClockTarget whether a Grandmaster PTP Instance is
         present.";
      reference
        "14.4.8 of IEEE Std 802.1AS";
    }
  }

  /* From dot1as-hs */

  augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance" {
    description
      "Augment IEEE Std 1588 instanceList.";
    container ptp-instance-sync-ds {
      description
        "The ptpInstanceSyncDS describes the synchronization status
         of the PTP Instance.";
      reference
        "14.8 of IEEE Std 802.1AS";
      leaf is-synced {
        type boolean;
        config false;
        description
          "The value of the global variable isSynced.";
        reference
          "14.8.2 of IEEE Std 802.1AS";
      }
    }
  }

  /* From dot1as-hs */

  augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
        + "/ptp-tt:ports/ptp-tt:port/ptp-tt:port-ds" {
    description
      "Augment IEEE Std 1588 commonServices.";
    leaf gptp-capable-state-machines-enabled {
      type boolean;
      description
        "A Boolean that is used to enable or disable the
         GptpCapableTransmit, GptpCapableReceive, and
         GptpCapableIntervalSetting state machines.";
      reference
        "14.8.55 of IEEE Std 802.1AS";
    }
  }

  /* From dot1as-hs */

  augment "/ptp-tt:ptp/ptp-tt:instances/ptp-tt:instance"
        + "/ptp-tt:ports/ptp-tt:port"
        + "/dot1as-gptp:port-statistics-ds" {
    description
      "Augment IEEE Std 802.1AS PortStatisticsDS.";
    leaf rx-sync-count-time-receiver-p {
      type uint32;
      config false;
      description
        "This counter increments whenever time synchronization
         information is received on a PTP Port when its port
         state is TimeReceiverPort.";
      reference
        "14.10.20 of IEEE Std 802.1AS";
```

```
      }
    }

    augment "/ptp-tt:ptp/ptp-tt:common-services" {
      description
        "Augment IEEE Std 1588 commonServices.

         IEEE Std 802.1ASed specifies the fault-tolerant-timing-module.";
      container fault-tolerant-timing-service {
        if-feature "fttm";
        description
          "The fault-tolerant-timing-module-system structure contains the
                  fault-tolerant-timing-module-list, which is a list of instances of the
                  fault-tolerant-timing-module Service.";
        reference
          "Set of all Fault-Tolerant Timing Module systems under management.";
        list fault-tolerant-timing-module-system {
          key "fault-tolerant-timing-module-index";
          description
            "List of instances of the fault-tolerant-timing-modules.";
          leaf fault-tolerant-timing-module-index {
            type uint8;
            default "1";
            description
              "Index for the fault-tolerant-timing-module system.";
          }
          container fault-tolerant-timing-module-ds {
            description
              "The fault-tolerant-timing-system describes the attributes of the respective
  instance of the fault-tolerant-timing-modules.";
            reference
              "14.23 Fault Tolerant Timing Module System Description Parameter Data Set
  (fttmSystemDescriptionDS) of IEEE Std 802.1ASed";
            uses fault-tolerant-timing-module-group;
          }
        }
      }
    }
  }
```

18. Hot Standby

Insert the following clause.

19. Fault-tolerant timing with time integrity

19.1 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 19.2. Details on the FTTM and its default operations are given in 19.3. General information about fault-tolerant timing with time integrity can be found in Annex J.

19.2 Fault-tolerant time synchronization concepts

19.2.1 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.

19.2.2 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 19.2.7.4 for an example of such a criterion.

19.2.3 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].

19.2.4 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.

19.2.5 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 19.2.6), 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 19.3).

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 19.3).

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.

19.2.6 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 19.2.3) 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.

19.2.7 Time error accumulation

As gPTP time is distributed through a network from a GM to a gPTP Instance, time error accumulates due to the following reasons:

- — 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 2.

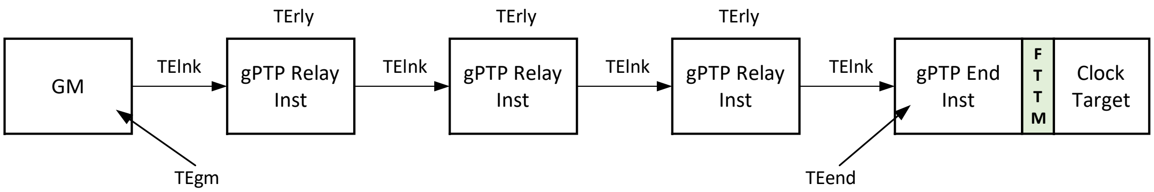


Figure 2—Time error accumulation across a network

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 19.2.7.1), if available, can be used by the FTTM to select the best 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.

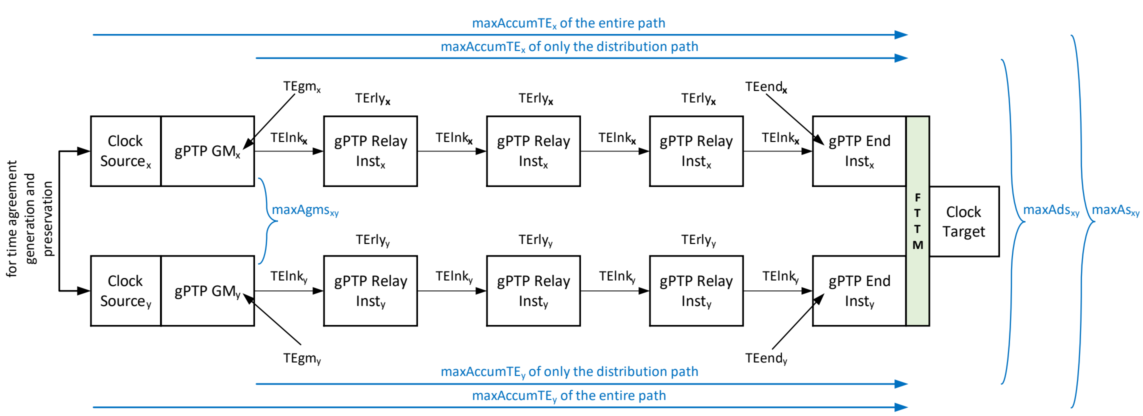


Figure 3—Time skew across two time distribution paths

Time skew between two time distribution paths is shown in Figure 3. The time error contributors across each path are the same as shown in Figure 2, 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 19.2.7.1, 19.2.7.2, 19.2.7.3, and 19.2.7.4.

19.2.7.1 maxAccumTE$_x$

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.

$$\text{maxAccumTE}_x = \max(|\text{TEgm}_x|) + \sum\max(|\text{TErly}_x|) + \sum\max(|\text{TElnk}_x|) + \max(|\text{TEend}_x|)$$

where ∑ is the summation symbol and represents a summation of all instances of the term adjacent to it

19.2.7.2 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.

$$\text{maxAgms}_{xy} = \max(|\text{TEgm}_x|) + \max(|\text{TEgm}_y|)$$

19.2.7.3 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.

$$\begin{aligned}\text{maxAds}_{xy} = {} & \sum\max(|\text{TErly}_x|) + \sum\max(|\text{TElnk}_x|) + \max(|\text{TEend}_x|) + \\ & \sum\max(|\text{TErly}_y|) + \sum\max(|\text{TElnk}_y|) + \max(|\text{TEend}_y|)\end{aligned}$$

19.2.7.4 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.

$$\begin{aligned}\text{maxAs}_{xy} &= \text{maxAgms}_{xy} + \text{maxAds}_{xy} \\ &= \text{maxAccumTE}_x + \text{maxAccumTE}_y\end{aligned}$$

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.

19.3 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 19.2.1) and integrity (see 19.2.2). 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.

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 algorithm.

Because the FTTM resides between gPTP Instances and a ClockTarget, its effect is localized. This allows different algorithms, each of which specifically serves the timing requirements of the corresponding ClockTarget, to be used by each FTTM. The default algorithms for the FTTM are defined in 19.3.3.

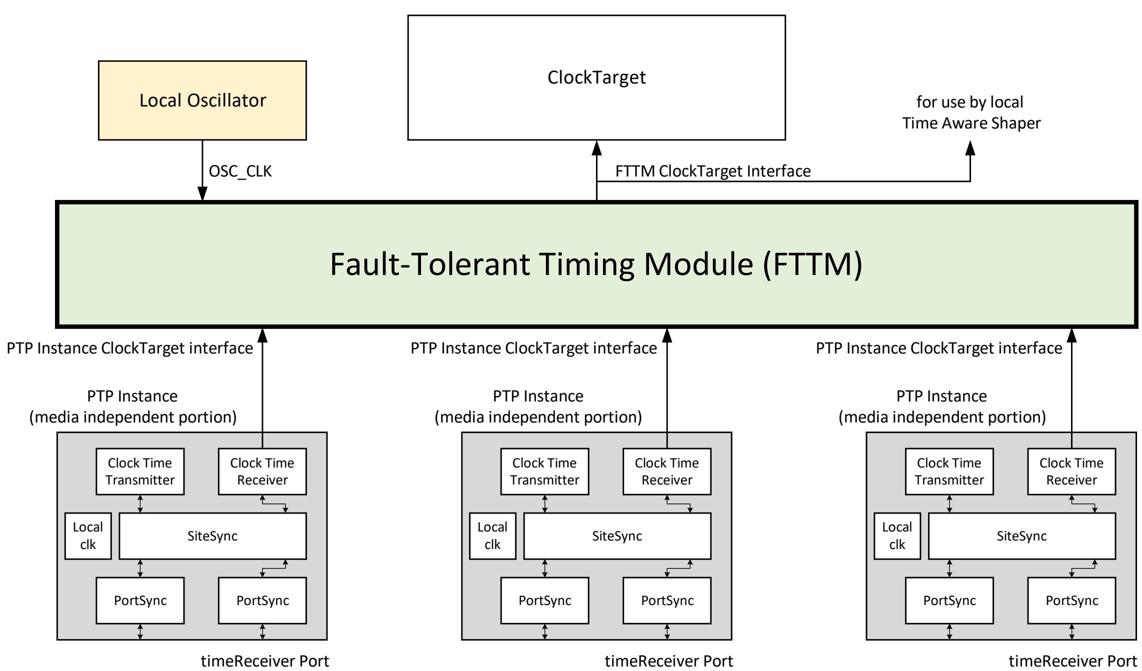


Figure 4— Fault-Tolerant Timing Module in operation

19.3.1 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 19.2.5 and 19.2.6, respectively.

19.3.2 Functional description

The FTTM shall consist 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.
- ClockTarget application interfaces (see Clause 9) providing time information to the FTTM, where gPTP End Instances serve as the ClockTimeReceiver entities and the FTTM serves as the ClockTarget entity. Time is passed to the FTTM via each ClockTarget application interface’s timeReceiverTimeCallback parameter.
- Zero or more instances of a Dependent Time Selection Function (DTSF).
- Zero or more instances of ClockTarget application interface(s) (see clause 9) 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 ClockTarget application interface’s timeReceiverTimeCallback parameter.
- One instance of an Independent Time Selection Function (ITSF).
- ClockTarget application interface(s) (see clause 9) 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.

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

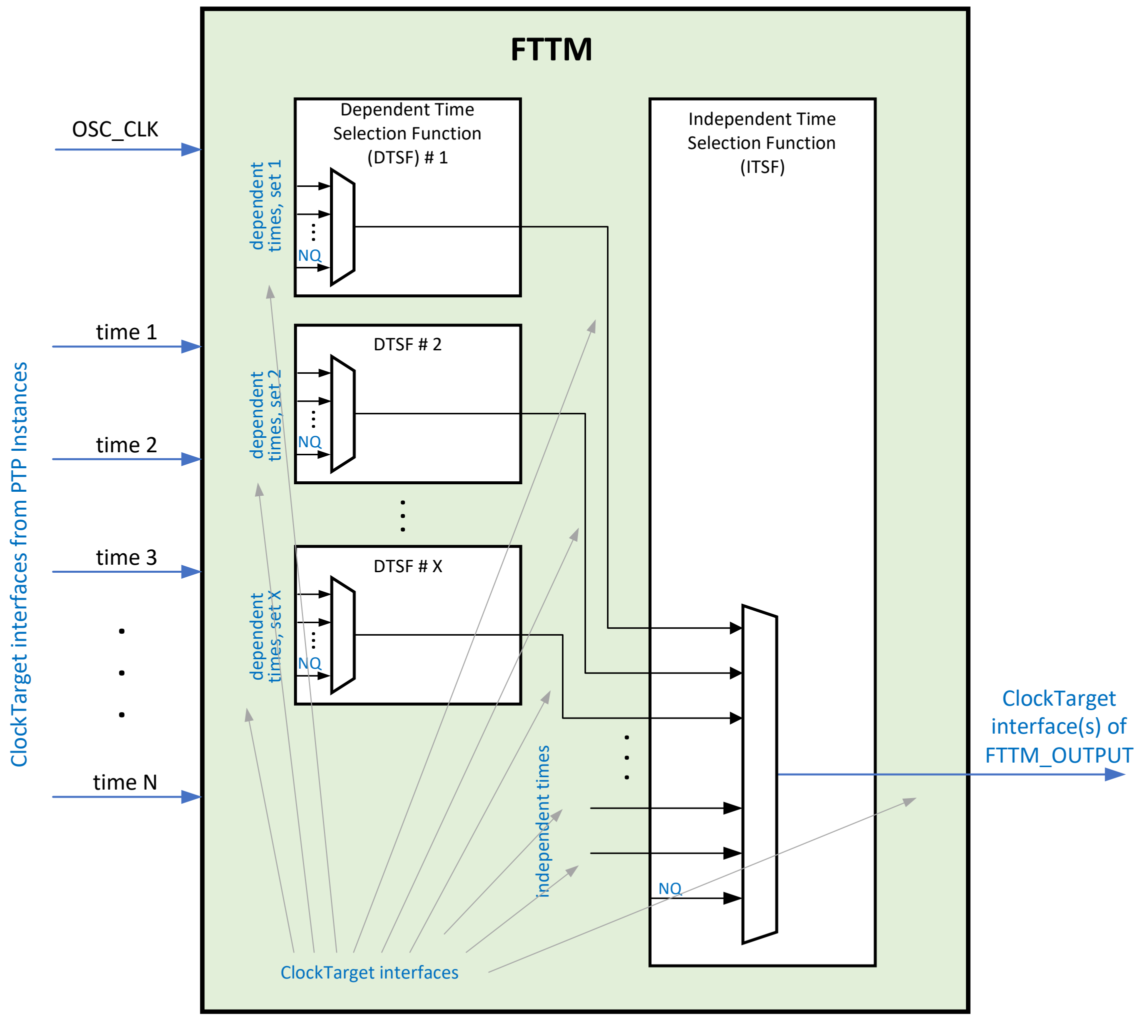


Figure 5—FTTM functional block diagram

19.3.2.1 Input ClockTarget interfaces

The ClockTarget interfaces that pass time information from the gPTP End Instances to the FTTM are designated as, from the FTTM's perspective, input ClockTarget interfaces. They may be any of the following types defined in clause 9, ClockTargetEventCapture, ClockTargetTriggerGenerate, and ClockTargetClockGenerator, or they may be of another type. However, they should all be of the same type. The ClockTargetPhaseDiscontinuity interface should also be provided by the gPTP End Instances to the FTTM.

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.

19.3.2.2 DTSF and ITSF

The DTSF and the ITSF each analyze their own set of input ClockTarget interfaces and, based on a specified criteria (e.g., see 19.2.7.4, 19.2.5, 19.2.6 for examples), determine which corresponding time(s) can be trusted (see 19.2.2). Being trusted does not mean the time is definitely 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 the DTSF or 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 shall be declared to be untrusted. Other conditions for determining whether a time is trusted are determined by the specified criterion.

19.3.2.2.1 DTSF

Each set of input ClockTarget interfaces that share a common dependency shall be processed as a group by one instance of the DTSF. This grouping allows each DTSF instance to produce an output that is independent from the outputs of the other DTSFs and from the other ClockTarget interfaces that are connected as inputs to the ITSF.

Each instance of the DTSF shall select one of its input times, or the Not Qualified (NQ) time (see 19.3.2.2.3) if none of its input times can be determined to be trusted, as its result. The selected time's ClockTarget interface is passed to the output of the DTSF. This output is passed to the ITSF as an independent time.

The default DTSF is described in 19.3.3.3. Other algorithms for the DTSF may also be used by the FTTM. Other algorithms for the DTSF are not required to produce the same result as the default DTSF to achieve time convergence.

19.3.2.2.2 ITSF

When operating in the enhanced availability and integrity scenario (see 19.3), the set of input ClockTarget interfaces that share no dependency with each other, which include the output ClockTarget interfaces of all instances of the DTSF, 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 19.3), it is expected that all the input times to the FTTM are connected to the ITSF regardless of their independence 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 19.3), the ITSF receives the sole input time to the FTTM. In this special scenario, the ITSF 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.

The selected time's ClockTarget interface shall be passed to the output of the ITSF and becomes the output of the FTTM, FTTM_OUTPUT. This output is passed to the application ClockTarget entity.

The default ITSF is described in 19.3.3.4. Other algorithms for the ITSF may also be used by the FTTM. Other algorithms for the ITSF are not required to produce the same result as the default ITSF to achieve time convergence.

19.3.2.2.3 Not Qualified (NQ) time

The Not Qualified (NQ) time is used to represent the condition where none of the input times to the DTSF or to the ITSF 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 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

19.3.2.3 Output ClockTarget Interfaces

The ClockTarget interfaces that pass time information from the FTTM to the application's ClockTarget entity are designated as, from the FTTM's perspective, output ClockTarget interfaces. They may be any of the following types defined in clause 9, ClockTargetEventCapture, ClockTargetTriggerGenerate, and ClockTargetClockGenerator, or they may be of another type. However, they should be consistent with the types on the FTTM's input ClockTarget interfaces. The ClockTargetPhaseDiscontinuity interface should also be provided by the FTTM to the application's ClockTarget entity.

19.3.3 Default operations

19.3.3.1 General

The default DTSF state machine, the default ITSF state machine, and the default selection algorithm used by both state machines are defined in this subclause.

Subclause 19.3.3.2 describes the default selection algorithm, with its variables and pseudo-code.

Subclause 19.3.3.3 describes the default DTSF state machine, with its variables and its state diagram.

Subclause 19.3.3.4 describes the default ITSF state-machine, with its variables and its state diagram.

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

The mid-value trusted time-index selection algorithm (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 used by the default DTSF and by the default ITSF.

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 19.2.7.4). 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.

19.3.3.2.1 MVTISA variables

The variables used in the MVTISA are described in this subclause.

19.3.3.2.1.1 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.

19.3.3.2.1.2 hystVal

The variable hystVal[x][y] holds the hysteresis added to maxAsVal[x][y] (see 19.3.3.2.1.3) for the times of the two 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 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 data type is as follows:

— Typedef UInteger32 [numTimeIndexes][numTimeIndexes] hystVal;

The values of hystVal[x][y] are passed down to this process from the default DTSF or the default ITSF from which this process is called.

19.3.3.2.1.3 maxAsVal

The value maxAsVal[x][y] is used as the criteria to determine the trustworthiness of the times from ClockTarget interfaces, with indexes x and y, to the process. See $maxAS_{xy}$ in 19.2.7.4.

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 values of maxAsVal[x][y] are passed down to this process from the default DTSF or the default ITSF from which this process is called.

19.3.3.2.1.4 medTimeIndex (fttmSelectedIndex)

The variable medTimeIndex gives the index number for the trusted input ClockTarget interface that is selected by the default DTSF instance and that is presented as the DTSF's output ClockTarget interface.

For the default DTSF instance, the range of values for the input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of NQ represents a not-qualified time

19.3.3.2.1.5 medTimeIndexChangeCnt (Uinteger16)

The variable medTimeIndexChangeCnt increments whenever the medTimeIndex value changes. The count rolls over to 0 if the count is incremented when its current value is 65535.

19.3.3.2.1.6 minValue (ExtendedTimestamp)

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

19.3.3.2.1.7 numSorted (UInteger8)

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

19.3.3.2.1.8 numTimeIndexes (UInteger8)

This variable numTimeIndexes contains the number of input ClockTarget interfaces connected to the process. This variable has a minimum value of 1. The value of numTimeIndexes does not include the NQ time (see 19.3.2.2.3).

19.3.3.2.1.9 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;

19.3.3.2.1.10 prevTimeIndexPairStatus

The variable prevTimeIndexPairStatus is a 2-dimensional array that holds the previous trust status between two ClockTarget interfaces of the default DTSF or the default ITSF. 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 defined as follows:

— Typedef fttmTrustStatus [numTimeIndexes][numTimeIndexes] prevTimeIndexPairStatus;

19.3.3.2.1.11 prevTrustState (fttmTrustState)

The variable prevTrustState holds the previous trust status of the default DTSF or default ITSF. Valid values are NO_TRUST, TIME_TRUST, and FREQ_TRUST.

19.3.3.2.1.12 timeIndexPairStatus

The variable timeIndexPairStatus is a 2-dimensional array that holds the current trust status between two ClockTarget interfaces of the default DTSF or the default ITSF. The variable timeIndexPairStatus[x][y] holds the current trust status between ClockTarget Interfaces x and y. Valid values are UNTRUSTED and TRUSTED.

The data type is defined as follows:

— Typedef fttmTrustStatus [numTimeIndexes][numTimeIndexes] timeIndexPairStatus;

19.3.3.2.1.13 timeIndexStatus (fttmTrustStatus)

The variable timeIndexStatus[x] holds the trust status of ClockTarget Interface x, Valid values are UNTRUSTED and TRUSTED.

19.3.3.2.1.14 ToD

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

For this process, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef ExtendedTimestamp [numTimeIndexes] ToD;

19.3.3.2.1.15 TodDiff

The variable TodDiff is a 2-dimensional array of extended timestamps. The array member TodDiff[x][y] gives the time skew between the two ClockTarget interfaces, x and 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;

19.3.3.2.1.16 trustState (fttmTrustState)

The variable trustState holds the type of trust the default DTSF or default ITSF has currently advanced to. Valid values are NO_TRUST, TIME_TRUST, and FREQ_TRUST.

19.3.3.2.1.17 x (UInteger8)

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

19.3.3.2.1.18 y (UInteger8)

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

19.3.3.2.2 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] = |ToD[x] - ToD[y]|
        }
    }


// ######################################################################
// Clear status before starting a new round of time index comparisons.
// ######################################################################
trustState = NO_TRUST
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] <= maxAs[x][y] &&
             prevTimeIndexPairStatus[x][y] == UNTRUSTED) ||
             (TodDiff[x][y] <= (maxAs[x][y] + hyst[x][y]) &&
              prevTimeIndexPairStatus[x][y] == TRUSTED)) &&
            (isSyncedStatus[x] && gmPresentStatus[x]) &&
            (isSyncedStatus[y] && gmPresentStatus[y]))
```

```
            {
                 // trust found for the pair
                 trustState = TIME_TRUST
                 timeIndexPairStatus[x][y] = TRUSTED
                 timeIndexStatus[x] = TRUSTED
                 timeIndexStatus[y] = TRUSTED
                 prevTimeIndexPairStatus[x][y] = TRUSTED
            }
            else
            {
                 // trust not found for the pair
                 prevTimeIndexPairStatus[x][y] = UNTRUSTED
            }
        }
    }


    // #######################################################################
    // If trustState = TIME_TRUST, find time index with the mid-value ToD.
    // #######################################################################
    // Trusted times detected.
    If {trustState == TIME_TRUST)
    {
      // Update previous trust status.
      prevTrustState = TIME_TRUST

      // 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 &&
               ToD[y] <= minValue)
           {
             minValue = ToD[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.
          medTimeIndex = orderedTimeIndex[INT((numSorted)/2)]
    }

    // If ITSFMode, check for transition to or sustaining frequency trust state.
    // If time trust just lost or if already using frequency trust, keep
    // existing median time index values if corresponding PTP Instance
    // is still valid.
    // Set previous trust status to FREQ_TRUST.
    else if {ITSFMode == TRUE &&
             prevTrustState == (TIME_TRUST || FREQ_TRUST) &&
```

```
          isSyncedStatus[medTimeIndex) == TRUE &&
          gmPresentStatus(medTimeIndex) == TRUE)
{
  medTimeIndex = medTimeIndex
  prevTrustState = FREQ_TRUST
}

// No time trust or frequency trust so set output time indexes to NQ
// and clear previous trust status to NO_TRUST.
else
{
  medTimeIndex = NQ
  prevTrustState = NO_TRUST
}

// Increment count of medTimeIndexChangeCnt if medTimeIndex changes state
if (medTimeIndex != prevMedTimeIndex)
  {
    medTimeIndexChangeCnt = medTimeIndexChangeCnt + 1
    prevMedTimeIndex = medTimeIndex
  }
```

19.3.3.3 Default DTSF state machine

19.3.3.3.1 General

The default DTSF state machine described in 19.3.3.3 operates within each instance of the default DTSF that is used in the FTTM.

The default DTSF uses the ClockTargetEventCapture interface.

19.3.3.3.2 State machine variables

The state machine variables described in 19.3.3.3.2 are unique to each instance of the default DTSF that is used in the FTTM.

19.3.3.3.2.1 gmPresentStatus

The variable gmPresentStatus is a vector of Boolean. The vector member gmPresentStatus[x] holds the gmPresent status from the ClockTargetEventCapture.result for Clock Target interface x of the default DTSF instance.

For this default DTSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] gmPresentStatus;

19.3.3.3.2.2 hystVal

The variable hystVal[x][y] holds the hysteresis added to maxAsVal[x][y] (see 19.3.3.3.2.5) for the times of the two 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 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 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 hyst management object array (see 14.23.7) based on the mapping of FTTM input Clock Target interfaces to the default DTSF instance, as determined by the fttmIndexToDtsfItsfMap management object (see 14.23.8).

19.3.3.3.2.3 isSyncedStatus

The variable isSyncedStatus is a vector of Boolean. The vector member isSyncedStatus[x] holds the isSynced status from the ClockTargetEventCapture.result for Clock Target interface x of the default DTSF instance.

For this default DTSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] isSyncedStatus;

19.3.3.3.2.4 ITSFMode (Boolean)

This variable ITSFMode is used to identify whether the MVTISA is used for default DTSF processing or for default ITSF processing.

When TRUE, the MVTISA is used for default ITSF processing.
When FALSE, the MVTISA is used for default DTSF processing.

For the default DTSF instances, the ITSFMode variable is set to FALSE.

19.3.3.3.2.5 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 19.2.7.4.

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 maxAS management object array (see 14.23.13) based on the mapping of FTTM input Clock Target interfaces to the default DTSF instance, as determined by the fttmIndexToDtsfItsfMap management object (see 14.23.8).

19.3.3.3.2.6 medTimeIndex (fttmSelectedIndex)

This medTimeIndex variable is the index number of the selected input ClockTarget interface of the default DTSF instance. This selected input ClockTarget interface is presented as the default DTSF instance's output ClockTarget interface.

For the default DTSF, the range of values for the input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of NQ represents a not-qualified time.

19.3.3.3.2.7 numTimeIndexes (UInteger8)

The variable numTimeIndexes contains the number of input ClockTarget interfaces assigned to the default DTSF instance, where the assignment is done via the fttmIndexToDtsfItsfMap management object (see 14.23.8).

19.3.3.3.2.8 prevTimeIndexPairStatus

The variable prevTimeIndexPairStatus is a 2-dimensional array that holds the previous trust status between two ClockTarget interfaces of the default DTSF or the default ITSF. 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 fttmTrustStatus [numTimeIndexes][numTimeIndexes] prevTimeIndexPairStatus;

19.3.3.3.2.9 prevTrustState (fttmTrustState)

The variable prevTrustState indicates the previously determined trust state of the default DTSF instance's state machine. Valid values are NO_TRUST and TIME_TRUST for the default DTSF instance.

19.3.3.3.2.10 ToD

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

For this default DTSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef ExtendedTimestamp [numTimeIndexes] ToD;

19.3.3.3.2.11 x (UInteger8)

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

19.3.3.3.2.12 y (UInteger8)

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

19.3.3.3.3 State diagram

The default DTSF state machine is shown in Figure 6. The default DTSF shall use the MVTISA of 19.3.3.2 to find all the trusted input times and to select the index that corresponds to ClockTarget interface that holds the median ToD value from the trusted input times. The ClockTarget interface of the selected index shall be presented as the output of the default DTSF.

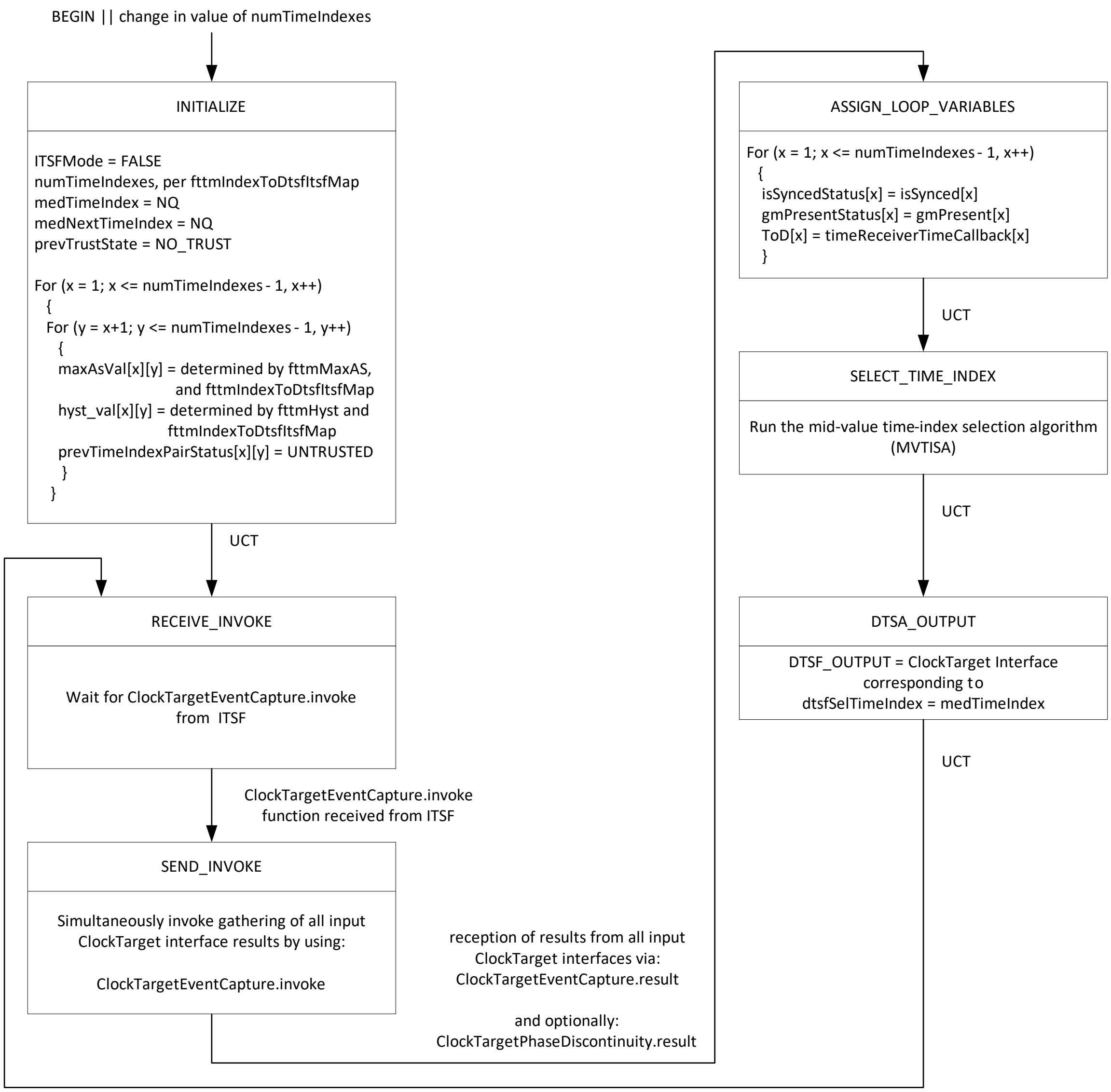


Figure 6— Default DTSF state machine

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

The RECEIVE_INVOKE state of the default DTSF state machine waits for a ClockTargetEventCapture.invoke event from the ITSF. When the event is received, the state machine unconditionally transfers to the SEND_INVOKE state.

The SEND_INVOKE state of the default DTSF state machine simultaneously sends a ClockTargetEventCapture.invoke event to all of its input ClockTarget interfaces to request timing information from them. Once all the input ClockTarget interfaces respond to the invoke event with their timing information results, this state transitions to the ASSIGN_LOOP_VARIABLES state.

The ASSIGN_LOOP_VARIABLES state of the default DTSF state machine assigns three timing result parameters (isSynced, gmPresent, and timeReceiverTimeCallback see clause 9) from each of the ClockTarget interfaces to DTSF variables. Once this assignment is finished, this state unconditionally transfers to the SELECT_TIME_INDEX state.

The SELECT_TIME_INDEX state of the default DTSF state machine runs the MVTISA (see 19.3.3.2) to find the trusted times, if any, from the input ClockTarget interfaces, sort them in ascending ToD order, and identify the index number of the input ClockTarget interface with the median ToD value. If no trusted time is found, then the index number for the NQ time is identified. This state machine then unconditionally transfers to the DTSF_OUTPUT state.

The DTSF_OUTPUT state of the default DTSF state machine passes the ClockTarget interface identified by the MVTISA of the SELECT_TIME_INDEX state to the output of the DTSF's output.

19.3.3.4 Default ITSF state machine

19.3.3.4.1 General

The default ITSF state machine uses the ClockTargetEventCapture interface.

19.3.3.4.2 State machine variables

The state machine variables described in 19.3.3.4.2 are unique to the default ITSF that is used in the FTTM.

19.3.3.4.2.1 gmPresentStatus

The variable gmPresentStatus is a vector of Boolean. The vector member gmPresentStatus[x] holds the gmPresent status from the ClockTargetEventCapture.result for Clock Target interface x of the default ITSF.

For the default ITSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] gmPresentStatus;

19.3.3.4.2.2 hystVal

The variable hystVal[x][y] holds the hysteresis added to maxAsVal[x][y] (see 19.3.3.4.2.6) for the times of the two 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 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 data type is as follows:

— Typedef UInteger32 [numTimeIndexes][numTimeIndexes] hystVal;

The value for each hystVal array member is selected from an array member from the hyst management object array (see 14.23.7) based on the mapping of FTTM input ClockTarget interfaces to the default ITSF, as determined by the fttmIndexToDtsfItsfMap management object (see 14.23.8), and on the ClockTarget interfaces selected by all the default DTSFs in the FTTM, as determined by each DTSFs medTimeIndex variable.

19.3.3.4.2.3 isSyncedStatus

The variable isSyncedStatus is a vector of Boolean. The vector member isSyncedStatus[x] holds the isSynced status from the ClockTargetEventCapture.result for Clock Target interface x of the ITSF.

For the default ITSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef Boolean [numTimeIndexes] isSyncedStatus;

19.3.3.4.2.4 ITSFMode (Boolean)

The variable ITSFMode is used to identify whether the MVTISA is used for default DTSF processing or for default ITSF processing.

When TRUE, it is used for default ITSF processing.
When FALSE, it is used for default DTSF processing.

For the default ITSF, the ITSFMode variable is set to TRUE.

19.3.3.4.2.5 fttmTimeInterfaceRateRatio

The variable fttmTimeInterfaceRateRatio[x] is equal to the ratio of the clock frequency of the time arriving on input ClockTarget interface x to the frequency of the FTTM's local clock, OSC_CLK. The variable fttmTimeInterfaceRateRatio is a vector of size numTimeIndexes of Float64 values and the data type is as follows:

— Typedef Float64 [numTimeIndexes] fttmTimeInterfaceRateRatio;

19.3.3.4.2.6 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 19.2.7.4.

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 maxAS management object array (see 14.23.13) based on the mapping of FTTM input ClockTarget interfaces to the default ITSF, as determined by the fttmIndexToDtsfItsfMap management object (see 14.23.8), and on the ClockTarget interfaces selected by all the default DTSFs in the FTTM, as determined by each DTSFs medTimeIndex variable.

19.3.3.4.2.7 medTimeIndex (fttmSelectedIndex)

The variable medTimeIndex gives the index number for the trusted input ClockTarget interface that is selected by the default ITSF and that is presented as the FTTM's output ClockTarget interface.

For the default ITSF, the range of values for the input ClockTarget interfaces ranges from 1 to numTimeIndexes and the value of NQ represents a not-qualified time

19.3.3.4.2.8 numTimeIndexes (UInteger8)

The variable numTimeIndexes contains the number of input ClockTarget interfaces assigned to and the number of DTSF ClockTarget interfaces connected to the default ITSF, where the assignments are done via the fttmIndexToDtsfItsfMap management object (see 14.23.8) and the fttmDtsfToItsfMap management object (see 14.23.6).

19.3.3.4.2.9 prevTimeIndexPairStatus

The variable prevTimeIndexPairStatus is a 2-dimensional array that holds the previous trust status between two ClockTarget interfaces of the default DTSF or the default ITSF. 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 fttmTrustStatus [numTimeIndexes][numTimeIndexes] prevTimeIndexPairStatus;

19.3.3.4.2.10 prevTrustState (fttmTrustState)

The variable prevTrustState indicates the previously determined trust state of the default ITSF's state machine. Valid values are NO_TRUST, TIME_TRUST, and FREQ_TRUST for the default ITSF.

19.3.3.4.2.11 rRlimit (Float64)

The variable rRlimit is an offset threshold, for rateRatio, at which the default ITSF determines whether it enters the FREQ_TRUST state or the NO_TRUST state. See 19.3.3.4.3.

19.3.3.4.2.12 rRSDlimit (Float64)

The variable rRSDlimit is an offset threshold, for the standard deviation of rateRatio, at which the default ITSF determines whether it enters the FREQ_TRUST state or the NO_TRUST state. See 19.3.3.4.3.

19.3.3.4.2.13 ToD

The variable ToD is a vector of extended timestamps. The vector member ToD[x] holds the latest timeReceiverTimeCallback result for input ClockTarget interface x to the ITSF (which can come from DTSFs or from FTTM input ClockTarget interfaces).

For this default ITSF, the vector is of size numTimeIndexes and the data type is as follows:

— Typedef ExtendedTimestamp [numTimeIndexes] ToD;

19.3.3.4.2.14 x (UInteger8)

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

19.3.3.4.2.15 y (UInteger8)

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

19.3.3.4.3 State diagram

The default ITSF state machine is shown in Figure 7 and Figure 8. The default ITSF shall use the MVTISA of 19.3.3.2 to find all the trusted input times and to select the index that corresponds to ClockTarget interface that holds the median ToD value from the trusted input times or the index that corresponds to the NQ time if no trusted time is available. The ClockTarget interface corresponding to the selected index shall be presented as the output of the default ITSF and, hence, as the output of the FTTM.

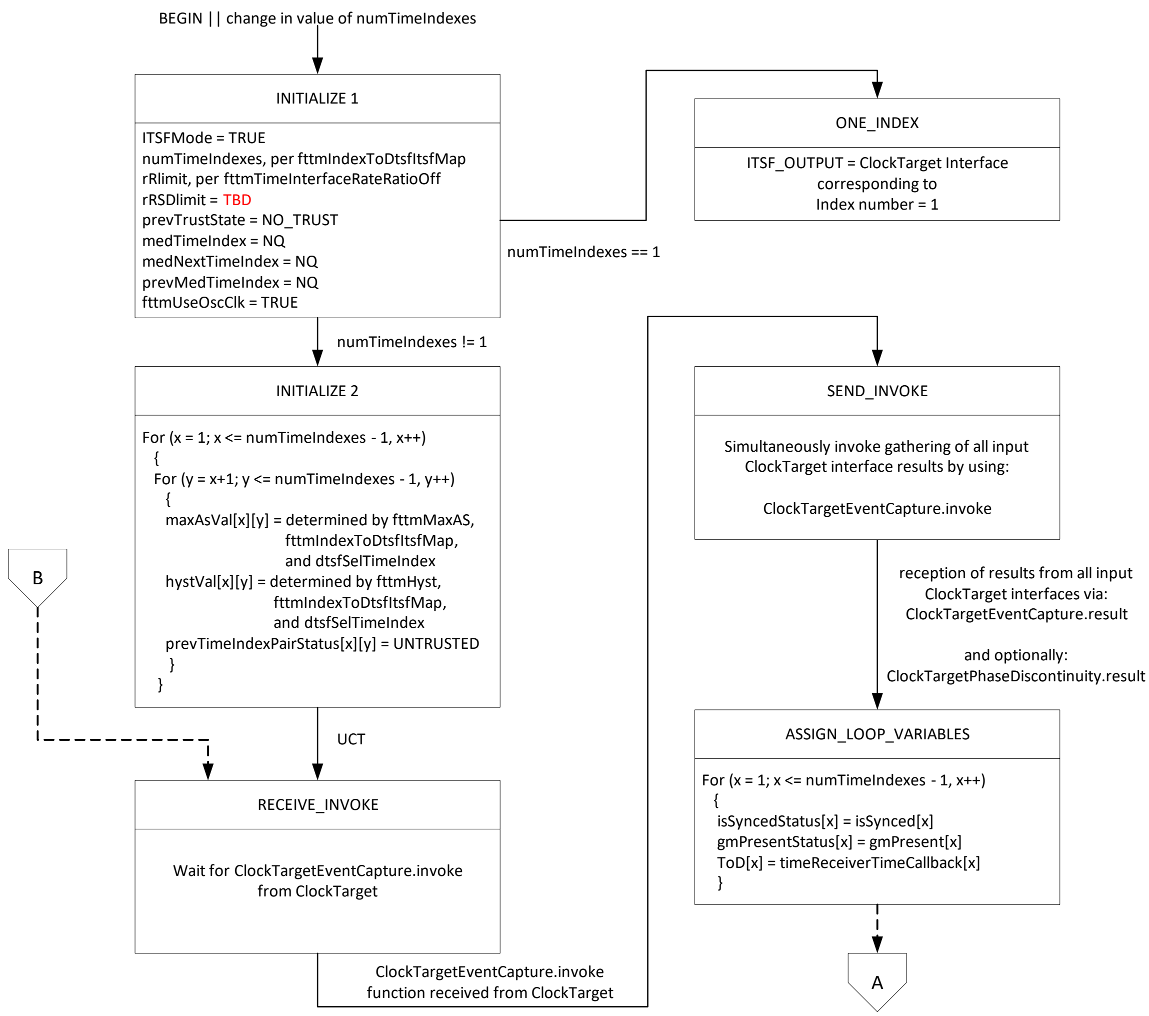


Figure 7—ITSF state machine, part 1

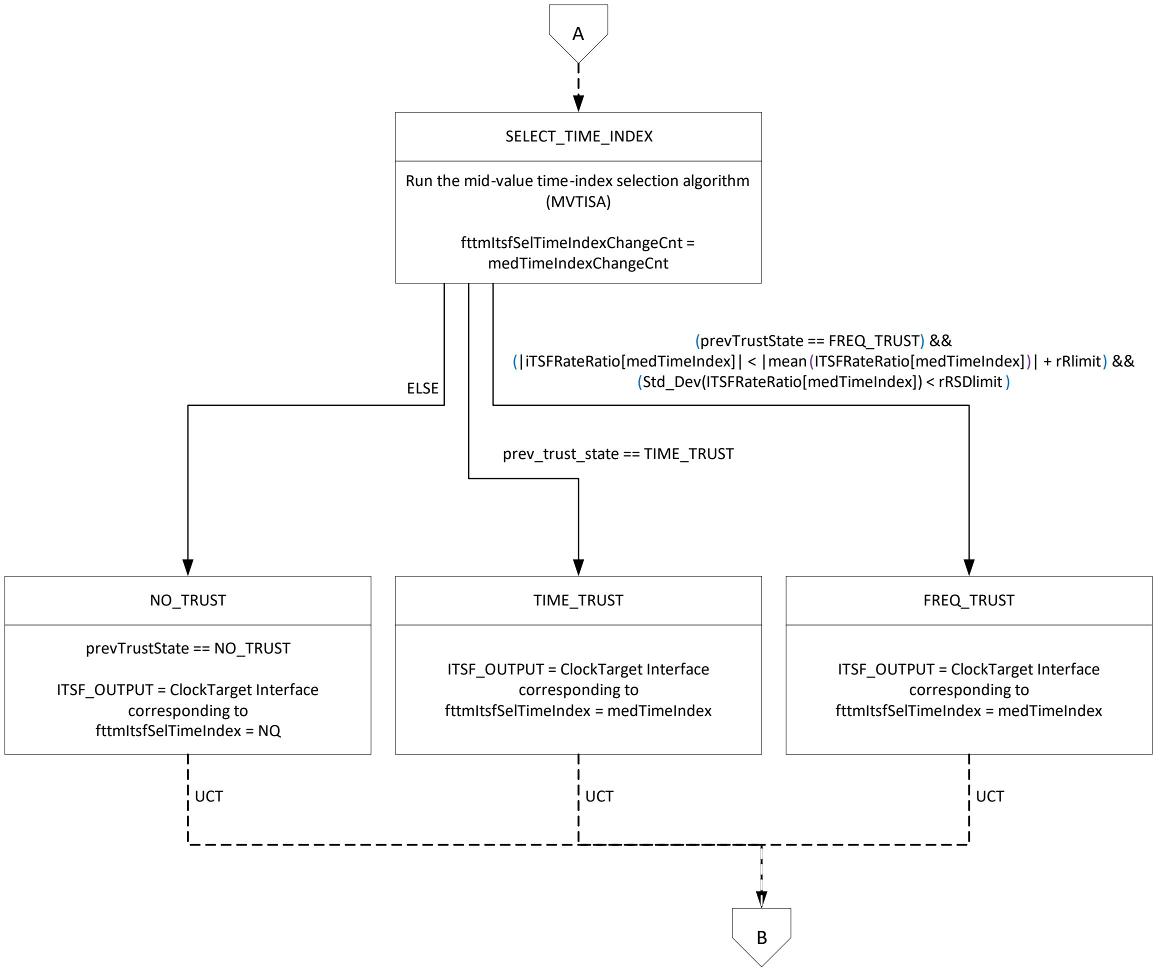


Figure 8—ITSF state machine, part 2

The INITIALIZE 1 state of the default ITSF state machine sets up the starting values of some variables. Once these starting values are configured, this state unconditionally transfers to the INITIALIZE 2 state except for the case in which the assigned value of numTimeIndexes is equal to one, in which the state instead transfers to the ONE_INDEX state.

The ONE_INDEX state of the default ITSF state machine is used when there is only one input ClockTarget interface connected to the ITSF. This corresponds to the regular availability and no integrity operational scenario described in 19.3. The state machine can only exit this state, to the INITIALIZE 1 state, via the BEGIN condition or if the value of numTimeIndexes is changed.

The INITIALIZE 2 state of the default ITSF state machine sets up the starting values of variables that are used only when the assigned value of numTimeIndexes is greater than one. Once these starting values are configured, this state unconditionally transfers to the RECEIVE_INVOKE state.

The RECEIVE_INVOKE state of the default ITSF state machine waits for a ClockTargetEventCapture.invoke event from the ClockTarget entity. When the invoke event is detected, the state machine unconditionally transfers to the SEND_INVOKE state.

The SEND_INVOKE state of the default ITSF state machine simultaneously sends a ClockTargetEventCapture.invoke event to all of its input ClockTarget interfaces. Once all the input

ClockTarget interfaces respond to the invocation with their timing information results, this state transitions to the ASSIGN_LOOP_VARIABLES state.

The ASSIGN_LOOP_VARIABLES state of the default ITSF state machine assigns three timing result parameters (isSyncedStatus, gmPresentStatus, and timeReceiverTimeCallback see clause 9) from each of the ClockTarget interfaces to ITSF variables. Once this assignment is finished, this state unconditionally transfers to the SELECT_TIME_INDEX state.

The SELECT_TIME_INDEX state of the default ITSF state machine runs the MVTISA (see 19.3.3.2) to find the trusted times, if any, from the input ClockTarget interfaces. If trusted times are found, it sorts them in ascending ToD order and identifies the index number of the input ClockTarget interface that has the median ToD value.

— If the MVTISA returns a prevTrustState = TIME_TRUST, this means that it has found a trusted time and selected a corresponding ClockTarget interface. The index of the selected ClockTarget interface is provided as the output of the SELECT_TIME_INDEX state and the state machine transitions to the TIME_TRUST state.
— If the MVTISA returns a prevTrustState = FREQ_TRUST, this means that it had, but no longer has, trusted time and is still operating with the last selected ClockTarget interface, which still has isSynced and gmPresent asserted as TRUE. This last selected ClockTarget interface is presented as the output of the SELECT_TIME_INDEX state. If this ClockTarget interface is still providing a time that is incrementing consistently with respect to the FTTM's local clock (OSC_CLK), as determined by the fttmTimeInterfaceRateRatio[x] measurement for the ClockTarget interface, then the state machine transitions to the FREQ_TRUST state.
— If neither of the above cases is true, then the ITSF has no trust in any of its input times. The state machine transitions to the NO_TRUST state.

The NO_TRUST state of the default ITSF state machine passes the ClockTarget interface corresponding to the NQ index to the output of the ITSF's output and, hence, the FTTM's output. This ClockTarget interface indicates that no trusted time is being passed from the FTTM. The state machine unconditionally transitions back to the RECEIVE_INVOKE state to wait for the start another round of searching for time with integrity.

The TIME_TRUST and the FREQ_TRUST states of the default ITSF state machine both pass the ClockTarget interface identified by the MVTISA of the SELECT_TIME_INDEX state to the output of the ITSF's output and, hence, the FTTM's output. This ClockTarget interface indicates that a trusted time is being passed from the FTTM. The state machine unconditionally transitions back to the RECEIVE_INVOKE state to wait for the start of another round of searching for time with integrity.

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.

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.

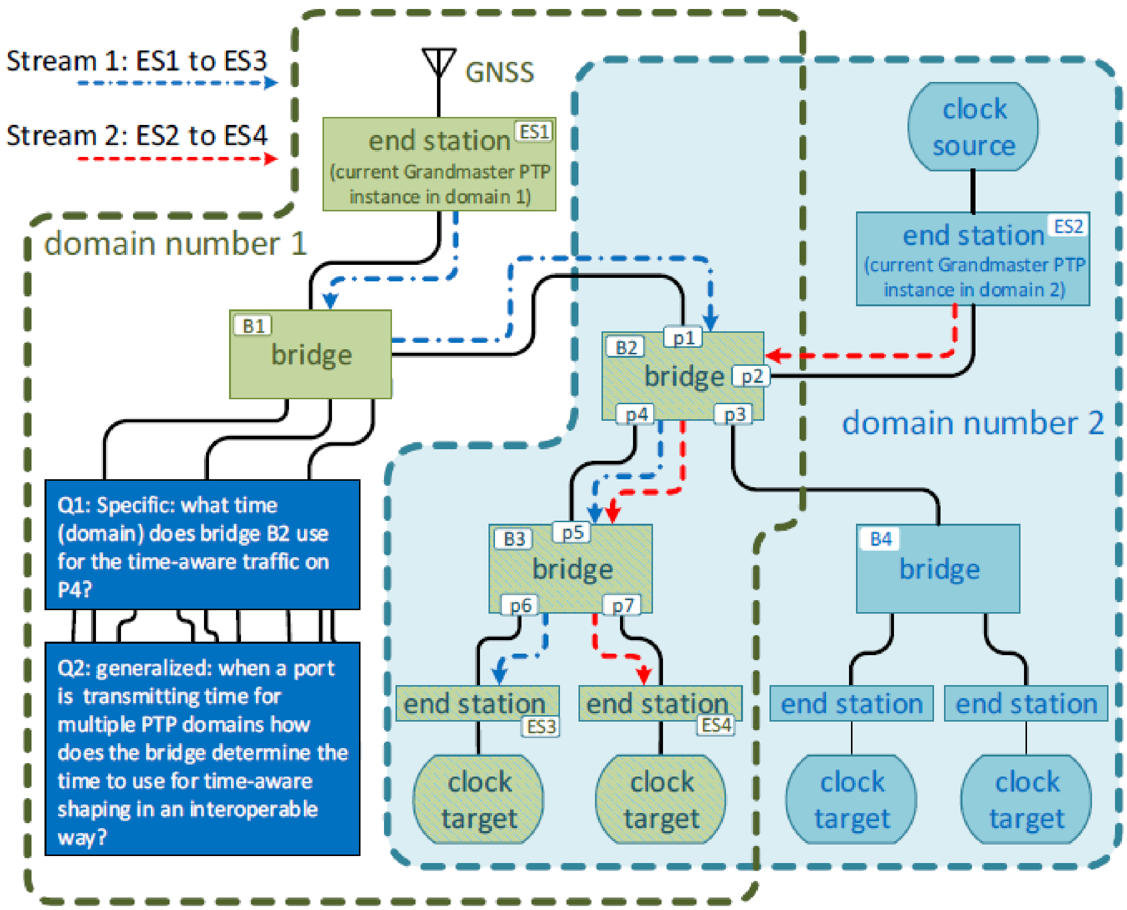
J.2 Clock domain management

As described in 19.2, 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 19.2.1).

Management of multiple PTP instances using a fault-tolerant timing module (FTTM) is discussed in 19.3.



<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 19.

There are many advanced YANG Language techniques to enforce relationships among instance data values, ensuring data integrity and coherence. These have not been added. These could be implemented once we've reached consensus on the simple module.

NOTE—YANG, a data modeling language for network configuration and management, employs several techniques to enforce relationships among instance data values, ensuring data integrity and coherence. One primary method is through "must" statements, which define conditions that data must satisfy, allowing for complex relationships and dependencies to be articulated. "When" statements are used to conditionally include or exclude parts of the model based on the values of other data, effectively linking the presence of data to specific conditions. Additionally, "leafref" types create explicit references between data elements, ensuring that values in one part of the model correspond to values elsewhere. Together, these mechanisms allow YANG to model sophisticated interdependencies within network configurations, ensuring that related data elements maintain a coherent and valid relationship throughout the model.

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-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 fault-tolerant-timing-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.

```
      <fault-tolerant-timing-system>
        <fault-tolerant-timing-function-index>1</fault-tolerant-timing-
function-index>
```

K.3 FTTM Configuration Details

Further detailing the FTTM configuration, the fault-tolerant-timing-system-ds element specifies the system's capabilities, including the number of input ClockTarget Interfaces (fttm-max-num-time-indexes), the presence of a local oscillator (fttm-use-osc-clk), and settings for DTSFs and ISTF.

```
<fault-tolerant-timing-system-ds>
  <fttm-max-num-time-interfaces>3</fttm-max-num-time-interfaces>
  <fttm-max-num-dtsfs>1</fttm-max-num-dtsfs>
  <fttm-use-osc-clk>true</fttm-use-osc-clk>
```

K.4 DTSF Configuration

The configuration continues with the definition of DTSFs within the FTTM, where each DTSF is uniquely identified for subsequent interconnection configurations.

```
<dtsf-list>
  <dtsf-instance-number>1</dtsf-instance-number>
</dtsf-list>
<mid-value-time-index-selection>0</mid-value-time-index-selection>
<user-defined>0</user-defined>
```

K.5 PTP Instance association to FTTM input indexes

The example then outlines the association of PTP Instances with the FTTM input ClockTarget interfaces, emphasizing the unique indexing of each PTP Instance within the full ieee1588-ptp-tt module. This section illustrates the configuration of three PTP Instances, with instance numbers 0x01234567, 0x89ABCDEF, and 0x10111213 associated with FTTM input ClockTarget interfaces index numbers 1, 2, and 3, respectively.

```
<fttm-ptp-instance-to-index-map-lists>
    <fttm-ptp-instance-to-index-map-list>
     <instance-number>0x01234567</instance-number>
     <fttm-input-index-number>1</fttm-input-index-number>
    </fttm-ptp-instance-to-index-map-list>
  <instance-number>0x89ABCDEF </instance-number>
  <fttm-input-index-number>2</fttm-input-index-number>
  <instance-number>0x10111213</instance-number>
  <fttm-input-index-number>3</fttm-input-index-number>
</fttm-ptp-instance-to-index-map-lists>
```

K.6 FTTM input index connection to ITSF and DTSF input indexes

The example then outlines the connection of FTTM input indexes to ITSF and DTSF input indexes, allowing the FTTM input ClockTarget Interfaces associated with the FTTM input indexes to be processed by an appropriate DTSF (as a dependent time) or by the ITSF (as an independent time).

The FTTM input index 1 is connected to input index 1 of DTSF #1. The FTTM input index 2 is connected to index 2 of DTSF #1. FTTM input index 3 is connected to ITSF input index 1. The above connections insinuate that FTTM input indexes 1 and 2 share a dependence and DTSF #1 will select between the two of them.

```
<fttmIndexToDtsfItsfMap-list>
  <fttm-input-index-number>1</fttm-input-index-number>
     <dtsf-instance-number>1</dtsf-instance-number>
     <input-index-number>1</input-index-number>
  <fttm-input-index-number>2</fttm-input-index-number>
     <dtsf-instance-number>1</dtsf-instance-number>
```

```
                <input-index-number>2</input-index-number>
             <fttm-input-index-number>3</fttm-input-index-number>
                <itsf>0</itsf >
                <input-index-number>1</input-index-number>
           </fttmIndexToDtsfItsfMap-list>
```

K.7 DTSF outputs to ITSF input indexes

The example outlines the connection of DTSF output interfaces to ITSF input indexes, 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 2. Now, each of the 3 input ClockTarget Interfaces of the FTTM have a selectable connection to the FTTM output ClockTarget Interface.

```
<fttm-dtsf-to-itsfMap-list>
   <dtsf-instance-number>1</dtsf-instance-number>
   <itsf-input-index-number>2</itsf-input-index-number>
</fttm-dtsf-to-itsfMap-list>
```

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.

As mentioned above, each PTP Instance is assigned its own unique index numbers in the YANG full ieee1588-ptp-tt module that uniquely identifies it, and this instance represents the matrix values of those index numbers.

```
<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-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.

As mentioned above, each PTP Instance is assigned its own unique index numbers in the YANG full ieee1588-ptp-tt module that uniquely identifies it, and this instance represents the matrix values of those index numbers.

```
<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-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 offset

The following XML instance data represents the fttmTimeInterfaceRateRatioOff vector and is equal to the maximum rate ratio offset of the clock frequency of the time arriving on the input interfaces to the frequency of the FTTM's local clock, OSC_CLK that allows the default ITSF to remain in the FREQ_TRUST state even when the TIME_TRUST state cannot be satisfied. In this example, the fttmTimeInterfaceRateRatioOff values are set to 0.0001 (i.e., 100ppm) for all the FTTM input indexes, 1, 2, and 3.

This example assumes that the fttmUseOscClk object is set to TRUE, enabling the OSC_CLK to be used.

```
<fttm-time-interface-rate-ratio-off-list>
  <fttm-input-index-number>1</fttm-input-index-number>
    <fttm-time-interface-rate-ratio-off>1.0001</fttm-time-interface-rate-ratio-off>
  <fttm-input-index-number>2</fttm-input-index-number>
    <fttm-time-interface-rate-ratio-off>1.0001</fttm-time-interface-rate-ratio-off>
  <fttm-input-index-number>3</fttm-input-index-number>
    <fttm-time-interface-rate-ratio-off>1.0001</fttm-time-interface-rate-ratio-off>
</fttm-time-interface-rate-ratio-off-list>
```

K.11 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.

```
        </fault-tolerant-timing-system-ds>
      </fault-tolerant-timing-system>
    </fault-tolerant-timing-service>
  </common-services>
</ptp>
```