Technical Guide

Precision Time Protocol & Transparent Clock

Feature Overview and Configuration Guide

Introduction

Precision Time Protocol (PTP) is an Ethernet or IP-based protocol for synchronizing time clocks on a collection of network devices using a timeTransmitter/timeReceiver distribution mechanism.

Both Network Time Protocol (NTP) and PTP protocols do the same thing and carry the same information. However, they do differ in how they operate, and they have different applications, with NTP providing general purpose time distribution, and PTP being used for high precision time distribution.

The **Transparent Clock** feature is an IEEE 1588 capability that is used by bridges or routers that assists other 1588 clocks in measuring and adjusting for packet delay. The Transparent Clock computes the variable delay as the PTP packets pass through the switch or the router.

Transparent Clock is available on selected AlliedWare Plus[1] platforms and is compliant with the IEEE 1588-2008 (version 2) standard, (referred to as IEEE 1588 throughout this document).

This guide describes how AlliedWare Plus uses PTP and the Transparent Clock in an IEEE 1588 network, and is limited to:

- End-to-End delay mechanism
- 1-Step end-to-end based time stamping mode

1.When using End-to-End delay mechanism

Contents

Products and software version that apply to this guide

This guide applies to all AlliedWare Plus products, that support IEEE 1588 Precision Time Protocol, running version **5.4.7** or later.

PTP in a VCStack environment is supported from version:

- 5.5.3-2.1 onwards on SBx908 GEN2, x950, and x550 Series switches.
- 5.5.3-1.1 onwards on x530L Series switches.

Note: 100G interfaces are not supported for stacking with PTP. Stack links need to be 40G rather than 100G if PTP is required on an x950 or SBx908 GEN2 stack.

Feature support may change in later software versions. For the latest information, see the following documents:

- The product’s Datasheet
- The AlliedWare Plus Datasheet
- The product’s Command Reference

These documents are available from the above links on our website at alliedtelesis.com.

Terminology

This document describes clock roles and PTP Port states using the optional alternative terms - timeTransmitter, instead of master, and timeReceiver, instead of slave, as defined in:

- Institute of Electrical and Electronics Engineers, "IEEE std. **1588g-2022**, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Amendment 2: Master-Slave Optional Alternative Terminology”, December 2022, www.ieee.org.

Overview of the PTP feature

PTP is a network-based time synchronization standard, but instead of millisecond-level synchronization, PTP networks aim to achieve nanosecond or even picosecond-level synchronization.

PTP time stamping is so accurate because it uses hardware time stamping instead of software, and PTP equipment is dedicated to one specialized purpose: keeping devices synchronized. For that reason alone, PTP networks have much sharper time resolutions, and unlike NTP, PTP devices will actually time stamp the amount of time that synchronization messages spend in each device, which accounts for device latency.

In this section we describe the type of applications where precise Ethernet or Ethernet/IP timing is used, the wall clock concept, synchronization and clock types, delay mechanisms, and time stamping modes.

Applications

There are several applications that require precise timing using Ethernet or Ethernet/IP:

- Telecom - applications, such as cellular, where not only frequency, but also phase precision is needed in order to control hand-off of mobile phones from one cell tower to the next.
- Factory Automation - legacy field buses are being replaced with Ethernet-based field buses and timing is needed for drive controllers and distributed I/O. For example, robots working in conjunction with automation conveyor belts need to be precisely timed.
- Power Substations - legacy process buses in power substations are being replaced with Ethernet-based buses. These buses have several components that need precision time for handling power transfers and recording fault measurements.

What is time?

IEEE 1588 uses the **Wall Clock** concept, that is it uses the Time of Day (ToD), not only hours, minutes, and seconds, but also the number of nanoseconds within the second. More precisely, PTP carries time that consists of 48 bits worth of seconds, and 32 bits worth of nanoseconds. The 48 bit seconds is actually the number of seconds that have elapsed since January 1, 1970 at midnight, which is known as the PTP epoch.

Synchronization types

Applications can use IEEE 1588 to achieve the following types of synchronization:

Frequency The nodes in the network have the 48 bits and in particular the 32 bits of time change at the same **rate**, without caring necessarily what the 48/32 bit values are. Telecom applications initially used IEEE 1588 to distribute frequency only.

Phase The nodes in the network not only have the 48/32 bits of time changing at the same rate, but have at least the seconds boundary time occurring at the same **time**. That is to say, when the nanoseconds time rolls over and increments the seconds time, all nodes do this at the same time. These nodes may not necessarily need to know what the year, month, day, and hour are.

Time of Day The nodes in the network are not only frequency and phase synchronized, but they also want to know what the year, month, day, hour, and seconds are, along with nanoseconds.

IEEE 1588 roles

To distribute time, IEEE 1588 is made up primarily of two main roles:

- **timeTransmitter** - the entity distributing time to timeReceivers. A timeTransmitter can also be a Grand Master (GM), that gets its time from a primary reference source, typically a GPS satellite signal.
- **timeReceiver** - the device synchronizing to the remote timeTransmitter.

IEEE 1588 clock types

There are different kinds of clocks defined in IEEE 1588:

- **Ordinary Clock** - only has one clock port. For an Ordinary Clock, a clock port is abstract and may not necessarily correspond to a physical port. The following can be an Ordinary Clock:
  - timeTransmitter - this distributes IEEE 1588 time to timeReceivers
  - timeReceiver - this synchronizes IEEE 1588 time with the timeTransmitters.
- **Boundary Clock** (BC) - has multiple clock ports, abstract or not. Unlike the Grand Master, which communicates individually with each timeReceiver, a Boundary Clock helps scale by combining a timeReceiver half and a timeTransmitter half. The timeReceiver half synchronizes using one clock port, while the timeTransmitter half uses other clock port(s) to distribute the clock to its set of timeReceivers
- **Transparent Clock** (TC) - assists in the delay measurement between a timeTransmitter and a timeReceiver by including a Correction Factor (CF) that tells the timeReceiver how much delay the Transparent Clock added. This in general applies to Layer 2 bridges, and a Transparent Clock's clock ports correspond to physical interfaces.

IEEE 1588 delay mechanisms

As detailed in the next section, in order for PTP nodes to achieve phase or Time of Day synchronization, timeReceivers have to determine the delay between themselves and the timeTransmitter. IEEE 1588 defines two different delay mechanisms that can be used to determine the delay between a timeTransmitter and a timeReceiver, they are End-to-End and Peer-to-Peer, but only one can be used at a time.

End-to-End (Delay Request/Response)

This delay mechanism requires the timeReceiver to measure the delay between itself and the timeTransmitter (thus End-to-End). The timeTransmitter and timeReceiver send IEEE 1588 messages called DELAY REQUEST and DELAY RESPONSE back and forth between the two, allowing the delay to be measured. Once the delay is known, the timeReceiver can adjust its time to be phase and Time-of-Day synchronized with the timeTransmitter.

In the ideal case, the delay between the timeTransmitter and the timeReceiver is constant, such as when using a wire. In practice, there are Layer 2 and/or Layer 3 devices in between that make the delay variable. As will be seen later, the role of a Transparent Clock is to add a correction to certain PTP messages which assists the timeReceiver in removing this variable delay.

Peer-to-Peer

This delay mechanism requires each network element to measure the delay between its input port and the device attached on the other end of the wire of this input port (the peer device). As the timeTransmitter sends its view of time (using SYNC messages) towards timeReceiver(s), each network element along the way receives the SYNC message and adds a correction to the SYNC message. The correction includes the measured wire delay of the input port the SYNC message was received on.

For Transparent Clocks, the correction also includes the delay through the bridge. This correction is cumulative as it traverses nodes hop-by-hop. As the SYNC message finally arrives at a timeReceiver, the accumulative correction in the SYNC message will contain the total delay from the timeTransmitter to the timeReceiver. This eliminates the timeReceiver from having to send messages back and forth with the timeTransmitter. Peer-to-Peer is a newer IEEE 1588 technology, and not all devices deployed today support Peer-to-Peer.

Technically, End-to-End delay mechanism can be used at the same time as Peer-to-Peer delay mechanism as long as the two are not used along the same IEEE 1588 messaging path. The different delay mechanisms are mutually exclusive of each other. That is to say, between a timeTransmitter and timeReceiver inclusive, all the IEEE 1588 participating nodes must use either the End-to-End or Peer-to-Peer delay mechanism but can not use a mix.

IEEE 1588 time stamping modes

The key to IEEE 1588's accuracy is the ability to Time Stamp specific IEEE 1588 messages as they enter and as they leave a physical interface. Hardware is most often used to accomplish this and is done as close to the edge of the physical port's hardware as possible. There are two types of time stamping modes used, 1-step and 2-step:

1-step

A Time Stamp is captured in real time as the message starts transmitting out the physical port. The message is edited on the fly to carry the captured time stamp.

2-step

A Time Stamp is captured in real time as the message starts transmitting out the physical port, but the implementation is unable to edit the packet on the fly, and thus can not carry the captured time stamp. A secondary message is used instead to carry the captured time stamp. Early implementations of IEEE 1588v2 used the 2-step time stamping mode, as the earlier hardware was unable to edit the message.

Most modern implementations support the 1-Step time stamping mode.

How the Transparent Clock works

These next sections describe how the Transparent Clock is used in an IEEE 1588 network using the **End-to-End** delay mechanism. But first, here is a brief overview of IEEE 1588v2 messaging.

IEEE 1588 messaging

IEEE 1588 supports a variety of messages. Not all of these are of interest to Transparent Clocks but they are listed here for completeness:

Event Messages - Those that require time stamping:

- SYNC - used by the timeTransmitter to convey time. It is used in all scenarios.
- DELAY REQUEST - used between the timeTransmitter and timeReceiver when using End-to-End delay Mechanism to measure delay. The timeReceiver sends this message to the timeTransmitter.
- PEER DELAY REQUEST - used between IEEE 1588 devices to measure the delay of an incoming link. Only used when Peer-to-Peer delay mechanism is used.
- PEER_DELAY_RESPONSE - used between IEEE 1588 devices to measure the delay of an incoming link. Only used when Peer-to-Peer delay mechanism is used.

General Messages - Those that do **not** require time stamping:

- FOLLOW UP - used by the timeTransmitter to convey a captured time stamp of a transmitted SYNC message. This is used in 2-Step mode to send the earlier captured time stamp of a SYNC message.
- DELAY RESPONSE - used between the timeTransmitter and timeReceiver when using End-to-End delay Mechanism to measure delay. The timeTransmitter uses this to respond to the timeReceiver.
- PEER DELAY RESPONSE FOLLOW UP - used between IEEE 1588 devices to measure the delay of an incoming link. Only used when Peer-to-Peer delay mechanism is used with 2-Step mode.
- ANNOUNCE - is sent and received by local clock ports with a variety of information. It can be used to determine which one out of several possible timeTransmitters is to be selected as the Best timeTransmitter. It can also be used between timeTransmitter and timeReceiver to negotiate a unicast service.
- MANAGEMENT - used between management devices and clocks.
- SIGNALLING - used by clocks for conveying things such as how often to send messages, supporting unicast services instead of multicasting etc.

A Transparent Clock is primarily interested only in **Event messages**.

Ideal IEEE 1588 End-to-End delay mechanism

Figure 1 below shows a timeTransmitter Ordinary Clock and a timeReceiver Ordinary Clock that are connected together by a yellow Ethernet cable. In this figure, the timeTransmitter can be considered as the Grand Master (GM) where it has acquired Time of Day (ToD) synchronization from a system such as a Global Positioning System (GPS).

The timeTransmitter as well as the timeReceiver have a running time stamp clock that is available at its physical port. The timeReceiver's time clock however is not yet synchronized with the timeTransmitter, and the following describes how synchronization is achieved:

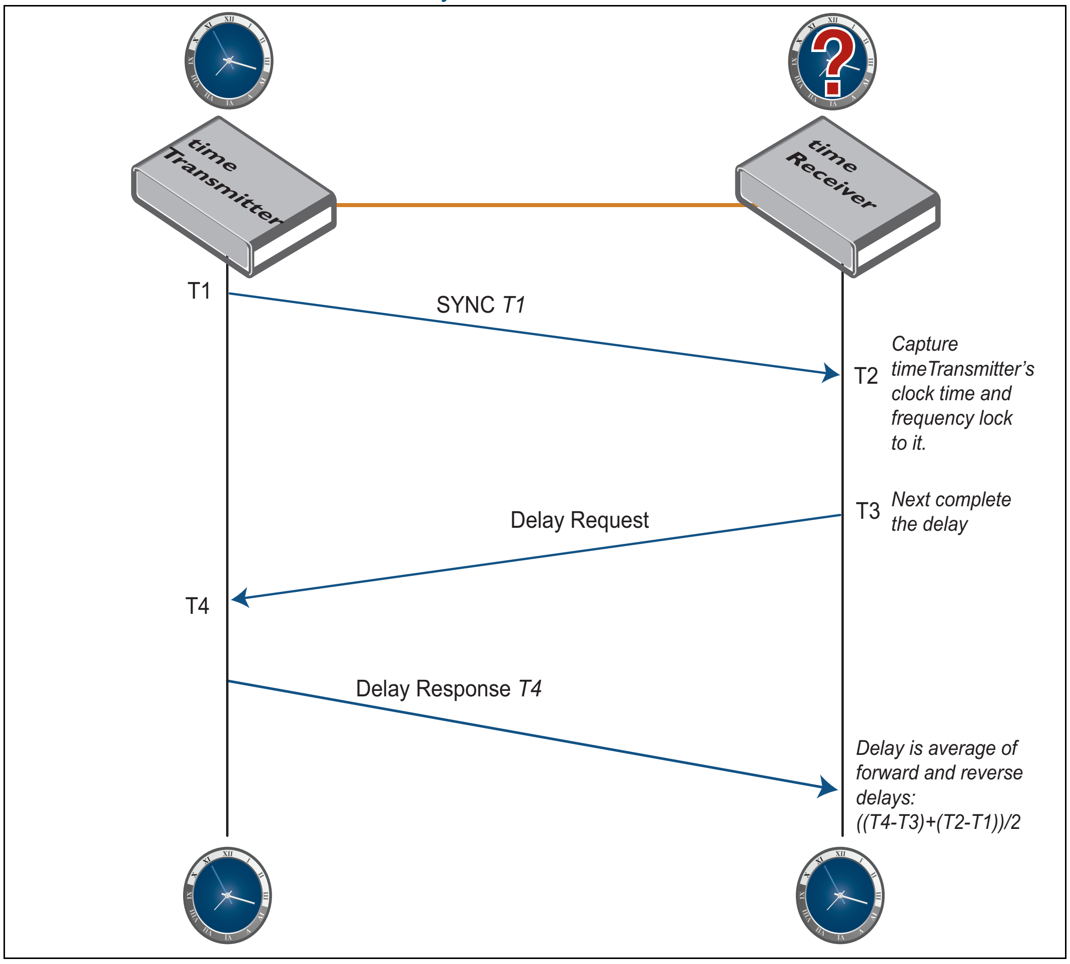
A typical PTP sequence involves a series of four messages between timeTransmitter and timeReceiver:

- The initial sync message from timeTransmitter to timeReceiver
- A delay request message from timeReceiver to timeTransmitter
- A final delay response message from timeTransmitter to timeReceiver

This sequence produces four different time stamps:

- T1 when the timeTransmitter sends the initial sync message
- T2 when the timeReceiver receives the initial sync message
- T3 when the timeReceiver sends the delay request
- T4 when the timeTransmitter receives the delay request

Figure 1: Ideal IEEE 1588 end-to-end delay mechanism


time Transmitter
time Receiver
T1
SYNC *T1*
T2
Capture timeTransmitter's clock time and frequency lock to it.
T3
Next complete the delay
Delay Request
T4
Delay Response *T4*
Delay is average of forward and reverse delays: ((T4-T3)+(T2-T1))/2

Here is a more detailed look at the PTP sequence shown above:

1. The IEEE 1588 timeTransmitter periodically sends out an IEEE 1588 SYNC message to the IEEE 1588 timeReceiver device. As the SYNC message leaves the timeTransmitter's physical interface, it captures a running time stamp in the timeTransmitter, shown as T1. In the 1-Step mode that is being illustrated here, the timeTransmitter sets the 'Origin Time Stamp' field in the SYNC message to T1 before the message completely exits the interface.
2. The IEEE 1588 timeReceiver receives the SYNC message and its running time stamp clock captures the time (T2) that the SYNC message starts to arrive at its physical port.
   - Although the timeReceiver could set its time stamp clock to that of the timeTransmitter using T2, it would leave the timeReceiver's clock in an inaccurate state due to the propagation delay of the wire. Also, the timeReceiver's time stamp clock will be running slightly faster or slower than the timeTransmitter's. In the beginning stages, although not shown, the next stage of operation is for the timeReceiver to try and frequency lock its running clock with the timeTransmitter's. During this phase, the timeReceiver will only receive SYNC messages until it believes its time stamp clock is changing at the same rate as the timeTransmitter's.
   - After frequency locking, the timeReceiver will next proceed to determine what the delay is between itself and the timeTransmitter.
3. The timeReceiver next calculates what the delay is by sending a DELAY REQUEST message to the timeTransmitter. As the message starts to be transmitted out the timeReceiver's physical

interface, the timeReceiver's running time stamp clock is used to capture the time (T3) and the timeReceiver stores this time while it waits for the reply.

4. The timeTransmitter receives the DELAY REQUEST and uses the timeTransmitter's running time stamp clock to capture the time (T4) as the message starts to be received on its physical interface. After retrieving the captured T4 value, the timeTransmitter will shortly thereafter send the timeReceiver a DELAY RESPONSE containing the captured T4 value.

5. The timeReceiver receives the DELAY RESPONSE message and extracts the T4 value in it.

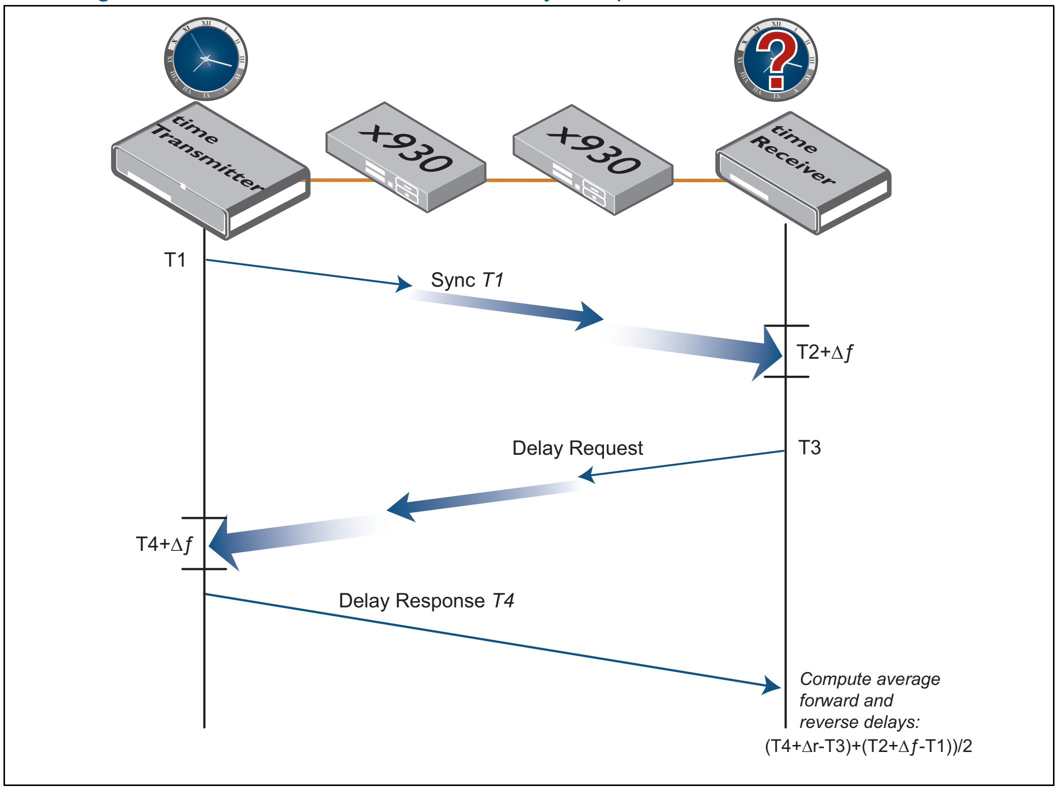
   - The timeReceiver can calculate the reverse delay as (T4-T3). It can then adjust its time stamp clock to account for the wire delay, at least in the beginning stages. After a few iterations of this to make sure the reverse delay measurement is stable, the timeReceiver can now measure the forward delay using captures of (T2-T1).

   - Finally, instead of using just the reverse delay, IEEE 1588 uses both the forward and reverse path delay in steady state to account for the wire delay. This delay is called the **mean path delay**, calculated as {(T4-T3) + (T2-T1)}/2. Once this is computed, the timeReceiver will readjust its clock to align with the timeTransmitter's which now takes into account the wire delay.

How bridges can introduce clock errors

The figure below shows how adding a couple of Ethernet bridges can introduce errors in the delay computation. To start with, the timeReceiver is not synchronized with the timeTransmitter.

Figure 2: Bridges can introduce clock errors in delay computation



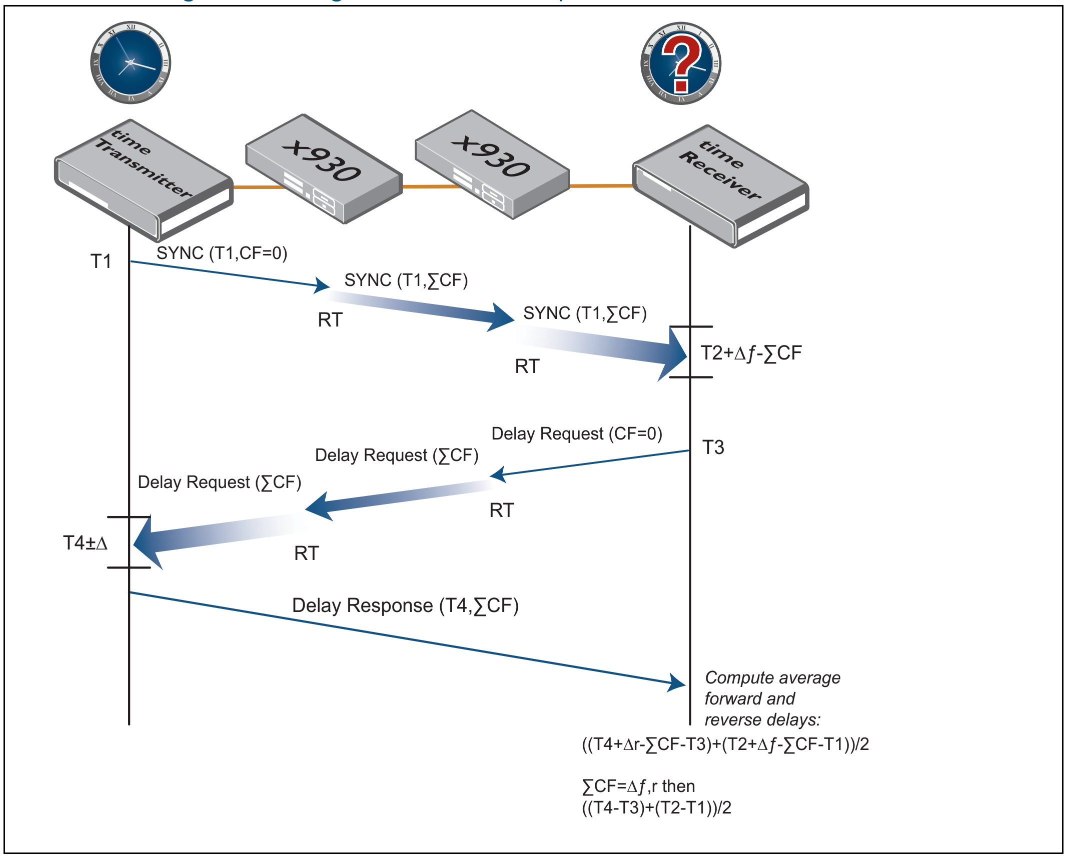
- As the SYNC message leaves the timeTransmitter, it traverse a couple of regular Ethernet bridges which do not participate in IEEE 1588.
- Bridges forward the SYNC message with a delay known as Residence Time (RT), influenced by traffic load and queuing delays. Under no load conditions, the queuing delay can be nearly zero, leaving a fairly constant propagation time through the bridge. Under high load conditions, queues may cause delays in IEEE 1588 messages, resulting in variable delays that worsen with additional bridges in the path.
- The same occurs in the reverse direction for the DELAY REQUEST messages. (The DELAY RESPONSE message is not time critical so bridge delays have no impact on them).
- The timeReceiver receives several SYNC messages over time, each have a different delay denoted by T2+Delta-Forward. The timeReceiver also receives several DELAY RESPONSE messages containing T4 which will also vary by T4+Delta-Reverse.

The result is that the timeReceiver's computation of the mean delay will be jittered by Delta-Forward and Delta-Reverse and in turn the timeReceivers time clock will be jittered as well as it constantly readjusts its time clock.

How the End-to-End Transparent Clock overcomes clock errors

The figure below shows how clock errors can be overcome when Ethernet bridges implement a Transparent Clock by adding an accumulative correction.

Figure 3: Overcoming errors using end-to-end Transparent Clock



- The timeTransmitter sends a SYNC message which in addition to containing a Origin Time Stamp field (which contains the timeTransmitter's Time of Day), also contains a Correction Field (CF). The timeTransmitter sets the CF to 0 prior to sending the SYNC message.
- As the SYNC message arrives at the first bridge, the bridge uses its time stamping clock to capture the time the message starts arriving at its physical port. Similarly as the SYNC message starts to leave the bridge's physical port heading toward the timeReceiver, it uses the same time stamping clock to capture the time the message starts leaving the physical port. With these two time stamp values, the bridge is able to calculate the Residence Time (RT) of the message, which is the delay through bridge, by taking the difference of the two captured time stamps[2]. As a 1-Step device, the bridge will add its residence time to the CF of the SYNC message as it leaves the exiting physical interface heading towards the timeReceiver. In the figure, this is shown as SumCF.
- The SYNC message arrives at the second bridge which also calculates its Residence Time and 'adds' it to the CF portion of the SYNC message. The CF is accumulative as makes its way to the

2.The transparent clock's time stamping clock need not be synchronized with a timeTransmitter, as it is computing the difference.

timeReceiver. The timeReceiver then receives the SYNC message, and captures its time stamp (T2) and extracts and stores away both T1 and CF from the message.

- Similarly in the reverse direction, the timeReceiver sends out a DELAY REQUEST message to the timeTransmitter with CF=0 and captures T3, and the bridges add their Residence Time to the CF portion of the DELAY REQUEST message in an additive way. When the DELAY REQUEST message arrives at the timeTransmitter, the timeTransmitter time stamps as usual (T4). The timeTransmitter then takes both the T4 captured time stamp and the CF that arrived and sends a DELAY RESPONSE to the timeReceiver.
- The timeReceiver now has in this one sequence of SYNC, DELAY REQUEST, DELAY RESPONSE messages: T1, T2 and accumulative CF in the forward direction, as well as T4, T3 and the accumulative CF in the reverse direction. At this point, the time delay error that was introduced by both bridges (shown as Delta) has been told to the timeReceiver via the accumulative CFs. From the figure, this is shown as SumCF = Delta-Forward or Delta-Reverse. The timeReceiver can effectively subtract out the bridge's residence time (which may be small under low traffic load conditions, or large under high load conditions) during each message sequence. This leaves the timeReceiver with the wire delay and thus is the same as in the first case above when no bridges were used.

Note: Note that IEEE 1588's End-to-End delay mechanism is susceptible to Layer 2 or Layer 3 topology changes. Although End-to-End Transparent Clock bridges eliminate their delay, wire delay remains a factor. Topology changes affecting wire delay may cause the timeReceiver to adjust its clock, introducing some stabilization time to accommodate the new delay

Limitations

The current AlliedWare Plus implementation of the IEEE 1588 Transparent Clock protocol has the following restriction:

- PTP messages (except for PEER_DELAY_REQUEST/RESPONSE) are switched through the AlliedWare Plus Layer 2 Transparent Clock like any other tagged or untagged packets. You must set up the VLAN that switches these messages on all the ports that are part of an IEEE 1588 domain.
- 2-Step mode is not supported. Only 1-Step is supported.
- Peer-to-Peer delay mechanism is not supported. Only End-to-End delay mechanism is supported.

Configuring PTP

PTP has two configuration levels:

- **Global** - where you can enable or disable PTP globally on a device. You can also configure various PTP clock parameters to help determine which clock in the network has the highest priority to be selected as the Grand Master.
- **Port** - where you can configure the interface port or LAG that is to be used for PTP Transparent Clock.

Global PTP clock configuration

AlliedWare Plus supports a **single** instance of a PTP clock. To create and configure the global PTP clock instance, use the following command:

```
awplus(config)# ptp-clk transparent transport-type ethernet delay-
mechanism e2e step-type onestep
awplus(config-ptp-clk)#
```

Note: For a Transparent Clock using the End-to-End Delay mechanism, the transport-type setting is ignored in general, as target AlliedWare Plus devices can handle IEEE 1588 packets in any of the above encapsulation types.

There are other options available on the CLI for this command, but they are not currently supported.

To destroy the PTP clock instance, use the following command:

```
awplus(config)# no ptp-clk
```

Note: This will also destroy the PTP port configurations if any such configuration already exists. See the command **clock-port**.

Enable PTP

By default, PTP is disabled. To enable the PTP clock in the AlliedWare Plus device, use the following command:

```
awplus(config)# ptp global
```

To disable the PTP clock in the AlliedWare Plus device, use the following command:

```
awplus(config)# no ptp global
```

Note this command can be used even if there is no global configuration yet.

Port PTP clock configuration

Configure the switch port or LAG that is to be used for PTP Transparent Clock. This allows the Correction Field to be updated as PTP Event packets are forwarded through the switch. Do this for every interface port or LAG that is to participate in PTP Transparent Clock. Unconfigured ports will not update the Correction Field correctly and consequently their value will be indeterminate. For a Transparent Clock, Ethernet switch ports or LAGs ports are supported on AlliedWare Plus devices.

Although only one global PTP Clock instance is supported, it must be configured first before configuring the PTP Port. See the command **ptp-clk**

To configure a PTP port, first get into the interface port or interface LAG context, then simply enter the **clock-port** command which allows Transparent Clock processing on that interface:

```
awplus(config-if)# clock-port
awplus(config-clk-port)#
```

Note: For AlliedWare Plus devices operating as End-to-End Transparent clocks, PTP packets that are encapsulated in Ethernet are Layer 2 switched, based on VLAN. For PTP frames arriving on one set of interface ports and needing to egress other interface port(s), ensure that the VLAN is a member of the necessary set of ports. Also ensure that the tagging of the VLAN on those ports is set properly.

PTP configuration example

1. First, configure the global PTP clock instance parameters. In this example, the AlliedWare Plus device is configured for a Transparent Clock using Layer 2 Ethernet as the Transport Layer and for use with an End-to-End delay mechanism:

```
awplus(config)# ptp-clk transparent transport-type ethernet delay-
mechanism e2e
awplus(config-ptp-clk)# exit
awplus(config)#
```

2. Next, enable the global PTP clock instance:

```
awplus(config)# ptp global
```

3. Finally, go to the interface ports that are to be used for PTP. Assuming the PTP messages are to be switched untagged through the device, first configure an untagged VLAN for the ports. In this example, it also assumes the ports have already been configured in trunk mode, and the untagged VLAN for IEEE 1588 PTP frames is to be configured as the native VLAN.

```
awplus(config)# interface port1.0.3-1.0.4
awplus(config-if)# switchport trunk native vlan 300
awplus(config-if)# clock-port
awplus(config-clk-port)# exit
awplus(config-if)#
```

Monitoring PTP

Global Use the **show ptp data transparent** command to show the **global** PTP clock's configuration.

```
awplus# show ptp data transparent
CLOCK(Transparent)
Global Enable          :  Enabled
Clock Identity         :  00:0c:25:ff:fe:03:92:47
Number Of Ports        :  2
Transport Type         :  Ethernet
Delay Mechanism        :  End To End
Primary Domain         :  0
Step-type              :  One Step
```

Where:

- **Clock** <***clock-mode***> - conveys the type of clock this instance is running. 'Transparent' is the only mode currently supported.
- **Global Enable** - indicates that the global PTP clock is enabled or disabled. See the command **[no] ptp global**. If disabled, then this device will not do any IEEE 1588 processing.
- **Clock Identity** <***clk-id***> - this device's Clock Identity using EUI-64 assigned numbers. Example: 00:0c:25:ff:fe:26:95:bf. For End-to-End TC, Clock Identity is not used but is shown here for informational purposes.
- **Number of Ports** <***number-ports***> - this is the number of interface ports and interface LAGs that have been configured to run PTP.
- **Transport Type** - this indicates the IEEE 1588 message encapsulation that this device supports. Currently only 'Ethernet' encapsulation is supported, as far as the configuration is concerned. However, the 'Ethernet' setting also supports IPv4 and IPv6 encapsulations.
- **Delay Mechanism** - this is the delay mechanism configured for this clock instance. Only 'End to End' is currently supported.
- **Primary Domain** <***domain***> - for Transparent Clock, this indicates the primary domain that the Transparent Clock is syntonized to. Syntonization[3] is not currently supported. This will always show the IEEE 1588 default value of '0'.
- **Step-type** - '1-Step' is currently supported.

3. **Syntonization**, in simple terms, is the process of accounting for the clock frequency difference between timeTransmitter and timeReceiver.

Port/LAG Use the **show ptp port** command to display the PTP clock port(s) configuration along with the IEEE 1588 PTP ‘Data Set’ as per the standard.

```
awplus# show ptp port
======================================================================
% Transparent clock
Global Enable                                :  Enabled
Clock Identity                               :  00:0c:25:ff:fe:03:92:47
Port Number                                  :  1 (Interface port1.0.1)
Peer Delay Request Interval (log base 2)     :  1
Peer Mean Path Delay                         :  0
Faulty Flag                                  :  False
======================================================================
% Transparent clock
Global Enable                                :  Enabled
Clock Identity                               :  00:0c:25:ff:fe:03:92:47
Port Number                                  :  2 (Interface port1.0.2)
Peer Delay Request Interval (log base 2)     :  1
Peer Mean Path Delay                         :  0
Faulty Flag                                  :  False
```

Where:

- **Global Enable** - this indicates whether the global PTP configuration has been enabled or not. See the command: **[no] ptp global**.
- **Clock Identity** <***clk-id***> - this device's Clock Identity using EUI-64 assigned numbers. Example: 00:0c:25:ff:fe:26:95:bf. Note: Except for Transparent Clock using End-to-End delay mechanism, most other clock types use the Clock Identity in 1588 Messages. For End-to-End TC, Clock Identity is shown here for informational purposes.
- **Port Number** <***port-id***> - this is the interface port number or interface LAG number. Example: port1.0.4, sa1, po1. The output of this show command is repeated for each interface that has been configured. ‘Port Number’ is an integer and starts with the lowest current value in use along with the corresponding interface shown. ‘Port Number’ increments sequentially.
- **Peer Delay Request Interval** <***pdelay-req-intvl***> - for Transparent Clocks running Peer-to-Peer delay mechanism, this is the interval for sending Pdelay Request messages shown in log base 2. For Transparent Clocks running End-to-End, this is not applicable. Only End-to-End delay mechanism is currently supported and thus the interval is considered as 0 which for log base 2 is always shown as ‘1’.
- **Peer Mean Path Delay** <***peer-path-delay***> - for Transparent Clocks running Peer-to-Peer delay mechanism, this is the measured mean delay to the peer over this port. For Transparent Clocks running End-to-End, this is not applicable. In this version, only End-to-End delay mechanism is supported and is always shown as ‘0’.
- **Faulty Flag**- for Transparent Clock, this indicates whether the interface port or interface LAG is operationally down, shown as ‘True’, or operationally up, shown as ‘False’.