PTP Configuration Guide

Application Note

CONFIDENTIAL

AN1295
Rev. APPL-2024.06
2024-06-28

Table of Contents

1. Introduction

This configuration guide provides information on configuration and status verification of various PTP protocol profiles and their features using Istax application on Microchip’s UNG platforms.

Precision time protocol or PTP is used for synchronising clocks through computer networks. PTP works between two clocks or devices in which the reference device acts as clock source or server and another device acting as client synchronises to this server. After exchanging messages containing timestamps with the server, the client device calculates the offset of its local clock from the server clock and adjusts its local clock accordingly. Local clock adjustment can be done using frequency adjustment or by stepping the clock time to expected time.

In Istax application, frequency can be adjusted using dpll(Digital phase locked loop) or through LTC(local time counter) of timestamping chip. Stepping the clock time is done by adjusting the LTC. When there is dpll on the board and if there is no explicit adjustment configuration, dpll controls the clock to all the devices on the board. If there is no dpll on board, frequency must be updated in LTC of the underlying timestamping chip.

On all Microchip platforms running Istax application, timestamping can be done either by Mac or by phy. If the phy existing on the port supports timestamping, then phy timestamping is used by the PTP protocol. If the port does not contain phy or if the phy does not support timestamping, then Mac timestamping is used by the PTP protocol.

Istax application also supports Synce or synchronised ethernet protocol which can be used for synchronising frequency using dpll and underlying physical layer of ethernet. Some of the telecom PTP profiles work in tandem with Synce.

Istax application supports various PTP profiles such as standard IEEE 1588 profile, telecom profiles such as 8275.1, 8275.2 and 8265.1, industrial profile like 802.1as profile. Telecom profiles like 8275.1 use synce frequency for more accurate frequency synchronisation. Configuration of various profiles and features is described in following sections.

2. Configuration and status of various profiles

2.1. Web configuration of profiles

To configure PTP profile in web interface, navigate to Configuration - PTP and click on 'Add New PTP Clock' button. After PTP clock configuration is displayed in table, select the profile in 'Profile' dropdown list. If no profile is selected, PTP clock of default 1588 type is used. Once the profile is selected, click on the 'Save' button to create appropriate clock configuration on the device.

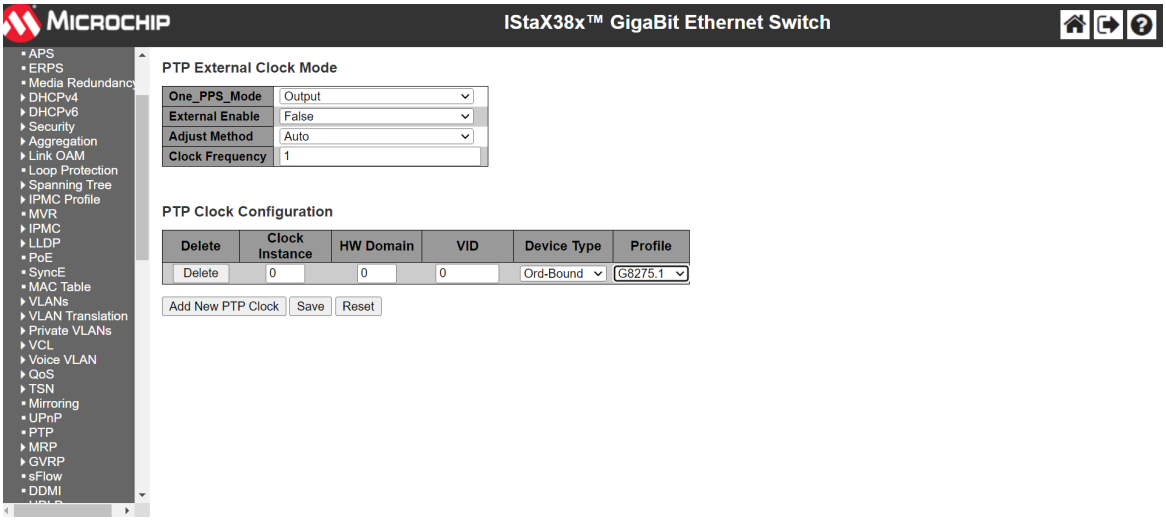

Figure 1. PTP Clock addition

Once the PTP clock is created, click on the link under 'Clock Instance' column to move to PTP clock port selection page. Under 'Port Enable and Configuration' table, select the checkbox for the port on which PTP needs to be enabled and click on the 'Save' button at the bottom of the page to enable PTP on specific port of device at the backend.

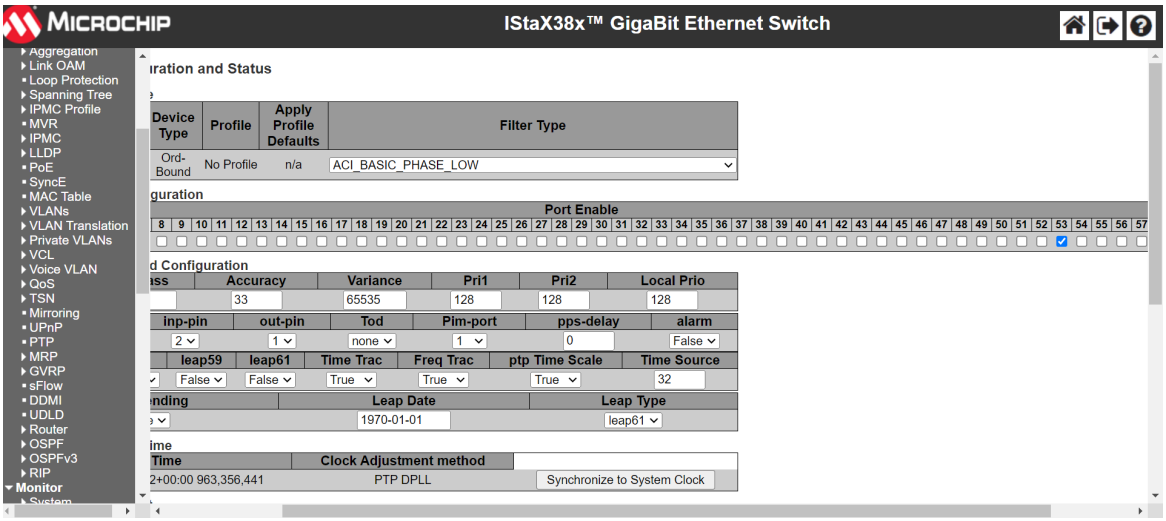

Figure 2. PTP clock port selection

The equivalent CLI commands for PTP profile configuration are described in following sections.

2.2. IEEE 1588

To configure IEEE 1588 profile on 10GigabitEthernet 1/10 port, below example configuration can be used. This profile can run on all boards with or without dpll on board.

Example 1. IEEE 1588 profile configuration

```
# config t
(config)# ptp 0 mode boundary profile ieee1588
(config)# interface 10gig 1/10
(config-if)# ptp 0
(config-if)#
```

On ports having phy timestamping, it is needed to provide 1PPS signal to the phy from the Mac to have a synchronised LTC running in both Mac and PHY. The following additional CLI command needs to be given to enable phy timestamping. This CLI command is applicable to all profiles. It is also possible to specify the clock domain from which 1pps signal needs to be generated.

Example 2. 1pps signal generation

```
(config)# ptp ext output auto
(config)#
(config)# ptp ext output auto clk-domain 1
```

As soon as PTP is enabled on both the devices, announce messages with corresponding clock properties are transmitted between the two devices. In each device, BMCA algorithm compares clock properties of both the devices to decide which device can act as server and client. If the user needs to explicitly make one device as server, then it is needed to configure priority1 field to lesser value than other device using following CLI command.

Example 3. configure priority1 to make server explicitly

```
(config)# ptp 0 priority1 10
```

Once the negotiation of server and client is completed, PTP status would reflect the PTP state of server and client.

Example 4. PTP state on server device

```
# show ptp 0 port-state interface 10GigabitEthernet 1/1
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper
Peer-delay
---- ------- --------- -------- ---- ---------- --------- ---------------- ----------
53 TRUE mstr FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
58 FALSE dsbl 99999
#
```

Example 5. PTP state on client device

```
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999
```

Once the client starts synchronising to the master device, how far the clock of client device is synchronised can be seen using following CLI commands.

Example 6. PTP clock offset on client device displayed with pico seconds resolution

```
# show ptp 0 current
stpRm OffsetFromMaster MeanPathDelay
----- ----------------- -----------------
1 -0.000,000,000,386 0.000,000,002,072
```

Example 7. PTP clock synchronisation status

```
# show ptp 0 slave
Slave port Slave state Holdover(ppb)
---------- ------------- -------------
10 PHASE_LOCKED 290.4
```

2.3. 8275.1

8275.1 profile can run in two modes i.e. packet and hybrid mode. In packet mode, frequency is derived from PTP packets alone. In hybrid mode, synce frequency is used along with timestamps from PTP packets to derive final frequency adjustment. Depending on availability of stable synce input, ptp application would switch to hybrid mode oblivious to user. Dpll is required on board to support hybrid mode. The following example shows configuration of 8275.1 profile.

Example 8. 8275.1 profile configuration

```
# config t
(config)# ptp 0 mode boundary profile g8275.1
(config)# interface 10GigabitEthernet 1/10
(config-if)# ptp 0
```

To make one device as server explicitly in 8275.1 profile, priority2 field needs to be configured such that server device has lesser value for this field compared to client device.

Example 9. configure priority2 field to make server explicitly.

```
(config)# ptp 0 priority2 10
```

PTP status can be verified using CLI commands for obtaining state and clock offset.

Example 10. PTP state on client device

```
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper
Peer-delay
---- ------- --------- -------- ---- ---------- --------- ---------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999
```

Example 11. PTP clock offset on client device

```
# show ptp 0 current
stpRm OffsetFromMaster MeanPathDelay
----- ------------------ ------------------
1 0.000,000,000,568 0.000,000,002,130
```

2.3.1. synce

Synce needs to be configured to switch to hybrid mode in 8275.1 profile. The following example CLI commands show synce configuration on port 10GigabitEthernet 1/10.

Example 12. synce configuration

```
(config)# network-clock clk-source 1 nominate interface 10gig 1/10
(config)# interface 10gig 1/10
(config-if)# network-clock synchronization ssm
```

To make one device as server and other device as client, clock quality level needs to be explicitly configured on server device using following CLI command.

Example 13. synce quality level configuration on server

```
(config)# network-clock ssm-freerun prc
(config)# network-clock ssm-holdover prc
```

Example 14. synce status on client device

```
# show network-clock

Selector State is: Acquire Lock

Alarm State is:
Clk: 1 2 3
LOCS: FALSE TRUE TRUE
SSM: FALSE FALSE FALSE
WTR: FALSE FALSE FALSE

LOL: FALSE
DHOLD: TRUE

SSM State is:
Interface Tx SSM Rx SSM Mode
10GigabitEthernet 1/10 QL_DNU QL_PRC Slave
```

Once the synce status is shown as locked, PTP application would wait for sometime and moves to hybrid mode where synce would be assisting in determing frequency of PTP client device. Some of the debug logs from servo show PTP moving to hybrid mode.

Example 15. Hybrid mode debug logs from servo

```
_APR_ZGN ASSM: mode switch complete on active server 0, to mode 1
```

The following CLI commands show hybrid or packet mode of PTP instance '0'.

Example 16. Servo log showing hybrid mode

```
# show ptp servo mode-ref
Servo [0] mode HYBRID ref 0
Servo [1] mode NONE ref -1
Servo [2] mode NONE ref -1
Servo [3] mode NONE ref -1
#
# show ptp ms-pdv apr cgu 0
# zl303xx_GetAprDeviceConfigInfo
Device 0x423c258 configuration:
The reference clock mode is 1
# zl303xx_GetAprServerConfigInfo
Device 0x423c258, Server clock 0 configuration data:
Timestamp is in format of 0
The server clock is in mode of 1 (1-hybrid, 0-packet)
Using BOTH (FORWARD & REVERSE) paths
The pull-in range is +/- 200.0ppm
The algorithm type mode is 6
The oscillator filter type is 0
The filter type is 13
```

```
The packet rate (fwd/rev): 16/16
- - - - - -
- - - - - -
```

Once Synce loses lock to reference, then again PTP application would switch back to packet mode.

2.4. 8275.2

8275.2 profile runs between two nodes situated far away using IP unicast encapsulation. After IP address configuration on client device, IP address of server device must be configured on client to initiate unicast connection between them.

Example 17. Example configuration on client device with IP address 1.1.1.2 to connect with server 1.1.1.1

```
(config)# vlan 2
(config-vlan)# exit
(config)# interface vlan 2
(config-if-vlan)# ip address 1.1.1.2 255.255.255.0
(config-if-vlan)# exit
(config)# interface 10gig 1/10
(config-if)# switchport access vlan 2
(config-if)# exit
(config)# ptp 0 mode boundary profile g8275.2 vid 2
(config)# interface 10gig 1/10
(config-if)# ptp 0
(config-if)# exit
(config)#
(config)# ptp 0 uni 0 1.1.1.1
(config)# end
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999
#
```

Similar to 8275.1 profile, priority2 field can be used as tie breaker to make one device server.

Example 18. configure priority2 field to make server explicitly.

```
(config)# ptp 0 priority2 10
```

2.5. 8265.1

Purpose of 8265.1 profile is to support synce operation between far away nodes using PTP. Similar to 8275.2 profile, IP unicast encapsulation is used for communication.

Example 19. example 8265 configuration and status

```
(config)# vlan 2
(config-vlan)# exit
(config)# interface vlan 2
(config-if-vlan)# ip address 1.1.1.2 255.255.255.0
(config-if-vlan)# exit
(config)# interface 10gig 1/10
(config-if)# switchport access vlan 2
(config-if)# exit
(config)# ptp 0 mode boundary profile g8265.1 ip4unicast vid 2
(config)#
(config)# interface 10gig 1/10
(config-if)# ptp 0
(config-if)# exit
(config)# do show ip in br
Interface Address Method Status
--------- ------------------ ------ ------
VLAN 2 1.1.1.2/24 Manual UP
(config)# ptp 0 uni 0 1.1.1.1
(config)# end
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999
#
```

Server device needs to transmit better synce quality level. This can be emulated by transmitting clock class corresponding to synce quality levels.

Example 20. Server transmitting clock class 84 equivalent to synce quality PRC

```
(config)# deb ptp 0 class 84
```

Synce on client device needs to lock to PTP. This can be done using below example configuration.

```
(config)# network-clock clk-source 1 nominate ptp 0
```

8265 profile uses the servo filter aci-freq-accuracy-fdd which would take around 4.5 hours for reaching frequency locked state. User need to wait for sufficient time to ensure proper output from PTP execution.

NOTE It must be noted that telecom profiles like 8275.x, 8265 are disabled on LAN966X platform from 2023.09 release.

2.6. 802.1as

802.1as profile is an industrial based PTP profile which operates between adjacent nodes. It uses two step ethernet encapsulation as part of PTP configuration.

Example 21. example 802.1as configuration

```
(config)# ptp 0 mode boundary profile 802.1as
(config)# interface 10gig 1/10
(config-if)# ptp 0
(config-if)# end
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE uncl FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999

802.1AS port status:
Port port-role is-mes-del as-cap rate-ratio cur-anv cur-syv sync-time-intrv cur-MPR AMTE comp-ratio comp-delay version minor-ver
---- --------- ---------- ------ ----------- ------- ------- ------------------- ------- ----- ---------- ---------- ---- --- ---------
10 Slave True True 641206 0 -3 0.375,000,000,000 0 False True True 2 1
#
```

Priority1 field can be used as tie breaker to make one device as server explicitly.

```
(config)# ptp 0 priority1 10
```

2.6.1. Multiple instances in different domains on same port

Instead of using chip’s hardware clock for phase or frequency adjustments, software clock can be used to run multiple instances simultaneously on the same port. The delay mechanism must be modified to use common mean link delay service to run multiple instances. Example configuration can be seen below.

Example 22. example multiple instance configuration in 802.1as profile.

```
(config)# ptp 0 mode boundary profile 802.1as clock-domain 3
(config)# ptp 0 domain 10
(config)# ptp 0 priority1 10
(config)# interface 25gig 1/1
(config-if)# ptp 0
(config-if)# ptp 0 delay-mecha common-p2p
(config-if)# end
#
# conf t
(config)# ptp 1 mode boundary profile 802.1as clock-domain 4
(config)# ptp 1 domain 20
(config)# interface 25gig 1/1
(config-if)# ptp 1
(config-if)# ptp 1 delay-mecha common-p2p
(config-if)# end
# show ptp 0 port-state interface 25GigabitEthernet 1/1
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
53 TRUE mstr FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
58 FALSE dsbl 99999

802.1AS port status:
Port port-role is-mes-del as-cap rate-ratio cur-anv cur-syv sync-time-intrv cur-MPR AMTE comp-ratio comp-delay version minor-ver
---- --------- ---------- ------ ----------- ------- ------- ------------------- ------- ----- ---------- ---------- ---- --- ---------
53 Master True True 602927 0 -3 0.000,000,000,000 0 False False False 2 1
#
#
# show ptp 1 port-state interface 25GigabitEthernet 1/1
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
53 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
58 FALSE dsbl 99999

802.1AS port status:
Port port-role is-mes-del as-cap rate-ratio cur-anv cur-syv sync-time-intrv cur-MPR AMTE comp-ratio comp-delay version minor-ver
---- --------- ---------- ------ ----------- ------- ------- ------------------- ------- ----- ---------- ---------- ---- --- ---------
53 Slave True True 602271 0 -3 0.375,000,000,000 0 False False False 2 1
#
```

Only one chip clock domain can be enabled on a port. So, when chip clock domain is used only one PTP instance can be run on the port. With software clock domains, there is no restriction on number of instances that can be run on the port.

NOTE

Multiple instances can be run simultaneously using only software clock domains for 802.1as profile. When multiple instances are allowed to modify hardware clock simultaneously, then the behavior will be unpredictable.

2.7. AED 802.1as

AED 802.1as profile is an automotive based PTP profile similar to the 802.1AS profile, the main difference is that the AED 802.1as profile does not use BMCA and therefore ptp ports are configured manually.

There should always be one grandmaster in the system and it can be configured either on the web (using the AED-GM device type) or via CLI commands as seen below.

Example 23. example AED 802.1as grandmaster configuration

```
(config)# ptp 0 mode aedgm profile 802.1as-aed
(config)# interface gig 1/2
(config-if)# ptp 0
(config-if)# end

# show ptp 0 port-state interface gig 1/2
```

Port	Enabled	PTP-State	Internal	Link	Port-Timer	Vlan-forw	Phy-timestamper	Peer-delay
2	TRUE	mstr	FALSE	Up	In Sync	Forward	FALSE	OK

Port	aed-port-role	is-mes-del	as-cap	rate-ratio	cur-syv	sync-time-intrv	cur-MPR	comp-ratio	comp-delay	versio
2	master	True	True	1640	-3	0.000,000,000,000	0	True	True	2

The remaining PTP instances in the system should be configured as bridge/end-station with the ports manually configured to either master or slave.

Specifically the aed-port-role field can be used to determine the role of the given port.

```
(config)# ptp 0 aed-port-role slave
```

Example 24. example AED 802.1as bridge/end-station configuration

```
(config)# ptp 0 mode boundary profile 802.1as-aed
(config)# interface gig 1/2
(config-if)# ptp 0
(config-if)# ptp 0 aed-port-role slave
(config-if)# end

# show ptp 0 port-state interface gig 1/2
```

Port	Enabled	PTP-State	Internal	Link	Port-Timer	Vlan-forw	Phy-timestamper	Peer-delay
2	TRUE	slve	FALSE	Up	In Sync	Forward	FALSE	OK

Port	aed-port-role	is-mes-del	as-cap	rate-ratio	cur-syv	sync-time-intrv	cur-MPR	comp-ratio	comp-delay	versio
2	slave	True	True	728745	-3	0.000,000,000,000	0	True	True	2

The user is also able to set the initial and operational intervals for both Pdelay_req and Sync packets.

The initialLogPdelayReqInterval is the startup interval which is fixed to 0 (1 packet/s). Once the peer delay communication has stabilized the device will move to the operLogPdelayReqInterval, which can be in the range of 0 (1 packet/s) and 3 (1 packet/8s).

The operLogPdelayReqInterval can be configured using the following command

Example 25. example AED 802.1as operLogPdelayReqInterval

```
(config-if)# ptp 0 aed-intervals operPdelayReq 3
```

The initialLogSyncInterval is the startup interval which can be in the range -5 (32 packets/s) to -3 (8 packets/s). Once the slave port has synchronized with the master, the slave port will request the master to change the syncInterval to operLogSyncInterval which can be in the range of -3 (8 packets/s) to 0 (1 packet/s). This request is done via the messageIntervalRequest TLV.

The initialLogSyncInterval and operLogSyncIntervals can be configured using the commands

Example 26. example AED 802.1as initialLogSyncInterval and operLogSyncInterval

```
(config-if)# ptp 0 aed-intervals initSync -3
(config-if)# ptp 0 aed-intervals operSync 0
```

3. PTP protocol data configuration

3.1. Clock modes

Istax application supports boundary clocks and transparent clocks in both one step and two step modes according to IEEE 1588 standard. Istax application takes care of profile specific requirements like supporting only boundary clock mode in 802.1as profile and one step clock for 8275, 8265 profiles.

3.1.1. Transparent clock

Transparent clocks update the residence time of a packet in correction field. In one-step transparent clock, PTP message received on a port is forwarded to egress port and then the residence time calculated by subtracting ingress time from egress time is updated in correction field of the message in egress port by the chip or Asic. There is no software processing involved in one step transparent clock. TC-mode configuration on the device decides how the ingress and egress timestamps are recorded in the packet.

In two-step transparent clock mode, PTP messages are forwarded to PTP application which would compute residence time in software according to IEEE 1588 standard before forwarding them on appropriate port. PTP application can store up to 25 outstanding entries of sync or delay request messages for two-step transparent clock before getting follow-up or delay response messages respectively. There are high chances of overunning buffer of delay request message if there are no quick delay response messages from the server side. To avoid overrunning buffers, it is recommended to change or remove the client configurations first on system having 2-step transparent clock. So, it is always recommended to use one-step transparent clock instead of two-step transparent clock to use hardware forwarding.

In both e2etransparent(end-to-end) or p2ptransparent clock(peer-to-peer), residence time is updated in correction field of outgoing message. But, in p2p transparent clock, peer delay is calculated on the ingress port and it is added to correction field of the outgoing message.

Example 27. example one step e2etransparent clock on ports 10gig 1/10, 10gig 1/12

```
(config)# ptp 0 mode e2etransparent onestep
(config)# interface 10GigabitEthernet 1/10,12
(config-if)# ptp 0
(config-if)#
```

Example 28. example one step p2ptransparent clock on ports 10gig 1/10, 10gig 1/2

```
(config)# ptp 0 mode p2ptransparent
(config)# interface 10gig 1/10,2
(config-if)# ptp 0
(config-if)# end
#
```

When p2p transparent clock is enabled on a port, peer delay is calculated on the port. The same peer delay would be added to the correction field of the incoming PTP message.

Example 29. example port-ds showing peer delay calculated on port 10gig 1/10

```
# show ptp 0 port-ds interface 10gig 1/10
Port Enabled Stat MDR PeerMeanPathDel Anv ATo Syv SyvErr Delm MPR DelayAsymmetry IngressLatency EgressLatency Ver Lpri NoSlv McAdr Two-step NoMstr
---- ------- ---- --- ------------------- --- --- --- ------ ---- --- ------------------- ------------------- ------- ------------ --- ---- ----- ----- -------- ------
10 True p2pt 0 0.000,000,002,291 1 3 0 No p2p 0 0.000,000,000,000 0.000,000,000,000 0.000,000,000,000 2 128 False deflt Clk Def. False
```

3.1.2. Boundary clock

Boundary clocks or masterOnly clock or slaveOnly clock negotiate with another clock to form client server role in synchronisation. After exchanging required parameters through Announce messages of PTP protocol, BMCA algorithm is calculated locally on each device to find the client and server roles in two communicating devices. The client device would synchronise its local time to server based on the timestamps received in PTP event messages.

Example 30. example boundary clock in one step mode

```
(config)# ptp 0 mode boundary onestep
(config)# interface 10GigabitEthernet 1/10
(config-if)# ptp 0
(config-if)# end
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 FALSE dsbl 99999
#
```

Example 31. example slaveOnly clock in two step mode.

```
(config)# ptp 0 mode slave twostep
(config)# interface 10GigabitEthernet 1/10
(config-if)# ptp 0
(config-if)# end
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper
Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
---------- ------- --------- ------
22 FALSE dsbl 99999
#
```

Example 32. example mode configuration for masterOnly clock.

```
(config)# ptp 0 mode master
```

3.1.3. Internal clock

In 2024.06 IStaX application release, a new internal clock mode is added to synchronize clock domains internal to the same board. The primary purpose of internal clock mode is to synchronize a software clock domain to a chip clock domain. When software clock domain is mapped to hardware clock domain, 1pps signal of that hardware clock domain can be monitored for any measurements.

Example 33. example internal clock synchronizing clock domain 1 to software clock domain 3. 1pps generation from clock domain 1.

```
(config)# ptp 1 mode internal clock-domain 1
(config)# ptp 1 internal src-clk-domain 3
(config)#
(config)# ptp ext outp clk-domain 1
```

3.2. Encapsulations

Istax application supports ethernet, IPv4 and IPv6 encapsulations. Ethernet and IPv4 encapsulations are supported for all clock modes. Currently, IPv6 encapsulation is supported for only one step e2etransparent clock.

In ethernet encapsulation messages, PTP header is embedded in payload of ethernet message. In IPv4 encapsulation, PTP header is encapsulated in payload of UDP message which in turn is embedded in IPv4 encapsulation. Similarly in IPv6 encapsulation, PTP header is embedded in UDP message running on IPv6 encapsulation. By default, if no encapsulation is mentioned in mode configuration CLI, then ethernet encapsulation is configured.

Example 34. example mode configuration using ipv4 encapsulation

```
(config)# ptp 0 mode boundary ip4multi
```

Example 35. example mode configuration using ipv6 encapsulation

```
(config)# ptp 0 mode e2etransparent ip6mixed
```

Example 36. example mode configuration using ethernet encapsulation

```
(config)# ptp 0 mode boundary ethernet
```

3.3. AFI or Automatic frame injection

In Istax application, Automatic frame injection(AFI) is used by PTP application to send Announce and Sync messages from the Server device using one step boundary or masterOnly clock modes. It is supported currently on Sparx-V and Jaguar-2 platforms.

If AFI is supported on the platform, then by default, AFI is used by application when the device is acting as server. On these platforms, PTP application would configure required parameters in Mac chip such as packet rate, message contents etc and then the Mac would take care of sending messages at according to packet rate. On unsupported platforms, software would take care of sending messages at required packet rate.

Example 37. example to disable AFI on Sparx-V platform.

```
(config)# no ptp 0 afi-sync
(config)# no ptp 0 afi-announce
```

The status of AFI usage can be known from running config. On Sparx-V or Jaguar-2 platform, disabling AFI will be shown in running config. On other platforms, there is no change in behaviour due to these CLI commands.

3.4. BMCA parameter configuration

Istax application allows configuration of various parameters like priority1, priority2, id etc that take part in BMC algorithm. Based on the output of BMC algorithm, client server roles between two devices is decided.

3.4.1. Web configuration of BMCA parameters

In PTP Clock Config web page, 'Clock Default DataSet' table contains various BMCA parameters available for configuration.

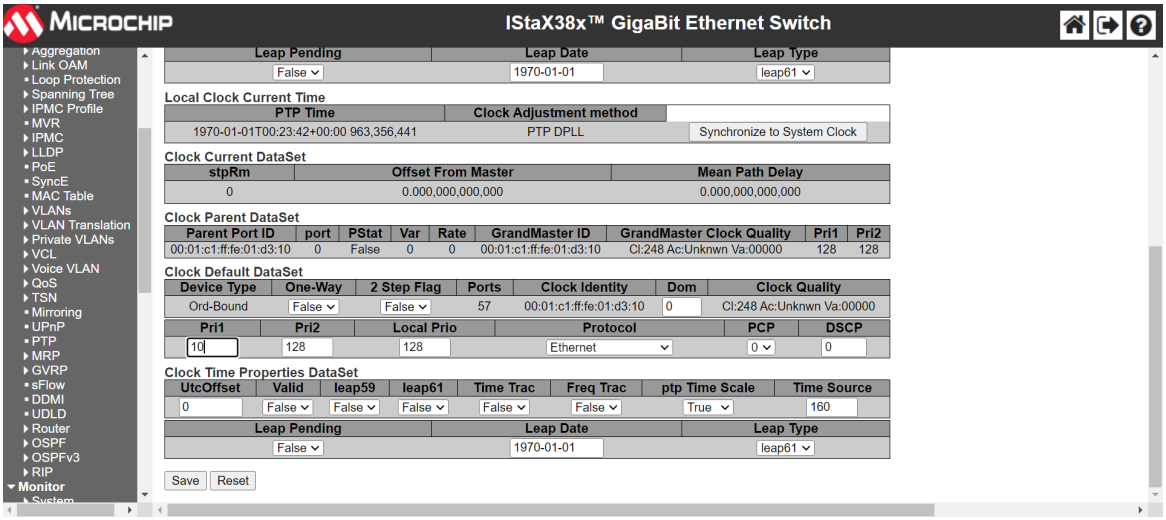

Figure 3. PTP Clock Config

The equivalent CLI commands are described in following sections.

3.4.2. priority1 and priority2

Priority1 or priority2 fields of the clock can be modified using below CLI commands. Lesser numerical value implies greater priority for this clock.

Example 38. example showing priority1 and priority2 configuration

```
(config)# ptp 0 priority1 20
(config)# ptp 0 priority2 10
```

3.4.3. Clock class, accuracy and variance

In Istax application, clock quality fields such as clock class, accuracy and variance are set to default values by PTP application during time of initialisation. These values are set based on Istax application properties. However, debug commands are available to modify these values. Lesser numerical values have higher priority.

Example 39. example showing configuration of clock class, accuracy and variance

```
(config)# debug ptp 0 class 30
(config)# debug ptp 0 accuracy 100
(config)# debug ptp 0 variance 1000
```

Debug commands cannot be saved on flash as part of start-up config and hence need to be configured explicitly after rebooting the device.

3.4.4. clock identity

Clock identity can be configured at the time of mode configuration. If no clock identity is provided, default value based on mac address is used by the application.

Example 40. example clock identity configuration

```
(config)# ptp 0 mode boundary id 11:11:11:11:11:11:11:11
```

3.4.5. Local priority

Local priority field is used in 8275 profile's BMC algorithm.

Example 41. example local priority configuration

```
(config)# ptp 0 localpriority 10
```

3.4.6. Domain

Domain number is exchanged in Announce messages such that client or server roles are formed only with the neighbors having same domain configuration. Each PTP profile sets its own domain number for communication. User can also configure domain number using following CLI command.

Example 42. example domain number configuration

```
(config)# ptp 0 domain 30
```

3.5. time properties configuration

Time properties data structures are used to identify the time scale and other characteristics of clock time. They can be configured using following CLI commands.

Example 43. example time properties DS parameters

```
(config)# ptp 0 time-property ?
freq-traceable frequency is traceable
leap-59 leap59 in current day
leap-61 leap61 in current day
leap-pending command includes a pending leap event
ptptimescale timing is a PTP time scale
time-source set time source
time-traceable timing is traceable
utc-offset set UTC offset
valid UTC offset is valid
<cr>
(config)# ptp 0 time-property utc-offset ?
←32768-32767> UTC offset value
(config)# ptp 0 time-property utc-offset 100
```

TIP 'Clock Time Properties DataSet' table in PTP Clock Config web page contains time properties for configuration.

3.6. Servo configuration

3.6.1. Filter type

Istax application supports various Zarlink servos such as aci-bc-full-on-path-phase, aci-basic-phase-low, aci-bc-partial-on-path-phase, aci-freq-accuracy-fdd etc. Istax also supports basic PID servo also known as basic servo. When a profile is selected, suitable servo for that profile is automatically linked. Currently, all PTP configurations which require Dpll to adjust frequency need to use zarlink servos. It is possible to switch between different servo filters using following CLI command.

Example 44. example servo filter change configuration

```
(config)# ptp 0 filter-type basic
```

TIP 'Clock Type and Profile' table in PTP Clock Config web page has 'Filter type' drop down box to select various filter types.

3.6.2. stable adjustment threshold, holdover stabilisation period of basic servo

While using basic servo, once the clock state of phase locked state is reached, then the frequency adjustment by servo is calculated and then it is compared with average adjustment obtained till that point of time. If this frequency adjustment is within limits for number of iterations equal to holdover stabilisation period, then valid holdover is considered to be obtained. Frequency adjustment by servo is calculated for every sync message after reaching phase locked state. Valid holdover is obtained after passing the limits of stable adjustment threshold for holdover stabilization period.

Example 45. example configuration for stable adjustment threshold, holdover stabilisation period.

```
(config)# ptp 0 ho adj-threshold ?
<1-3000> [1..3000] max frequency adjustment change within the holdover
stabilization period (in units of 0.1 ppb)
(config)# ptp 0 ho adj-threshold 1 filter ?
<10-86400> [10..86400] Holdover filter and stabilization period
(config)# ptp 0 ho adj-threshold 1 filter 1000
```

3.6.3. Basic servo filter constants

Basic servo follows P.I.D controller model in deriving frequency adjustment from current offset. The constants to be used for each controller i.e. 'P' or 'I' or 'D' or gain can be configured using following CLI. These constants are applied in the denominator parts of P.I.D controllers.

frequency adjustment = (prop + integrate + diff) * gain .
prop = offset / ap .
integrate = integral(offset) / ai .
diff = differential(offset) / ad .

Example 46. example basic servo filter constants

```
(config)# ptp 0 servo ap 2
(config)# ptp 0 servo ai 30
(config)# ptp 0 servo ad 30
(config)# ptp 0 servo gain 2
```

NOTE In PTP Clock Config web page, the tables 'Basic Filter Parameters' and 'Basic Servo Parameters' can be configured only when 'BASIC' filter type is selected.

To display current calculated state of P.I.D controller, the following CLI command can be used.

Example 47. example CLI command to display P.I.D states

```
(config)# ptp 0 servo displaystates (config)# 0.000000005122, 0.000000013752,
9443401, 300428, 9142837, 136, 13, 13, 13, 1992903
0.000000005042, 0.000000012652, 9439585, 276394, 9163492, -301, 12, 12,
12, 1992894
0.000000004993, 0.000000011209, 9426275, 244878, 9181792, -395, 11, 11,
11, 1992903
0.000000004921, 0.000000010098, 9418575, 220601, 9198278, -304, 10, 10,
10, 1992903
0.000000004854, 0.000000008672, 9401500, 189454, 9212436, -390, 8, 8, 8,
1992894
0.000000004780, 0.000000007813, 9395638, 170682, 9225191, -235, 7, 7, 7,
1992903
(config)#no ptp 0 servo displaystates
```

3.7. IP Unicast Client configuration

Currently, IPv4 protocol is supported for PTP unicast communication on Istax software. To initiate unicast connection to the Server device, IP address of the Server device needs to be configured on the client device. Each IP unicast client can be configured to initiate communication with up to 5 servers indexed from 0 to 4. If no communication duration is specified, 100 seconds is the duration for which the client tries to connect to server.

Example 48. example ip unicast client 0 connection to server 1.1.1.1

```
(config)# ptp 0 uni 0 1.1.1.1
(config)# ptp 0 uni 0 duration 1000
```

TIP In 'Clock Default DataSet' table of PTP Clock Config web page, 'IPv4Uni' must be selected from drop down box of 'Protocol' field to configure Unicast reference address in 'Unicast Slave Configuration' table.

3.8. Logging timestamps

Timestamps in both directions can be logged on to the console for debugging using logging CLI commands on client device. In forward direction, t1 followed by t2 timestamps are displayed. In backward direction, t4 followed by t3 timestamps are displayed. The timestamps t1,t2,t3,t4 must be correlated to sync and delay request message timestamps of PTP protocol.

Example 49. example timestamp logging in both directions

```
(config)# ptp 0 log 4
(config)# MasterUUID 00:01:C1:FF:FE:xx:xx:xx
#MasterIP n.a.
#ProbeUUID 00:01:c1:FF:FE:xx:xx:xx
#ProbeIP n.a.
#Title: Microchip Test Probe/1588 Timestamp Data/Transmit and receive Timestamp
F,45248,0001150089,416686001,0001150088,938276286,+0000000308
B,00039,0001150090,018557986,0001150089,540034217,+0000113565
F,45249,0001150090,413128939,0001150089,934718928,+0000000302
B,00040,0001150090,579500211,0001150090,100976674,+0000113170
(config) no ptp 0 log
```

3.9. Generic instance specific configurations

3.9.1. Path trace TLV

To append path trace TLV to announce message, the following CLI command need to be used.

Example 50. example CLI to append path trace TLV.

```
(config)# ptp 0 path-trace-enable
```

3.9.2. clock-domain

Clock domain is the index of local time counter maintained either in hardware or software. Some platforms like Sparx-V, Jaguar-2 maintain up to 3 clock domains in switch chip i.e. 3 different LTC counters in chip. Some platforms like Ocelot, Maserati maintain only 1 clock domain in chip i.e. only 1 LTC counter. It must be noted that DPLL and PHY operate in only clock domain 0.

Example 51. example clock domain configuration

```
(config)# ptp 0 mode boundary clock-domain 1
```

3.9.3. vlan

If Vlan id is enabled on a port, PTP mode configuration must include corresponding vlan id.

```
(config)# ptp 0 mode boundary vid 2
```

3.10. Software Clock

Software Clock is the one where the frequency or phase adjustments are done in software maintained data structure instead of actual chip clock. Software clock uses underlying chip clock domain 0 for timestamping. Apart from chip clock domains, there are 4 software clock domains possible in istax software. For example on Sparx-V platform, apart from 3 chip clock domains, 4 more software clock domains are possible. So, the software clock domain number starts from 3 on this platform and maximum clock domain that can be configured is 6 on this platform.

Software clock domains can be used only with 802.1as profile. It must be noted that when adjustments are done in actual chip clock, 1pps output from clock domain 0 will be in sync with phase and frequency adjustments. But in software clock, 1pps output from the chip clock domain 0 will not be in sync with software clock time.

Example 52. example software clock domain configuration

```
(config)# ptp 0 mode boundary profile 802.1as clock-domain 3
```

3.11. Virtual port

After configuring PTP profile, virtual port needs to be configured with appropriate source of 1-pps signal and TOD message. 1-PPS signal can be transmitted through RS-422 interface or through SMA cable between Input/Output pins. TOD can be transmitted through RS-422 serial ports or Ethernet interface. Virtual port is currently supported on profiles such as IEEE 1588, G8275.1, G8275.2 and 802.1as. Some of the boards like the caracal platform types do not support virtual port due to lack of PPS signal connectors on the boards.

TIP In PTP Clock Config web page, virtual port parameters can be configured in 'Virtual Port Enable and Configuration' table.

3.11.1. 1-PPS through RS-422 interface, TOD through RS-422 interface

In this type of configuration, RJ-45 Ethernet cable connected between Reference board and client board is sufficient to transfer both 1-PPS signal and TOD information.

Example 53. example client board config

```
# conf t
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode sub
(config)# ptp 0 virtual-port tod ser proto zda
(config)#
```

Example 54. example server board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode main-man
(config)# ptp 0 virtual-port tod ser proto zda
```

3.11.1.1. Auto mode on server

Instead of using main-man mode, main-auto mode can be configured on server. But, this mode works only with serial protocol 'polyt' format. In this mode, the 1 pps signal sent to client device is reflected back to server device and the timestamp of when it is received on server device is noted and sent as correction field adjustment in the TOD message. Since, polyt serial format allows nano second transmission in TOD message, it is used for the auto mode TOD messages. Average one way delay of the 1pps signal is calculated more accurately using Auto mode.

example main-auto mode configuration on server device

```
(config)# ptp 0 virtual-port mode main-auto
(config)# ptp 0 virtual-port tod ser proto polyt
```

3.11.2. 1-PPS through RS-422 interface, TOD through Ethernet interface

In this type of configuration, 1-PPS signal is transmitted through RS-422 interface and TOD is sent through Ethernet interface. So, RJ-45 cable is required between RS-422 serial ports and another Ethernet cable is required between Ethernet ports on the board.

Example 55. example client board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode sub
(config)# ptp 0 virtual-port tod pim interface 10GigabitEthernet 1/2
```

Example 56. example server board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode main-man
(config)# ptp 0 virtual-port tod pim interface 10GigabitEthernet 1/1
```

3.11.3. 1-PPS through SMA connectors, TOD through RS-422 interface

In this type of configuration, 1-PPS signal is transmitted through SMA connectors or IO pins on the board. TOD is transmitted through RS-422 interface. So, SMA cable needs to be connected from output pin of Reference board to input pin of Client board. RJ-45 cable needs to be connected between RS-422 Timing out serial port on Reference board to Timing in serial port on Client board.

Example 57. example client board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode pps-in 2
(config)# ptp 0 virtual-port tod ser proto zda
```

Example 58. example server board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode pps-out 1
(config)# ptp 0 virtual-port tod ser proto zda
```

3.11.4. 1-PPS through SMA connectors, TOD through Ethernet interface

In this type of configuration, 1-PPS signal is transmitted through SMA connectors or IO pins on the board. TOD is transmitted through Ethernet interface. So, SMA cable needs to be connected from output pin of Reference board to input pin of Client board. Ethernet port connection between boards is same as the one used for running PTP protocol.

Example 59. example client board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode pps-in 2
(config)# ptp 0 virtual-port tod pim interface 10GigabitEthernet 1/2
```

Example 60. example server board config

```
(config)# ptp 0 mode boundary profile g8275.1
(config)# ptp ext outp auto
(config)# ptp 0 virtual-port mode pps-out 1
(config)# ptp 0 virtual-port tod pim interface 10GigabitEthernet 1/1
```

3.11.5. virtual port status

Virtual port state is appended to PTP status of any other physical port.

```
# show ptp 0 port-state interface 10GigabitEthernet 1/1
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
1 FALSE dsbl FALSE Down In Sync Discard FALSE OK
VirtualPort Enabled PTP-State Io-pin
----------- ------- --------- ------
22 TRUE slve 2
```

3.11.6. Delay compensation on virtual port

The time taken for the 1pps signal to reach Sub device can be compensated using the below software command. This delay is provided to Servo as the peer delay on the virtual port path.

```
(config)# ptp 0 virtual-port mode sub pps-delay 70
```

To identify the compensation needed for virtual port, user can find identify the lag of 1pps signal on virtual port using trace logs. Initially, user can make sure that PTP port attains ‘slave’ state and virtual port is passive state by adjusting the clock class of virtual port greater than ptp port i.e. set the clock class of virtual port to greater than 248. Once the PTP port attains phase locked state and when the offset to master is almost less than 10 nano seconds, the below trace log can be enabled to see the offset of virtual port compared to ptp port.

Trace log which shows ‘t2’ timestamp of virtual port at offset of 79ns.

```
# deb trace module level ptp 1_pps deb
I ptp/1_pps 09:41:45 136/io_pin_pps_in_slave_handler#8909: inst: 0, devicetype 1, protocol 0
D ptp/1_pps 09:41:45 136/io_pin_pps_in_slave_handler#8910: io_pin 2, time 0 s_msb 1570 s 79 ns 39936 ps
D ptp/1_pps 09:41:45 136/ptp_external_input_slave_function#1014: DUMP: enable_t1[1] new_t1 [0], new_t2[0]
D ptp/1_pps 09:41:45 136/ptp_external_input_slave_function#1039: t2 0 s_msb 1570 s 79 ns 39936 ps
```

3.11.7. ‘not-master’ option configuration on PTP port for switchover

When virtual port attains ‘slave’ state, PTP Ethernet port would attain ‘master’ state with BMCA algorithm. But, for smooth transition during switchover, PTP Ethernet port should continuously receive timestamps. To enable PTP port receive timestamps continuously when it is not in ‘slave’ state, ‘not-master’ option should be enabled on PTP port immediately after configuring PTP on the port. With ‘not-master’ option enabled on the Ethernet port, PTP port moves to passive state when virtual port attains ‘slave’. In passive state, PTP Ethernet port continue to receive timestamps and update them to servo. When the servo continues to receive timestamps simultaneously on virtual port and ethernet port, switchover from virtual port to ptp Ethernet port will be smooth.

```
(config)# interface 25GigabitEthernet 1/1
(config-if)# ptp 0 not-master
(config-if)# end
# show ptp 0 port-state interface 25GigabitEthernet 1/1
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
53 TRUE pass FALSE Up In Sync Forward FALSE OK
VirtualPort Enabled PTP-State Io-pin
```

```
---------- ------- --------- ------
58 TRUE slve 2
#
```

3.11.8. RS-422 interface baudrate

Baudrate of RS-422 interface used by virtual port can be configured using below CLI command.

```
(config)# ptp rs422 baudrate 9600 flowctrl none parity none stopbits 1 wordlength 8
```

3.11.9. UTC offset on virtual port

UTC time received on virtual port can be adjusted to TAI timescale used in PTP using below mentioned virtual port's CLI for time properties.

```
(config)# ptp 0 virtual-port time-property utc-offset 50 valid
```

3.11.10. clock accuracy on virtual port

Grandmaster's accuracy on virtual port can be configured using below CLI command. By default clock accuracy of virtual port is set to 100ns(0x21).

```
(config)# ptp 0 virtual-port accuracy 120
```

3.11.11. clock class on virtual port

The Clock class of grandmaster to which virtual port is locked can be configured through below CLI command. By default clock class of virtual port is set '6'.

```
(config)# ptp 0 virtual-port class 150
```

3.11.12. Grandmaster's Clock Identity on virtual port

The Clock-Identity of grandmaster to which virtual port is locked can be configured through below CLI command.

```
(config)# ptp 0 virtual-port clock-identity 0a:0b:0c:00:01:02:03:04
```

3.11.13. local priority on virtual port

Local priority unique to every port of the device can be configured for virtual port also using below CLI command.

```
(config)# ptp 0 virtual-port local-priority 20
```

3.11.14. priority2 field on virtual port

Priority2 field of Grandmaster clock to which virtual port is locked can be configured using below CLI command.

```
(config)# ptp 0 virtual-port priority2 100
```

3.11.15. Steps Removed field on virtual port

Distance of grandmaster clock represented by ‘stepsRemoved’ field can be configured on virtual port using below CLI command.

```
(config)# ptp 0 virtual-port steps-removed 2
```

3.11.16. Clock variance on virtual port

Clock variance of Grandmaster Clock locked to virtual port can be configured using below CLI command.

```
(config)# ptp 0 virtual-port variance 100
```

3.11.17. Statistics of 1pps and TOD on virtual port Receiver

Statistics of the received 1pps signal and TOD count can be obtained by using below debug CLI command.

```
# deb ptp pps-tod statistics
one_tod_cnt : 250610
one_pps_cnt : 257921
missed_one_pps_cnt : 615
missed_tod_rx_cnt : 7914
```

3.11.18. Alarm Configuration on virtual port Sender

To emulate GNSS status message, Alarms can be transmitted from virtual port sender to receiver.

```
(config)# ptp 0 virtual-port alarm enable
```

3.11.19. Alarm Status on virtual port Receiver

On receiving alarm from sender, the flags ‘frequencyTraceable’, ‘timeTraceable’ in the time properties of virtual port on receiver are updated. By default, the accuracy of virtual port is set 100ns i.e. 0x21. When alarm is received, accuracy is set 254 i.e. considered as unknown value. But, the final time properties of PTP clock of receiver device depends on whether the PTP clock reached ‘phase locked’ state or holdover and if it is in holdover, whether the servo state is holdover in-spec or out of spec. Below log shows ‘TimeTrace’ and ‘FreqTrace’ fields of PTP clock as false after receiving alarm.

```
# show ptp 0 time-property
UtcOffset Valid leap59 leap61 TimeTrac FreqTrac ptpTimeScale TimeSource
--------- ----- ------ ------ -------- -------- ------------ ----------
0 False False False False False True 32
```

3.11.20. Current Configuration on virtual port

Current configuration of virtual port participating in the BMCA algorithm decision can be seen using below CLI command. When an alarm is received, virtual port’s current configuration is not changed but the PTP clock’s time properties and parent-ds data structures reflects alarm’s result in ‘accuracy’, ‘TimeTrace’ and ‘FreqTrace’ fields.

```
# show ptp 0 virtual-port
clockinst :0
class :6
accuracy :100 ns
variance :65535
localPriority :128
priority1 :128
priority2 :128
io-pin :128
enable :TRUE
steps-removed :0
clock-identity :00:01:c1:ff:fe:01:c7:74
```

4. PTP system specific configurations

4.1. Adjustment method

On boards having dpll, PTP application can adjust frequency through dpll or LTC. If Auto option is selected then application will decide the adjustment method based on requirements. By default, dpll is used for adjusting frequency. On boards not having dpll, LTC is the only source for adjusting frequency. If the user wants to change the default adjustment method, then it has to be done before configuring clock mode like boundary clock etc. It is recommended to use 'auto' adjustment method.

Example 61. example to change adjustment method to ltc or back to auto mode

```
(config)# ptp adj-method ltc
(config)#
(config)# ptp adj-method auto
(config)# ptp 0 mode boundary
```

It is possible to configure the adjustment method using different CLI commands shown below.

```
(config)# ptp ext ltc
```

```
(config)# ptp ext auto
```

In the running config, adjustment method can be found using CLI command starting with 'ptp ext'.

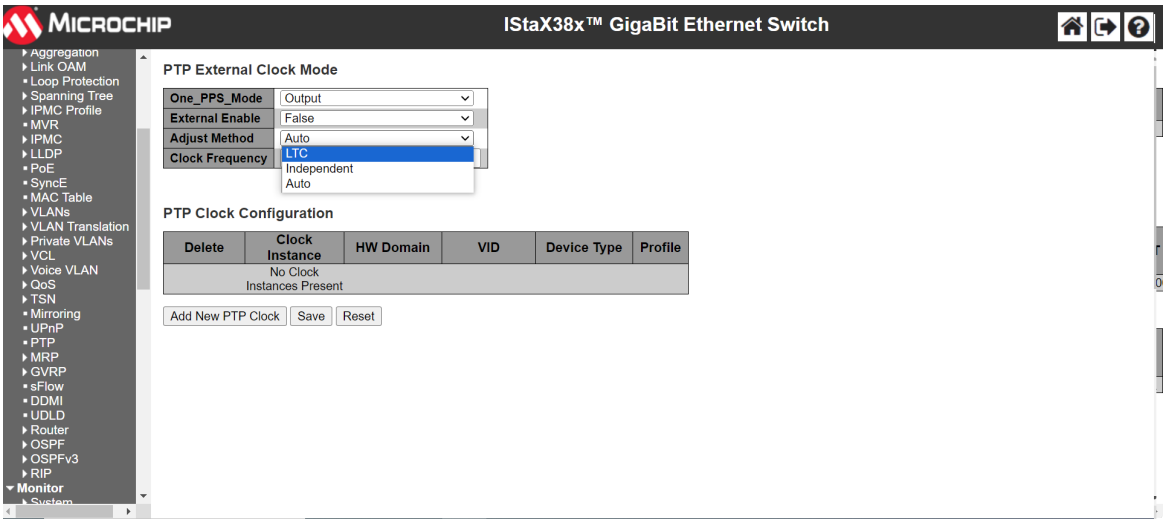

Figure 4. PTP Adjustment method configuration

4.2. 1pps signal output

1pps signal output from SMA connectors of boards running ISTAX software is enabled by default from '2023.06' release. To configure 1pps signal output generation explicitly, the CLI command described at 1pps signal generation can be used.

4.3. Synchronizing system time and ptp time

Instead of using server clock, PTP instance can synchronise its time to local system clock time. But, PTP mode must be configured as 'master' to use it because on boundary clocks or client clocks, PTP application locks to different server clocks.

Example 62. PTP instance 0 is locked to system clock time

```
(config)# ptp system-time get 0
```

Every second, time of clock domain 0 is configured in system clock. .system clock is synchronised to PTP local time of clock domain 0.

```
(config)# ptp system-time set
```

TIP In PTP Clock Config web page, under 'Local Clock Current Time' table, click on the 'Synchronize to System Clock' button to synchronize PTP time to system time.

4.4. TC internal mode

TC(transparent clock) internal mode determines how correction field update is calculated. The possible TC modes can be 30-bit, 32-bit, 44-bit, 48-bit. On all boards having PHY timestamping, 30-bit mode is supported and 32-bit mode is not supported. Support for other modes depends on the type of phy available on board. TC internal mode is used not only for 1-step TC for all messages but also used for updating correction in delay request message in 1-step BC.

In 30-bit mode, 30-bit nano second part of ingress time is placed in reserved bytes and on egress side, ingress time is derived from 30-bit nano seconds, subtracted from egress time and result is updated in correction field.
In 32-bit mode, along with 30-bit nano seconds part of ingres time, 2-bits of seconds part is appended to reserved bytes. On egress side, similar to 30-bit mode, ingress time is subtracted from egress time and result is updated in correction field.
In 44-bit mode, truncated 44-bit ingress time is subtracted from correction field on ingress side. On egress side, 44-bit time is added to correction to obtain the residence time.
In 48-bit mode, ingress time is converted to nano seconds format and subtracted from correction field on ingress side. On egress side, full 48-bit nano seconds part is added to correction field to calculate residence time.

In 1-step boundary clocks, delay request message requires correction field update. On all platforms using switch timestamping, LTC time is updated in origin timestamp of delay request message by application and Mac takes care of updating correction field by calculating the difference of LTC times. This is done using 30-bit TC mode. On platforms like Jaguar-2 also having both switch and PHY timestamping, application updates current LTC time in origin timestamp field. When the packet passes through Mac, Mac would insert 30-bit nano seconds in reserved field and PHY would update difference in correction field. On some platforms having Lan8814, 48-bit TC mode is used and application subtracts current time from correction field and PHY adds its local time to correction field to update the difference.

TC internal mode must be saved in config and board must be rebooted to apply the mode at booting time.

Example 63. TC internal mode

```
(config)# ptp tc-internal mode ?
<0-3> 0 = MODE_30BIT, 1 = MODE_32BIT, 2 = MODE_44BIT, 3 = MODE_48BIT
(config)# ptp tc-internal mode 3

Successfully set the TC internal mode...
Internal TC mode Configuration has been set, you need to reboot to activate the
changed conf.
(config)#
```

4.5. Switching between PHY and Mac timestamping

If the user wants to switch to Mac timestamping on the ports using PHY timestamping, then below CLI command must be applied and saved in startup config. In the next reboot, board would be booting with PHY timestamping disabled.

Example 64. Disabling PHY timestamping

```
(config)# ptp phy-ts dis
PHY timestamping mode disabled
(config)# end
# copy running-config startup-config
Building configuration...
% Saving 1299 bytes to flash:startup-config
# reload cold
```

It must be noted this command will disable PHY timestamping on all ports having such capability on the board. Similarly, PHY timestamping can be enabled back using 'no' form of same CLI command. Before applying this command, it must be ensured that there is no PTP configuration on the board. After saving the command in startup config, board must be rebooted.

5. Port specific configuration

5.1. Web configuration of PTP Clock Port specific config

In PTP Clock Config page, under 'Port Enable and Configuration' table, click on link 'Ports Configuration' at the far right end of the table to move to PTP Clock Ports Config web page for any PTP port specific configuration.

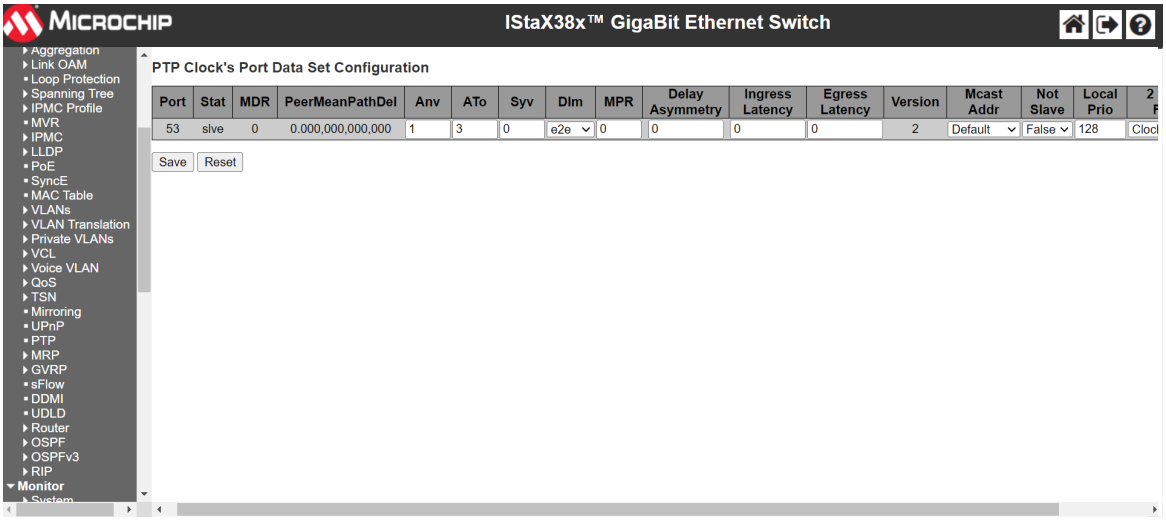

Figure 5. PTP Clock Ports Config

The equivalent CLI commands are described in following sections.

5.2. Latency

To have better path delays and accuracy compared to reference system, latencies can be configured for any port.

Example 65. example showing how to reduce path delay by 20ns

```
(config-if)# ptp 0 ingress-latency 20
(config-if)# ptp 0 egress-latency 20
```

Istax application software configures default latencies in the system. User can consider type of cables and port used to calculate expected path delay. For example, fibre cable of 1m contributes path delay of 4ns. Based on actual length of cable connected between the devices, expected path delay is calculated. Excess difference between expected path delay and actual path delay measured by PTP application must be compensated through latency adjustments. Istax application usually takes care of path delays based on port types. Otherwise, User must initially try to add or subtract difference of path delays on ingress and egress side such that final values of path delays are in expected range. User must adjust latencies slowly in multiple cycles by comparing difference between expected values and desired values in each cyle. After adjusting path delays to expected values and ensuring that offset of client to server is tending to '0', 1pps signal error can also be reduced by adjusting latencies equally in ingress and egress directions. While updating latencies for 1pps error, it must be ensured that path delays should not be changed. Therefore, when offset to server is tending to '0', 1pps signal error compared to reference system must be equally distributed in both directions to reduce the error near to '0'.

Example 66. example showing compensation of 1pps error of -20ns

```
(config-if)# ptp 0 ingress-latency 20
(config-if)# ptp 0 egress-latency -20
```

5.3. asymmetry

Asymmetry between egress and ingress directions can be configured for each port.

```
(config-if)# ptp 0 delay-asymmetry 100
```

5.4. two-step

Eventhough PTP mode is configured as one-step clock, two-step flag can be enabled for any specific port.

```
(config-if)# ptp 0 two-step true
```

5.5. delay mechanism

By default delay mechanism is enabled according to PTP clock mode. For example, end to end clock is configured for PTP instance, delay mechanism on port can be modified to peer-to-peer delay mechanism from end to end mechanism or viceversa.

```
(config-if)# ptp 0 delay-mechanism p2p
```

5.6. announce, sync, delay request or peer delay request packet intervals

Announce message interval or sync message interval can be configured for each port. Delay request or peer delay request message interval can be configured each port. On any port, it is possible to use either end to end delay mechanism or peer delay mechanism. So, same CLI command is used for configuring delay request interval or peer delay message interval. Irrespective of delay request message interval configuration, the actual rate of delay request message transmission also depends on server device’s delay request message interval configuration. Negative interval configuration indicates number of packets sent within a second. Positive interval configuration indicates number of seconds waited for sending one packet.

```
(config-if)# ptp 0 sync-interval -3
(config-if)# ptp 0 announce interval -3
(config-if)# ptp 0 delay-req interval -3
```

5.7. Master only configuration

If it is required not to make some port synchronise to any other reference or server, then that port can be configured with 'master-only' CLI command.

```
(config-if)# ptp 0 master-only
```

5.8. Announce receipt timeout

To maintain synchronisation status with reference, timeout for receipt of announce messages can be configured. Timeout value multiplied by announce message interval gives the time at which timeout happens.

```
(config-if)# ptp 0 announce timeout 3
```

5.9. local priority

Local priority used by 8275 BMCA algorithm can be overridden for every port.

```
(config-if)# ptp 0 localpriority 10
```

5.10. 802.1as Port Specific Configurations

5.10.1. Web configuration of 802.1as specific port config

If '802.1as' profile is selected, addition tables such as '802.1AS Port Data Set Configuration' are displayed in PTP Clock Ports Config web page for '802.1as' specific configuration.

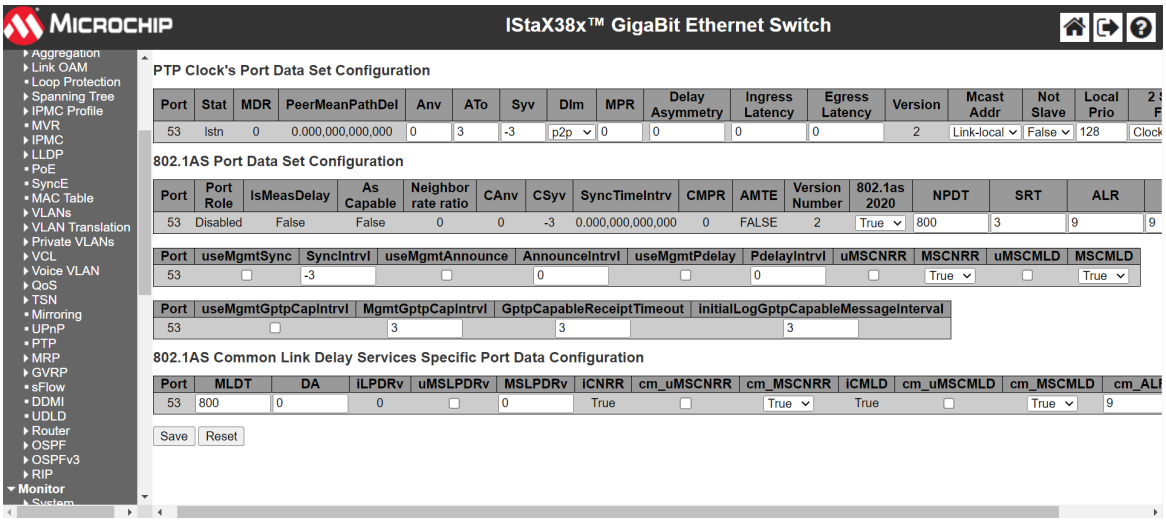

Figure 6. 802.1as config in PTP Clock Port Config page

Equivalent CLI commands for 802.1as profile specific port configurations are described in following sections.

5.10.2. 802.1as 2020

This allows the user to switch between the 802.1as 2020 and 802.1as 2011 version, which will update frame specific items such as minorVersionPTP and tlvType. Default value is 2020 which conforms with the 802.1as 2020 standard.

```
(config-if)# ptp 0 802.1as 2020
```

5.10.3. Common mean link delay service(CMLDS)

If multiple PTP instances need to be enabled between two adjacent devices on the same port, then common mean link delay service or CMLDS need to be enabled on that port. CMLDS must be used with software clocks. CMLDS calculates peer mean path delay, neighbor rate ratio and shares these values with all the instances using the CMLDS service regularly. Before enabling another PTP instance on port, it is mandatory to enable CMLDS on the port for earlier configured PTP instance.

Example 67. Enabling cmlds on port

```
(config)# interface 10gig 1/13
(config-if)# ptp 0 delay-mechanism common-p2p
```

5.10.4. Allowed lost responses

This is the number of Pdelay_Req messages without valid responses above which a port is considered to be not exchanging peer delay messages with its neighbor. Default value is 9 according to 802.1as 2020 standard.

```
(config-if)# ptp 0 allow-lost-resp 3
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS.

```
(config-if)# ptp cmlds allow-lost-resp 3
```

5.10.5. Allowed faults

This is the number of faults above which port is not capable of interoperating with its neighbor. Here faults refer to either invalid neighbor rate ratio or computed mean link delay greater than threshold. Default value of faults is 9 according to 802.1as 2020 standard.

```
(config-if)# ptp 0 allow-faults 9
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS.

```
(config-if)# ptp cmlds allow-faults 9
```

5.10.6. compute MeanLinkDelay

computeMeanLinkDelay variable determines whether to compute mean link delay using peer delay messages or not. Usually, this value is obtained from fields in signalling message from neighbor. But, the user can also configure this variable from CLI using below command. Final result depends on the value obtained from linkDelayInterval statemachine which obtaines value from neighbors.

```
(config-if)# ptp 0 compute-meanlinkdelay
```

If the user wants to compute meanLinkDelay irrespective of linkDelayInterval state machine, then the option 'force' must be given.

```
(config-if)# ptp 0 compute-meanlinkdelay force
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS as show below.

```
(config-if)# ptp cmlds compute-meanlinkdelay
```

5.10.7. mean link delay threshold

It is the threshold propagation time to neighbor above which the corresponding port cannot use 802.1as protocol. Default value is 800ns.

```
(config-if)# ptp 0 delay-thresh 1000
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS as show below.

```
(config-if)# ptp cmlds pdelay-thresh 1000
```

5.10.8. compute neighbor rate ratio

computeNeighborRateRatio variable determines whether to compute neighbor rate ratio or not. Usually, this value is obtained from signalling message from neighbor. But, the user can also configure this variable from CLI. Final result depends on signalling message fields. The user can also force this option to apply irrespective of neighbor’s signalling message.

```
(config-if)# ptp 0 compute-neighbor-rate-ratio ?
force force indicates to use management settable
compute-neighbor-rate-ratio
<cr>
(config-if)# ptp 0 compute-neighbor-rate-ratio force
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS as show below.

```
(config-if)# ptp cmlds compute-neighbor-rate-ratio
```

5.10.9. Announce interval configuration

Announce interval is derived from signalling message from neighbor. User can also force the announce message interval irrespective of neighbor’s configuration using the option 'usemgtSettableLogAnnounceInterval'. The CLI 'ptp 0 announce interval' is used for initial value of announce message interval. The default value is 0 (1 packet/sec) according to 802.1as 2020 standard.

```
(config-if)# ptp 0 mgtSettableLogAnnounceInterval -2
(config-if)# ptp 0 usemgtSettableLogAnnounceInterval 1
```

In order to instruct the neighbor to change its Announce interval via the message interval request TLV, the following command can be used

```
(config-if)# ptp 0 announce interval -2
```

It is also possible to use the special values 126 (default) and 127 (stop) (section 10.6.4.3.8 - 802.1as-2020 standard).

```
(config-if)# ptp 0 announce interval default
(config-if)# ptp 0 announce interval stop
```

5.10.10. gPTP capable message Interval

A signalling message containing gPTP capable TLV is exchanged with neighbor for every 'gPTP capable message interval'. The variable 'initialLogGptpCapableMessageInterval' from 802.1as standard can be set using following command. This is used during initialisation or when there is no valid settings from neighbor.

```
(config-if)# ptp 0 gptp-interval -2
```

The variable 'useMgtSettableLogGptpCapableMessageInterval' determines the source of the gPTP capable message interval. If the value is TRUE, the value of 'currentLogGptpCapableMessageInterval' is set equal to the value of 'mgtSettableLogGptpCapableMessageInterval'. If the value is FALSE, the value of 'currentLogGptpCapableMessageInterval' is determined by the GptpCapableMessageIntervalSetting state machine.

```
(config-if)# ptp 0 mgtSettableLogGptpCapableMessageInterval -1
(config-if)# ptp 0 useMgtSettableLogGptpCapableMessageInterval 1
```

The timeout for receiving gPTP capable messages can be configured in terms number of gPTP capable message intervals.

```
(config-if)# ptp 0 gptp-to 3
```

5.10.11. peer delay request interval

Peer delay message interval can be taken either from linkDelayIntervalSetting state machine or from user configuration. LinkDelayIntervalSetting state machine obtains values using signalling messages from neighbor. If 'useMgtSettableLogPdelayReqInterval' is true, then the value of 'currentLogPdelayReqInterval' is set equal to the value of 'mgtSettableLogPdelayReqInterval'. Otherwise, it is set by 'linkDelayIntervalSetting' state machine. The CLI command 'ptp 0 delay-req interval ..' sets the initial value.

```
(config-if)# ptp 0 mgtSettableLogPdelayReqInterval -3
(config-if)# ptp 0 usemgtSettableLogPdelayReqInterval 1
```

In order to instruct the neighbor to change its peer delay request interval via the message interval request TLV, the following command can be used.

```
(config-if)# ptp 0 delay-req interval -3
```

It is also possible to use the special values 126 (default) and 127 (stop) (section 10.6.4.3.6 - 802.1as-2020 standard).

```
(config-if)# ptp 0 delay-req interval default
(config-if)# ptp 0 delay-req interval stop
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then this configuration must be done for CMLDS as show below.

```
(config-if)# ptp cmlds pdelayreq-interval -3
```

5.10.12. Sync message interval

The CLI command 'ptp 0 sync-interval ..' sets the initial value of sync message interval. After exchanging signalling messages with neighbor, 'SyncIntervalSetting' state machine derives sync message interval to be used. This value can be overridden with user configured value using options 'usemgtSettableLogSyncInterval' and 'mgtSettableLogSyncInterval'. If 'usemgtSettableLogSyncInterval' is true, then the value of 'mgtSettableLogSyncInterval' is used as sync message interval. Otherwise, the values derived from 'SyncIntervalSetting' state machine are used.

```
(config-if)# ptp 0 mgtSettableLogSyncInterval -3
(config-if)# ptp 0 usemgtSettableLogSyncInterval 1
```

On the client side, sync message timeout in terms of number of sync message intervals can be configured using below CLI command.

```
(config-if)# ptp 0 sync-rx-to 3
```

In order to instruct the neighbor to change its sync interval via the message interval request TLV, the following command can be used

```
(config-if)# ptp 0 sync-interval -3
```

It is also possible to use the special values 126 (default) and 127 (stop) (section 10.6.4.3.7 - 802.1as-2020 standard).

```
(config-if)# ptp 0 sync-interval default
(config-if)# ptp 0 sync-interval stop
```

5.10.13. Clear gPTP statistics on port

It is possible to clear the PTP statistics on port using below CLI command.

```
(config-if)# ptp 0 statistics clear
```

If the port uses common mean link delay service(CMLDS) for peer delay measurement, then it is possible to use cmlds related CLI command for monitoring and clearing statistics.

```
(config-if)# ptp cmlds statistics clear
Port Parameter counter
---- ---------------------------------------- ----------
53 rxPdelayRequestCount 59899
```

```
rxPdelayResponseCount 59898
rxPdelayResponseFollowUpCount 59898
rxPTPPacketDiscardCount 1
pdelayAllowedLostResponsesExceededCount 1
txPdelayRequestCount 59899
txPdelayResponseCount 59899
txPdelayResponseFollowUpCount 59899

counters cleared
(config-if)# ptp cmlds statistics
Port Parameter counter
---- ---------------------------------------- ----------
53 rxPdelayRequestCount 6
rxPdelayResponseCount 6
rxPdelayResponseFollowUpCount 6
rxPTPPacketDiscardCount 0
pdelayAllowedLostResponsesExceededCount 0
txPdelayRequestCount 6
txPdelayResponseCount 6
txPdelayResponseFollowUpCount 6
(config-if)#
```

6. Calibration

6.1. Calibration of ethernet ports using external reference

It is possible to calibrate the latencies of ethernet ports using external reference system. To initiate calibration process and measure 1pps offsets between our device and reference system, the following CLI command needs to be used. This CLI command creates a client which tries to synchronise to reference device with ethernet encapsulation and by using basic servo filter.

```
# ptp cal port 10gig 1/10 start
```

Once the client device moves to 'PHASE_LOCKED' state, then 1pps offsets must be measured. These offsets must be provided as input to our device using below CLI command. Required latencies will be stored on our device after using this CLI command.

```
# ptp cal port 10gig 1/10 offset 20 cable-latency 5
```

From the next reboot, calibrated latencies will be applied to that port during link up time and they can be inspected using 'show ptp cal' command. If the user does not want to store calibrated latencies, they can be reset using below CLI command.

```
# ptp cal port 10GigabitEthernet 1/4 reset
```

6.2. Calibration of 1pps signal path delay

It is possible to measure the 1pps signal one way delay on our boards by using cables looped back between two input and output SMA connectors or RS-422 ports. After making required connections to SMA connectors, the following CLI command can be given. With RS-422 ports, the option 'rs422' must be used in CLI command accordingly.

```
# ptp cal 1pps 4 ?
rs422 RS-422 port is used for 1pps delay measurement.
sma SMA connectors are used for 1pps delay measurement.
# ptp cal 1pps 4 sma
Calibration of 1PPS input (cable_latency = 4)
Deleting any existing PTP instances
Creating PTP clock used for calibration
Now waiting up to 20 seconds for measurement to be performed.
Measured 1pps one-way delay through SMA connectors : 59

#
```

7. Status and information display

To monitor PTP status in web, click on the tabs 'monitor - ptp - ptp'. In the 'PTP status' page, click on the link under 'Inst' column to go to PTP Clock Status page. PTP Clock Status page contains all the non-port specific clock information.

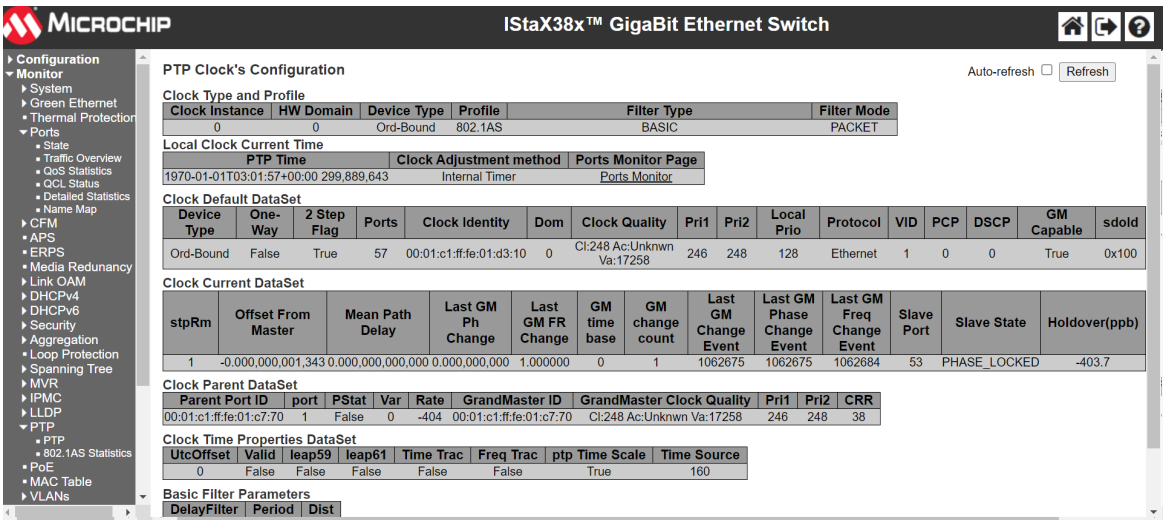

Figure 7. PTP Clock Status

To monitor PTP status on specific ports, click on the 'Ports Monitor' link under 'Local Clock Current Time' table in PTP Clock Status page.

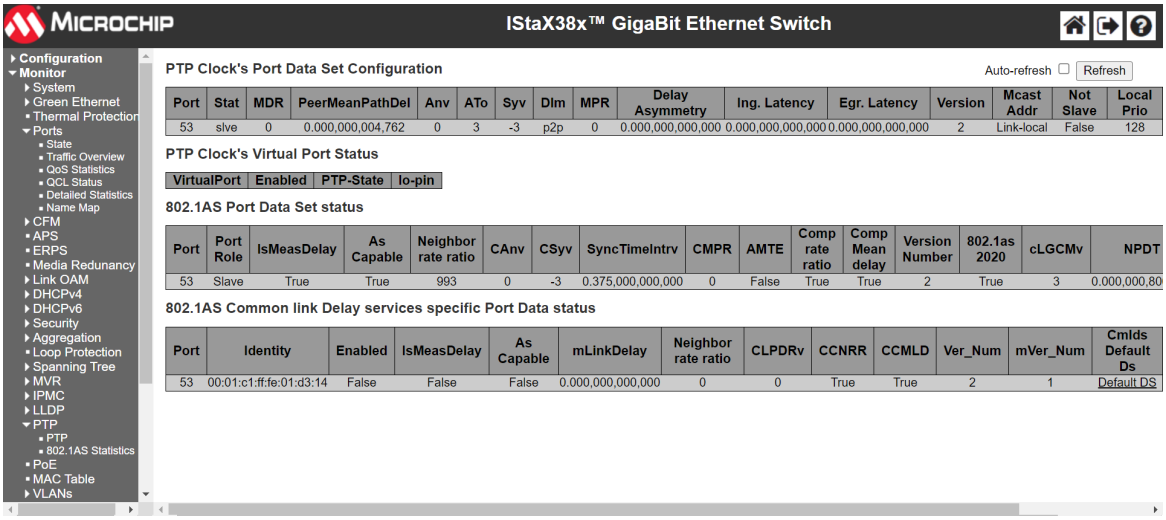

Figure 8. PTP Clock Port Status

Equivalent CLI commands for obtaining PTP clock status information are described below.

7.1. PTP Port status

PTP port status can be seen using following CLI command.

```
# show ptp 0 port-state interface 10gig 1/10
Port Enabled PTP-State Internal Link Port-Timer Vlan-forw Phy-timestamper Peer-delay
---- ------- --------- -------- ---- ---------- --------- --------------- ----------
10 TRUE slve FALSE Up In Sync Forward FALSE OK
```

After enabling PTP on a port, PTP port state must be verified whether the state moved to 'mstr' or 'slve'. If 'PTP-State' column shows 'dsbl', then PTP protocol is not running on that port.
If 'PTP-state' is disabled, then user must ensure 'Vlan-forw' status as 'Forward'. If the vlan forwarding status is shown as 'Discard', then it must be ensured that 'vid' value given in PTP instance mode configuration command is same as the VLAN-ID applied on the port. If both are same, then the CLI command 'show spanning-tree' must be used to see that the PTP port is in forwarding state.
If the port contains timestamping PHY, then the field 'phy-timestamper' will be set to TRUE. When phy timestamping is used, the phy LTC time and switch LTC time are compared and if the difference is high, then 'Port-Timer' goes 'out of sync'. The CLI command 'ptp ext outp auto' must be given to ensure 1pps signal is generated from switch to the phy.

7.2. PTP port-ds

PTP port data structures can be displayed using following command. When the delay mechanism is peer to peer, then 'PeerMeanPathDelay' is verified using 'port-ds' command. 'MeanPathDelay' in 'CurrentDS' will be '0' for peer to peer delay mechanism and instead 'PeerMeanPathDelay' must be referred in this case.

```
# # show ptp 0 port-ds interface 10gig 1/10
Port Enabled Stat MDR PeerMeanPathDel Anv ATo Syv SyvErr Delm MPR DelayAsymmetry IngressLatency EgressLatency Ver Lpri NoSlv McAdr Two-step NoMstr
---- ------- ---- --- ------------------- --- --- --- ------ ---- --- ------------------- ------------------- ------------------- --- ---- ----- ----- -------- ------
10 True slve 0 0.000,000,000,000 1 3 0 No e2e 0 0.000,000,000,000 0.000,000,000,000 0.000,000,000,000 2 128 False deflt Clk Def. False
```

7.3. PTP port statistics

Statistics of PTP protocol packets on specific port can be seen using following command.

```
# show ptp 0 port-statistics interface 10gig 1/10
Port Parameter counter
---- ---------------------------------------- ----------
10 rxSyncCount 7030
rxFollowUpCount 0
rxPdelayRequestCount 0
rxPdelayResponseCount 0
rxPdelayResponseFollowUpCount 0
rxAnnounceCount 3515
rxPTPPacketDiscardCount 0
syncReceiptTimeoutCount 0
announceReceiptTimeoutCount 0
pdelayAllowedLostResponsesExceededCount 0
txSyncCount 0
txFollowUpCount 0
txPdelayRequestCount 0
txPdelayResponseCount 0
txPdelayResponseFollowUpCount 0
txAnnounceCount 0
#
```

In Web, click on the tabs 'monitor - PTP - 802.1AS Statistics' to see port specific statistics.

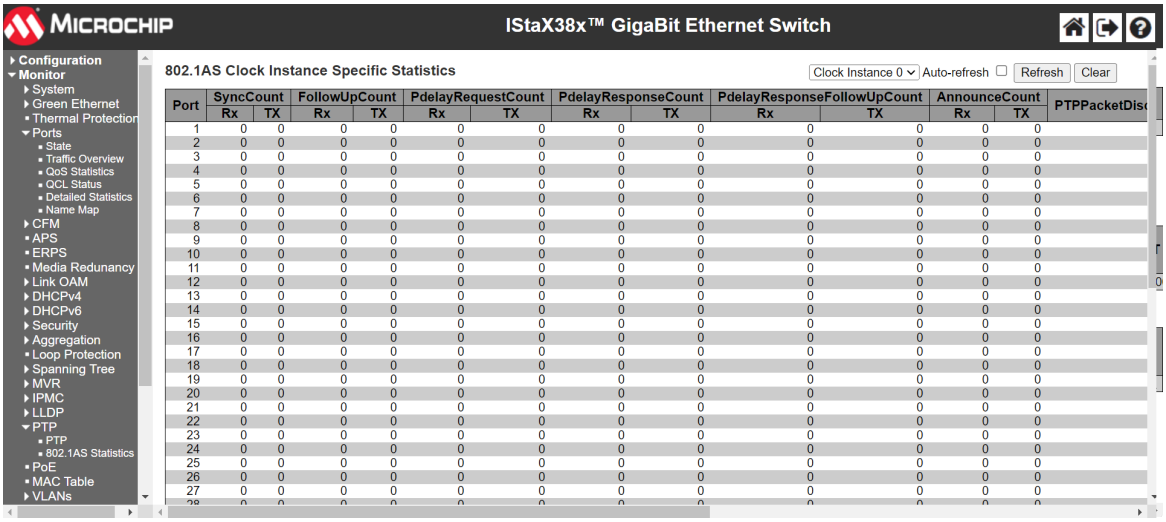

Figure 9. PTP port specific statistics

7.4. PTP Client generic statistics

Generic Statistics of PTP client common to entire clock can be enabled and monitored using below debug CLI commands. These statistics are useful to identify any packet loss.

```
# deb ptp 0 slave statistics enable
#
# deb ptp 0 slave statistics
master_to_slave_max : 0.000,000,001,787 sec
master_to_slave_min : 0.000,000,000,378 sec
master_to_slave_mean : 0.000,000,000,975 sec
master_to_slave_cur : 0.000,000,001,397 sec
slave_to_master_max : -NAN sec
slave_to_master_min : NAN sec
slave_to_master_mean : 0.000,000,000,000 sec
slave_to_master_cur : 0.000,000,000,000 sec
sync_pack_rx_cnt : 17
sync_pack_timeout_cnt : 0
delay_req_pack_tx_cnt : 0
delay_resp_pack_rx_cnt : 0
sync_pack_seq_err_cnt : 0
follow_up_pack_loss_cnt : 0
delay_resp_seq_err_cnt : 0
delay_req_not_saved_cnt : 0
delay_req_no_intr_cnt : 0
#
```

Virtual port statistics can be referred from section Statistics of 1pps and TOD on virtual port Receiver .

7.5. PTP current DS

PTP current data structure shows offset of client from server and mean path delay. When delay mechanism is peer to peer, peer mean path delay will be shown in 'port-ds'.

```
# show ptp 0 current
stpRm OffsetFromMaster MeanPathDelay
----- ------------------ ------------------
1 0.000,000,002,335 0.000,000,001,980
```

7.6. PTP client locking state

On client side, PTP application goes through various states like 'FREQ_LOCKING', 'FREQ_LOCKED', and 'PHASE_LOCKED' where each reflects the corresponding synchronisation status of application. PTP lock status can be seen using following command.

```
# show ptp 0 slave
Slave port Slave state Holdover(ppb)
---------- ------------ -------------
10 PHASE_LOCKED -142.1
#
```

With most of PTP configurations, when offset to server becomes less than 1000ns, 'PHASE_LOCKED' state is attained.

7.7. PTP client local time

Local clock time used by PTP instance can be verified using following command. Clock adjustment method shows source of clock i.e. LTC or Synce dpll or PTP dpll.

```
# show ptp 0 local-clock
PTP Time (0) : 1970-01-06T01:17:05+00:00 646,177,828
Clock Adjustment method: PTP DPLL
#
```

Local time stored in clock domain of chip can be retrieved using following debug command. This command is particularly useful when there are multiple clock domains available on chip.

```
# deb tod show clock domain 1 clock domain 1 : 1970/01/02
00:56:26.241301021 #
```

7.8. PTP client server negotiation data structures

Before forming client server relation, both the PTP devices communicate their clock attributes through announce messages. After receiving neighbor’s announce message, PTP application would decode parameters of that message, runs the BMCA algorithm to compare neighbor’s clock attributes with its attributes. To identify reasons for roles assigned to each device, it is needed to inspect attributes of client and server clocks using following commands. Attributes like clock quality(clock class, accuracy, variance). priority1, priority2 and port identity are shown in these CLI command output. The outputs from client must be compared with server to analyse the role deciding decisions.

Example 68. Default clock data structure

```
# show ptp 0 default
ClockId HW-Domain DeviceType Profile 2StepFlag Ports
vtss_appl_clock_identity
------- --------- ---------- ---------- --------- ----- ------------------------
0 0 Ord-Bound No profile False 21 00:01:c1:ff:fe:01:c7:70

Dom vtss_appl_clock_quality Pri1 Pri2 Lpri
--- ------------------------------ ---- ---- ----
0 Cl:248 Ac:Unknwn Va:00000 128 128 128

Protocol One-Way VID PCP DSCP PathTraceEnable
---------- --------- ----- --- ---- ---------------
Ethernet False 1 0 0 False
```

Example 69. Foreign master record showing clock attributes of neighbors

```
# show ptp 0 foreign-master-record
Port ForeignmasterIdentity ForeignmasterClockQality Pri1 Pri2 Lpri Qualif Best
---- --------------------------- --------------------------- ---- ---- ---- ------ -----
10 00:01:c1:ff:fe:01:d3:10 53 Cl:248 Ac:Unknwn Va:00000 128 10 128 True True
```

Example 70. Grandmaster’s clock attributes

```
# show ptp 0 parent
ParentPortIdentity port Pstat Var ChangeRate
--------------------- ---- ----- --- ----------
00:01:c1:ff:fe:01:d3:10 53 False 0 -142

GrandmasterIdentity GrandmasterClockQuality Pri1 Pri2
---------------------- ------------------------ ---- ----
00:01:c1:ff:fe:01:d3:10 Cl:248 Ac:Unknwn Va:00000 128 10
```

7.9. Time properties

PTP time properties provide the time format and traceability information. They can be seen using following command.

```
# show ptp 0 time-property
UtcOffset Valid leap59 leap61 TimeTrac FreqTrac ptpTimeScale TimeSource
--------- ----- ------ ------ -------- -------- ------------ ----------
0 False False False False False True 160
```

7.10. Filter-type

Current servo filter-type used by PTP application can be seen using following command.

```
# show ptp 0 filter-type
Clockinst: 0, filter type: aci-basic-phase-low
```

7.11. Unicast status

It is possible to verify unicast client or server status using following commands.

Example 71. unicast client

```
# show ptp 0 slave-table-unicast
Index IP-addr State MAC-addr Port Srcport clock id Srcport port Grant
----- ----------- ----- ----------------- ---- ---------------------- ------------ ------
0 1.1.1.1 SYNC 00-01-c1-01-d3-10 10 00:01:c1:ff:fe:01:d3:10 53 -3
#
# show ptp 0 uni
index duration ip_address grant CommState
----- -------- -------------- ----- ---------
0 100 1.1.1.1 -3 SYNC
1 100 0.0.0.0 0 IDLE
2 100 0.0.0.0 0 IDLE
3 100 0.0.0.0 0 IDLE
4 100 0.0.0.0 0 IDLE
#
```

Example 72. unicast server or reference

```
# show ptp 0 master-table-unicast
ip_addr mac_addr port Ann Sync
-------------- ----------------- ---- --- -----
1.1.1.2 00-01-c1-01-c7-70 53 0 -3
```

7.12. pps enabled status

1-pps enabled or not can be known using following command.

```
# show ptp ext
PTP External One PPS mode: Output, Clock output enabled: False, frequency : 1,
Preferred adj method : Auto, PPS clock domain : 1
```

7.13. RS-422 baudrate

Baudrate used by RS-422 interface can be known using following CLI command.

```
# show ptp rs422 baudrate
Parameters of RS422 port are: baudrate = 115200, parity = none, wordlength =
8, stopbits = 1, flags = 00000000
```

7.14. Calibrated latencies

After executing calibration test on a port, latencies obtained after calibration can be seen using following command.

```
# show ptp cal
PTP Port Calibration

Mode: 10M
Port Ingress latency Egress latency
-------------------------------------------------
21 0.000 0.000

Mode: 100M
Port Ingress latency Egress latency
-------------------------------------------------
1 0.000 0.000
3 0.000 0.000
4 0.000 0.000
-------------------------------------------------
- - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - -
-------------------------------------------------
```

Calibrated latencies must be explicitly applied on port to take effect.

7.15. Servo status, dpll type

In 8275 profile, servo switches between packet and hybrid mode. The current state of servo for each instance can be known using following CLI command.

```
# show ptp servo mode-ref
Servo [0] mode PACKET ref 0
Servo [1] mode NONE ref -1
Servo [2] mode NONE ref -1
Servo [3] mode NONE ref -1
```

While using Zarlink servo, device structure used by servo can be dpll specific or it can be generic dpll. Generic dpll is generally used for LTC adjustment. For example, if zl30772 dpll is used by servo for adjustment, then output is shown as dpll specific.

```
# show ptp servo source
Servo current source is type NONE ref 0, DPLL_type DPLLSpecific
```

End of Document.