

# GEM TSU Interface Details and IEEE 1588 Support

## Contents

Chapter 1: Introduction .....	2
Chapter 2: GEM TSU Clock Source .....	2
GEM TSU Clocking Source Selection in PCW .....	2
Use PS Internal IOPLL as GEM TSU Clock .....	2
Use MIO Clock as GEM TSU Clock .....	3
Use EMIO Clock as GEM TSU Clock .....	4
GEM TSU Clock block diagram .....	5
Internal IOPLL as TSU Clock Source Register Configuration .....	5
MIO50/51 as TSU Clock Source Register Configuration .....	5
EMIO as TSU Clock Source Register Configuration .....	6
Chapter 3: TSU and PTP signals.....	6
TSU Signals .....	6
PTP signals.....	8
Chapter 4: Testing PTP on board .....	9
Setup .....	10
Master Setup.....	10
Slave Setup.....	10
Steps for testing PTP .....	10
Expected PTP drift log .....	10
DTS example.....	11
Chapter 5: Accuracy for GEM PTP support on Zynq UltraScale+ .....	11
Chapter 6: Additional Information.....	12
Registers.....	12

## Chapter 1: Introduction

The 1588 time stamp unit (TSU) in GEM is a timer implemented as a 102 bit register. The 48 upper bits count seconds and the next 30 lower bits count nanoseconds and the lowest 24 bits count sub nanoseconds. The 54 lower bits roll over when they have counted to one second.

An external TSU port for GEM0 is exposed for users in the top level. The external port is 94 bits wide, conforming to the most significant bits of the internal TSU count as defined above.

## Chapter 2: GEM TSU Clock Source

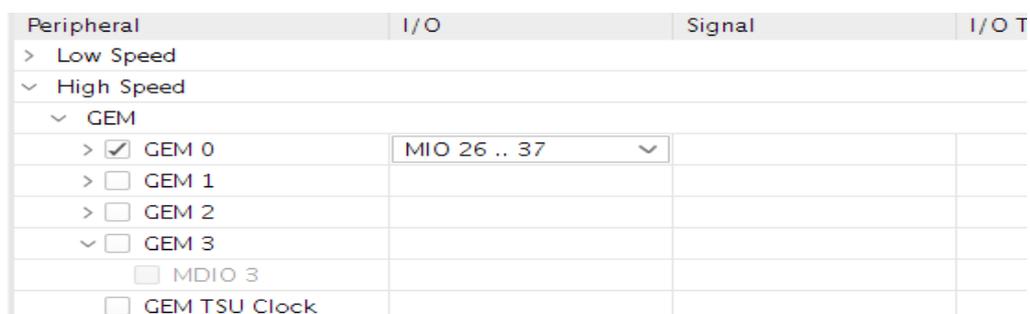
When using GEMs, currently there are 3 different clock sources you can select for the TSU clock.

The below section discusses how you make the selection in PCW configuration.

### GEM TSU Clocking Source Selection in PCW

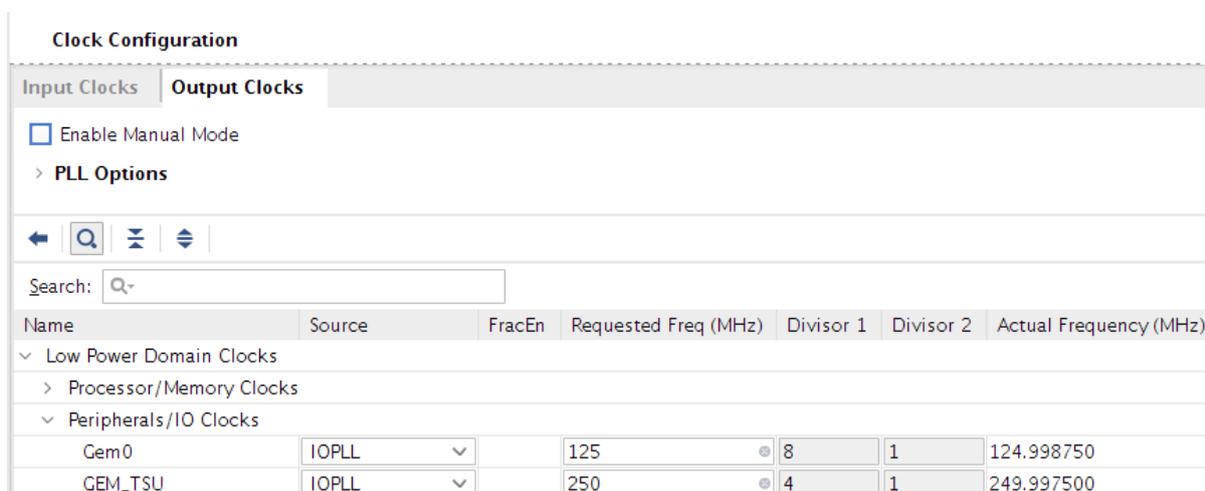
Use PS Internal IOPLL as the GEM TSU Clock

When you enable GEM (for example, GEM0 ) but without selecting the GEM TSU Clock, it will use the 250MHz PS internal IOPLL as the GEM TSU clock by default.



Peripheral	I/O	Signal	I/O T
> Low Speed			
> High Speed			
> <input checked="" type="checkbox"/> GEM 0	MIO 26 .. 37		
> <input type="checkbox"/> GEM 1			
> <input type="checkbox"/> GEM 2			
> <input type="checkbox"/> GEM 3			
> <input type="checkbox"/> MDIO 3			
> <input type="checkbox"/> GEM TSU Clock			

Figure 1: Enable GEM0



Clock Configuration						
Input Clocks		Output Clocks				
<input type="checkbox"/> Enable Manual Mode						
> PLL Options						
← 🔍 ⚙️ ⚡						
Search: 🔍						
Name	Source	FracEn	Requested Freq (MHz)	Divisor 1	Divisor 2	Actual Frequency (MHz)
> Low Power Domain Clocks						
> Processor/Memory Clocks						
> Peripherals/IO Clocks						
Gem0	IOPLL		125	8	1	124.998750
GEM_TSU	IOPLL		250	4	1	249.997500

Figure 2: GEM\_TSU Clock to PS IOPLL

When using GEM0, the emio\_enet0\_enet\_tsu\_timer\_cnt[93:0] port will be exposed to users. This port is only available for GEM0 due to the MIO pin count limitation. This port is the TSU timer count value. The upper 48 bits are seconds value and lower 46 bits are nanoseconds / sub-nanoseconds. Bit 46 toggles every second, i.e. 1 pps.



Figure 3: External TSU Timer Port Available

The fmio\_gem0\_tsu\_clk\_to\_pl\_bufg & fmio\_gem\_tsu\_clk\_from\_pl signals should also be exposed to the toplevel as fmio\_gem0\_tsu\_clk\_to\_pl\_bufg is the clock that the 94 bit timer is synchronizing to. In the 2018.2 PCW, this pair of signals are hidden with this configuration. This is planned to be fixed in a future release.

#### Use MIO Clock as GEM TSU Clock

To use MIO 50 or 51 as a TSU Clock, users will need to enable “GEM TSU Clock” and enable MIO in the PCW GUI.

Peripheral	I/O	Signal	I/O Type
> Low Speed			
> High Speed			
> GEM			
> <input checked="" type="checkbox"/> GEM 0	MIO 26 .. 37		
> <input type="checkbox"/> GEM 1			
> <input type="checkbox"/> GEM 2			
> <input type="checkbox"/> GEM 3			
<input checked="" type="checkbox"/> GEM TSU Clock	MIO 50		

Figure 4: Choose MIO 50 as TSU Clock Source

Input Clocks		Output Clocks	
Search: <input type="text"/>			
Name	Source	Input Freq (MHz)	Range (MHz)
Input Reference frequency			
PSS_REF_CLK	PS_REF_CLK	33.333	27.00 : 60.00
GT Lane Reference frequency			
Peripheral Reference frequency			
GEM TSU	MIO 50	250	0.000000 : 250.000000

Figure 5: GEM\_TSU Clock to MIO 50



Figure 6: TSU loopback Clock Ports Available

These two ports (fmio\_gem\_tsu\_clk\_to\_pl\_bufg and fmio\_gem\_tsu\_clk\_from\_pl ) can be connected in a loopback fashion. The 94 bit TSU timer port emio\_enet0\_enet\_tsu\_timer\_cnt is synchronous to fmio\_gem0\_tsu\_clk\_to\_pl\_bufg clock and local processing of timer data depends on this clock.

Please note, MIO\_26 should not be used for any GEM signal. This will be “fixed” in our PCW GUI in future version release.

#### Use EMIO Clock as GEM TSU Clock

To use EMIO for the TSU Clock, users will need to enable “GEM TSU Clock” and select EMIO in the PCW GUI.

Name	Source	Input Freq (MHz)	Range (MHz)
Input Reference frequency			
PSS_REF_CLK	PS_REF_CLK	33.333	27.00 : 60.00
GT Lane Reference frequency			
Peripheral Reference frequency			
GEM TSU	EMIO	250	0.000000 : 250.000000

Figure 7: GEM TSU Clock to EMIO

Another additional port emio\_enet\_tsu\_clk will be exposed to users.



Figure 8: EMIO TSU Clock Port Available

We have verified EMIO as a clock source by taking Si570 sysclk (300Mhz) input from a ZCU102 board and connecting it to an MMCM. The MMCM in the PL generates a 250Mhz clock which is connected to the EMIO TSU CLOCK named “emio\_enet\_tsu\_clk” in the Vivado IP Integrator design.

## GEM TSU Clock block diagram

Below is the block diagram for the GEM TSU Clock structure.

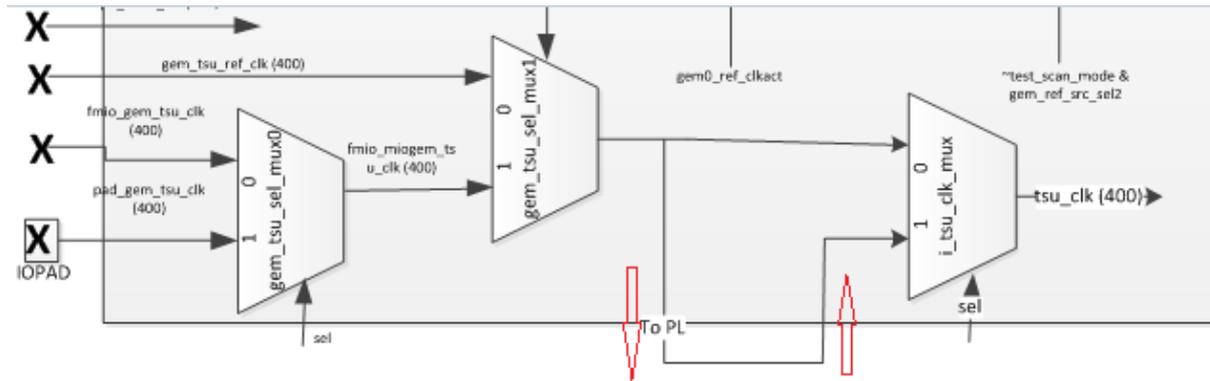


Figure 9: GEM TSU Clock Structure

The 3 clocks on the left side are the TSU Clock source, depending on what is being selected in PCW configuration.

- ✓ Gem\_tsu\_ref\_clk: PS Internal IOPLL
- ✓ fmio\_gem\_tsu\_clk: EMIO
- ✓ pad\_gem\_tsu\_clk: MIO

The two red-highlighted paths are fmio\_gem0\_tsu\_clk\_to\_pl\_bufg and fmio\_gem\_tsu\_clk\_from\_pl.

The MUX will decide which TSU Clock is in use.

When configuring the TSU Clock in PCW as per design requirement, the relative registers will be setup automatically in psu\_init.tcl.

### Internal IOPLL as TSU Clock Source Register Configuration

GEM\_CLK\_CTRL -

- bit[22] = 1'b0 or 1b'1 (see if the fmio signal is being looped back)
- bit[21:20] = 2b'00

GEM\_TSU\_REF\_CTRL -

- bit[2:0] = 3b'000 or 3b' 010 (Set divisors according to clock selection)
- bit[24] = 1

The above configuration has been tested.

You can also simply exposed the bits from tsu\_timer\_cnt that we need from probing (bit 45 and 46 for 1PPS) to an output header. I.E., tsu\_timer\_cnt[93:0] → Inverse of bit 45 is the 1PPS signal.

The register settings are the same as above and usually fmio\_gem\_tsu\_clk\_to\_pl\_bufg and fmio\_gem\_tsu\_clk\_from\_pl are looped back.

### MIO50/51 as TSU Clock Source Register Configuration

GEM\_CLK\_CTRL -

- bit[22] = 1b'0 or 1b'1 (See if the fmio signal is being looped back)
- bit[21:20] = 2b'11
- 

GEM\_TSU\_REF\_CTRL -

- bit[2:0] = 3b' 000 or 3'b010

- bit[24] = 1b'0

MIO\_PIN\_XX

- MIO\_PIN\_50/51 bit[1] = 1 as per user selection

The above configuration has not been tested as no hardware is available.

### EMIO as TSU Clock Source Register Configuration

GEM\_CLK\_CTRL -

- bit[22] = 1b'1 (fmio\_gem\_tsu\_clk\_to\_pl\_bufg and fmio\_gem\_tsu\_clk\_from\_pl should always be looped back)
- bit[21:20] = 2b'11

GEM\_TSU\_REF\_CTRL –

- bit[24] = 1b'0

In this case, MIO\_PIN\_50/51 should NOT be configured for GEM TSU clock, ie, 0xFF1800C8 is set to 0x00000010 and register 0xFF1800CC is set to 0x00000010. This way the clock is automatically picked from EMIO.

The documentation (U1087) will be updated for GEM\_CLK\_CTRL register bit[21:20] as below in a future release:

00/10: TSU clock from PLL

01: Reserved

11: TSU clock from MIO[50] or MIO[51] or EMIO

## Chapter 3: TSU and PTP signals

### TSU Signals

If users would like to control how the TSU registers are incremented, they can enable TSU signals in PCW and get access to emio\_enet0\_tsu\_inc\_ctrl[1:0] and emio\_enet0\_tsu\_timer\_cmp\_val signals.

PS-PL Configuration	
<div style="display: flex; align-items: center; gap: 10px;"> <span>←</span> <span>🔍</span> <span>⌵</span> <span>⌶</span> </div>	
Search: <input type="text" value="Q"/>	
Name	Select
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">v</span> General         </div>	
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">&gt;</span> Interrupts         </div>	
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">&gt;</span> Fabric Reset Enable <input checked="" type="checkbox"/> </div>	
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">&gt;</span> Address Fragmentation         </div>	
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">v</span> Others         </div>	
Use LPD_DMA	0 <span style="font-size: 0.8em;">v</span>
Use FPD_DMA	0 <span style="font-size: 0.8em;">v</span>
Use RTC	0 <span style="font-size: 0.8em;">v</span>
Use Event RPU	0 <span style="font-size: 0.8em;">v</span>
Use Proc Event Bus	0 <span style="font-size: 0.8em;">v</span>
Live Audio	0 <span style="font-size: 0.8em;">v</span>
Live Video	0 <span style="font-size: 0.8em;">v</span>
Enable DDR Refresh Signals	<input type="checkbox"/>
Errors to and from PMU	<input type="checkbox"/>
PMU AIB request and acknowl...	<input type="checkbox"/>
<div style="background-color: #f2f2f2; padding: 2px;"> <span style="font-size: 0.8em;">v</span> GEM 0         </div>	
TSU Signals	<input checked="" type="checkbox"/>
External FIFO Interface	<input type="checkbox"/>
PTP Interface	<input type="checkbox"/>

Figure 10: Enable TSU Signals



Figure 11: TSU Ports Available

If the gem\_tsu\_inc\_ctrl signal is not forced, the TSU timer behaves four different ways.

When gem\_tsu\_inc\_ctrl [1:0] =

- 2b'11: timer register increments as normal.
- 2b'01: timer register increments by an additional nanosecond.
- 2b'10: timer increments by a nanosecond less.

- 2b'00:
  - For GEM0, The “nanoseconds” timer register is cleared and the “seconds” timer register is incremented with each clock cycle.
  - For GEM1/2/3, the timer register increments as normal but the timer value is copied to the sync strobe register.

To reload the sync strobe registers, you will need to toggle gem\_tsu\_inc\_ctrl between 2b'11 and 2'b00.

The emio\_enet0\_tsu\_timer\_cmp\_val is the TSU timer comparison valid signal, synced with TSU Clock. It is asserted high when the upper 70 bits of the TSU timer count value are equal to the programmed comparison value.

### PTP signals

When enabling PTP signals, the PTP interface will be exposed too.

Name	Select
<ul style="list-style-type: none"> <li>General               <ul style="list-style-type: none"> <li>Interrupts</li> <li>Fabric Reset Enable <input checked="" type="checkbox"/></li> <li>Address Fragmentation</li> </ul> </li> <li>Others               <ul style="list-style-type: none"> <li>Use LPD_DMA 0</li> <li>Use FPD_DMA 0</li> <li>Use RTC 0</li> <li>Use Event RPU 0</li> <li>Use Proc Event Bus 0</li> <li>Live Audio 0</li> <li>Live Video 0</li> <li>Enable DDR Refresh Signals <input type="checkbox"/></li> <li>Errors to and from PMU <input type="checkbox"/></li> <li>PMU AIB request and acknowl... <input type="checkbox"/></li> </ul> </li> <li>GEM 0               <ul style="list-style-type: none"> <li>TSU Signals <input checked="" type="checkbox"/></li> <li>External FIFO Interface <input type="checkbox"/></li> <li>PTP Interface <input checked="" type="checkbox"/></li> </ul> </li> </ul>	

Figure 12: Enable PTP Interface

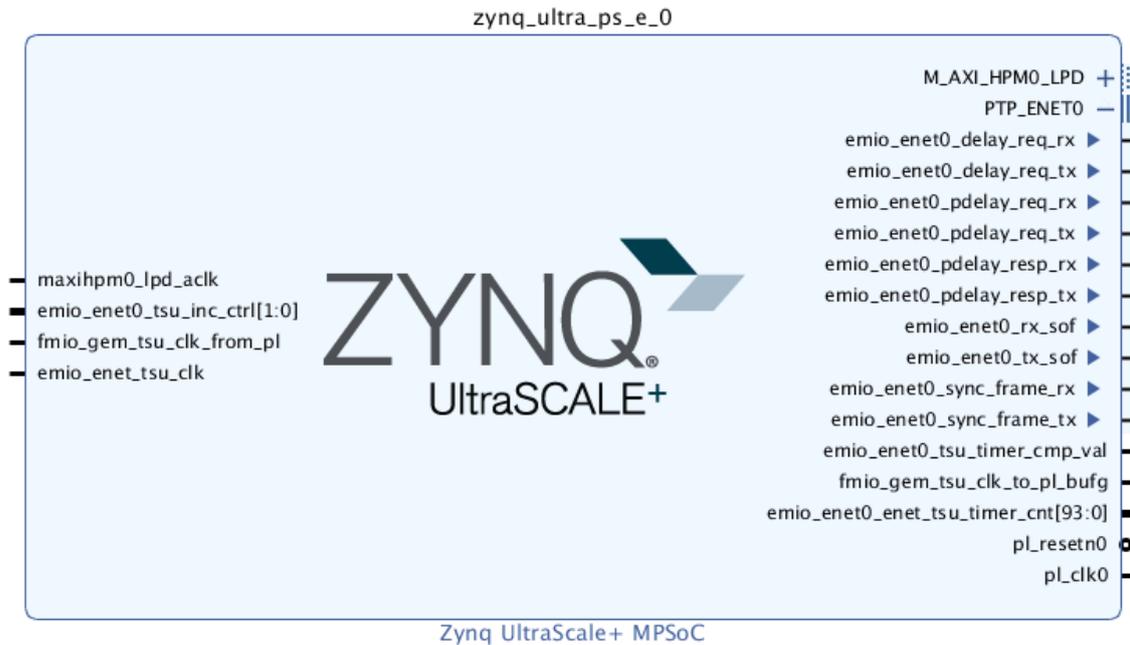


Figure13: PTP Interface Available

IEEE 1588 PTP frame recognition and Time Stamp Unit signals:

- ✓ emio\_enet0\_delay\_req\_rx: Asserted high synchronous to rx\_clk if PTP delay request frame is detected on receive.
- ✓ emio\_enet0\_delay\_req\_tx: Asserted high synchronous to tx\_clk if PTP delay request frame is detected on transmit.
- ✓ emio\_enet0\_pdelay\_req\_rx: Asserted high synchronous to rx\_clk if PTP peer delay request frame is detected on receive.
- ✓ emio\_enet0\_pdelay\_req\_tx: Asserted high synchronous to tx\_clk if PTP peer delay request frame is detected on transmit.
- ✓ emio\_enet0\_pdelay\_resq\_rx: Asserted high synchronous to rx\_clk if PTP peer delay response frame is detected on receive.
- ✓ emio\_enet0\_pdelay\_resq\_tx: Asserted high synchronous to tx\_clk if PTP peer delay response frame is detected on transmit.
- ✓ emio\_enet0\_rx\_sof: Asserted high synchronous to rx\_clk when the SFD is detected on a receive frame
- ✓ emio\_enet0\_tx\_sof: Asserted high synchronous to tx\_clk when SFD is detected on a transmit frame, deasserted at end of frame.
- ✓ emio\_enet0\_sync\_frame\_rx: Asserted high synchronous to rx\_clk if PTP sync frame is detected on receive.
- ✓ emio\_enet0\_sync\_frame\_tx: Asserted high synchronous to tx\_clk if PTP sync frame is detected on transmit.

## Chapter 4: Testing PTP on board

This test validates MACB Linux driver's IEEE1588 support with master-slave synchronization. In order to test this feature in MACB, make sure the HW timestamp support option is turned on.

For example, in the older kernel, MACB\_USE\_HWSTAMP needs to be enabled. In 2017.4, it is CONFIG\_MACB\_EXT\_BD; and in 2018.1 it is CONFIG\_MACB\_USE\_HWTSTAMP.

<https://github.com/Xilinx/linux-xlnx/blob/master/drivers/net/ethernet/cadence/Kconfig>

This takes care of PTP dependencies in kernel as well. The TSU clock is obtained from the design via DTG. It is typically 250MHz.

## Setup

A test setup is required where the ZynqMP board is typically the slave. The master is usually a setup with stable high precision clock used for hardware time stamping and (optionally) a 1PPS signal output.

### Master Setup

The Oregono SYN1588 NIC card is one such card that we used. It can be inserted into the PCIe slot of any Linux machine. This card comes with a custom boot image in a pendrive. You can insert it into the Linux machine and boot from it.

In this case, install Linux PTP on the Linux machine and use the corresponding commands.

### Slave Setup

Use any ZynqMP board with an Ethernet interface. If planning on capturing a 1PPS signal on the CRO, then use DC4 with GEM0 only. In order to capture 1PPS signal from ZynqMP, the design has to be updated to pull out the "Inverse of bit 45" of GEM0 time counter to a probe-able header on the board.

### Steps for testing PTP

On the Linux Host machine (Master):

```
ptp4l -i <interface_name> -m
```

On the ZynqMP (Slave):

```
ptp4l -i eth0 -s -m
```

### Expected PTP drift log

Slave (ZynqMP) synchronization output sample:

```
e2e_slave
ptp4l -i eth0 -s -m
e2e_master
ptp4l -i <interfacename> -m
e2e_slave
ptp4l -i eth0 -s -m
root@Xilinx-ZC1751-DC1-2016_1:~# ptp4l -i eth0 -s -m
ptp4l[136.261]: selected /dev/ptp0 as PTP clock
ptp4l[136.262]: driver changed our HWTSTAMP options
ptp4l[136.262]: tx_type 1 not 1
ptp4l[136.262]: rx_filter 1 not 12
ptp4l[136.262]: port 1: INITIALIZING to LISTENING on INITIALIZE
ptp4l[136.262]: port 0: INITIALIZING to LISTENING on INITIALIZE
ptp4l[136.391]: port 1: new foreign master 001ec0.ffff.862d3a-1
ptp4l[140.391]: selected best master clock 001ec0.ffff.862d3a
ptp4l[140.391]: port 1: LISTENING to UNCALIBRATED on RS_SLAVE
ptp4l[142.568]: master offset -506797907644 s0 freq +0 path delay -369
ptp4l[143.568]: master offset -506797901693 s1 freq +5951 path delay -833
ptp4l[144.568]: master offset -1165 s2 freq +4786 path delay -833
ptp4l[144.568]: port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
ptp4l[145.568]: master offset -930 s2 freq +4671 path delay -369
ptp4l[146.568]: master offset -117 s2 freq +5205 path delay -369
ptp4l[147.568]: master offset -16 s2 freq +5271 path delay -202
```

ptp4l[148.568]: master offset 38 s2 freq +5320 path delay -36  
ptp4l[149.568]: master offset 95 s2 freq +5389 path delay 70  
ptp4l[150.568]: master offset 46 s2 freq +5368 path delay 214  
ptp4l[151.568]: master offset 165 s2 freq +5501 path delay 214  
ptp4l[152.568]: master offset 120 s2 freq +5506 path delay 257  
ptp4l[153.568]: master offset 107 s2 freq +5529 path delay 257  
ptp4l[154.568]: master offset 51 s2 freq +5505 path delay 275  
ptp4l[155.568]: master offset 45 s2 freq +5514 path delay 275  
ptp4l[156.568]: master offset -17 s2 freq +5465 path delay 310  
ptp4l[157.568]: master offset -7 s2 freq +5470 path delay 310  
ptp4l[158.568]: master offset 17 s2 freq +5492 path delay 310  
ptp4l[159.568]: master offset -16 s2 freq +5464 path delay 314  
ptp4l[160.568]: master offset 4 s2 freq +5480 path delay 314  
ptp4l[161.568]: master offset -1 s2 freq +5476 path delay 314  
ptp4l[162.568]: master offset -16 s2 freq +5460 path delay 315  
ptp4l[163.568]: master offset 8 s2 freq +5480 path delay 311  
ptp4l[164.568]: master offset -3 s2 freq +5471 path delay 311  
ptp4l[165.568]: master offset -8 s2 freq +5465 path delay 311  
ptp4l[166.568]: master offset 7 s2 freq +5478 path delay 309  
ptp4l[167.568]: master offset -17 s2 freq +5456 path delay 312  
ptp4l[168.568]: master offset -7 s2 freq +5461 path delay 312  
ptp4l[169.568]: master offset -5 s2 freq +5461 path delay 313  
ptp4l[170.569]: master offset -5 s2 freq +5459 path delay 313  
ptp4l[171.569]: master offset -5 s2 freq +5458 path delay 313  
ptp4l[172.569]: master offset 2 s2 freq +5463 path delay 313

### DTS example

```
&gem0 {  
    tsu-clk=<250000000>;  
    ...  
};
```

## Chapter 5: Accuracy for GEM PTP support on Zynq UltraScale+

The MACB driver supports IEEE 1588 for the Zynq MPSoC GEM. Master-slave synchronization was tested using a Linux PTP application with a Linux server as mastered by our engineering team. +/- 50ns drift was observed.

The accuracy can be brought down to < +/- 10ns (7 is what we observed in external compliance at UNH-IOL (University of New Hampshire Inter-Operability lab)). Please note that low drift such as +/- 5ns will take some tuning and is extremely dependent on the system. We do not universally claim that this drift is achievable.

For this, the static offset will have to be subtracted – this is specific to the system. After synchronization, this static offset for a system can be observed by plotting the drift overnight, for example, (or capturing it in a CRO (oscilloscope)) and equalizing it to see the mean drift. The drift is commonly captured by connecting an oscilloscope to both the master and slave and analyzing their 1PPS signal. Then you will have to subtract/add this parameter inside the driver's adjfreq function to improve the actual drift.

## Chapter 6: Additional Information

### Registers

#### TSU\_TIMER\_INCR

- "ns\_increment" and "tsu\_timer\_incr\_sub\_nsec" are not used with "alt\_ns\_incr".
- "num\_incs" will be used along with alt\_ns\_incr only, but our current driver does not use alt\_ns\_incr.

#### NETWORK\_CONTROL

- ext\_tsu\_port\_enable bit has no effect.

#### GEM\_CLK\_CTRL

- bit[21:20] –
  - 00/10: TSU clock from PLL
  - 01: Reserved
  - 11: TSU clock from MIO[50] or MIO[51] or EMIO