Speedster7t Ethernet User Guide (UG097)

Speedster FPGAs

Preliminary Data

Achronix®
Data Acceleration

Copyrights, Trademarks and Disclaimers

Preliminary Data

This document contains preliminary information and is subject to change without notice. Information provided herein is based on internal engineering specifications and/or initial characterization data.

Achronix Semiconductor Corporation

2903 Bunker Hill Lane
Santa Clara, CA 95054
USA

Website: www.achronix.com
E-mail : info@achronix.com

Table of Contents

Chapter - 1: Introduction

Summary

Speedster®7t devices include high-speed Ethernet interfaces, which can support a wide variety of Ethernet packet protocols and speeds of up to 400 Gbps per channel. These Ethernet interfaces are paired with latest generation SerDes which individually support 100 Gbps data rates. With eight of these SerDes per Ethernet interface, each interface can support 2× 400 Gbps Ethernet IP channels.

The number of Ethernet interfaces varies according to the device. In the descriptions below, the AC7t1500 device is used as an example. This device has two Ethernet interfaces, allowing for 4× 400 Gbps interfaces, for a combined total bandwidth of 1.6 Tbps.

Features

400G/200G PCS Layer

- 400G over 8× 50G SerDes or 4× 100G SerDes
- 200G over 4× 50G SerDes or 2× 100G SerDes

> **Note**
>
> 400G/200G do not support 25G SerDes. The following configurations are not available; 16× 25G or 8× 25G.

100G PCS Layer

- 100G Base-R PCS according to IEEE 802.3 Clause 82 specification.
- 2× SerDes lane with 53 Gbps or 1x SerDes lane with 106 Gbps.
- 4× SerDes lane with 25G (KR4) or 26.5G (KP4).
- Supports Reed-Solomon FEC (RS-FEC) implementing RS(528,514) and RS(544,514) for 100G-KR and 100G-KP applications respectively.

10/25/40/50G PCS Layers

- Configurable Base-R PCS compliant with IEEE 802.3 Clauses 49, 82, 107, 133 for 10G, 25G, 50G operation respectively.
- Independent 64bit XLGMII MAC interfaces per channel.
- Supports Reed-Solomon FEC (RS-FEC) implementing RS(528,514) and RS(544,514) for 25G and 50G applications.
- Optional support for EEE fast-wake (i.e., transfer of LPI sequences, no deep sleep).
- Optional Base-R (Firecode) FEC according to Clause 74 of IEEE 802.3.

Reed-Solomon FEC (RS-FEC)

- Support for RS(528, 514) (KR) codewords and RS(544, 514) (KP) depending on mode of operation.
- Support for RS(272, 258) low-latency variant.
- Support for 25G (Clause 108) and 50G (Clause 134) and 25/50G Ethernet Consortium specifications.
- Support for error indication to PCS when uncorrectable errors are detected.

MAC

- 1588 precision timing, one-step operation, for for all data rates, 10 to 400G.
- IEEE 802.3br is supported in 10...100G by providing two transmit and receive interfaces to the application.

System Multirate and Multichannel

The Ethernet interface can be configured with up to eight lanes of SerDes and PCS. Each lane is independently usable for 10G or 25G or 50G or 100G Ethernet rates.

- Up to four 50G Ethernet channels using two 25 Gbps lanes each.
- Up to two 100G Ethernet channels using four 25 Gbps lanes each.
- Up to four 100G Ethernet channels using two 50 Gbps lanes each.
- Up to two 200G Ethernet channels using two 50 Gbps lanes each.
- Up to four 200G Ethernet channels using two 100 Gbps lanes each.
- Up to two 400G Ethernet channels using four 100 Gbps lanes each.
- One 400G Ethernet channel using eight 50 Gbps lanes.

Architecture

The architecture of each Ethernet interface is shown below.

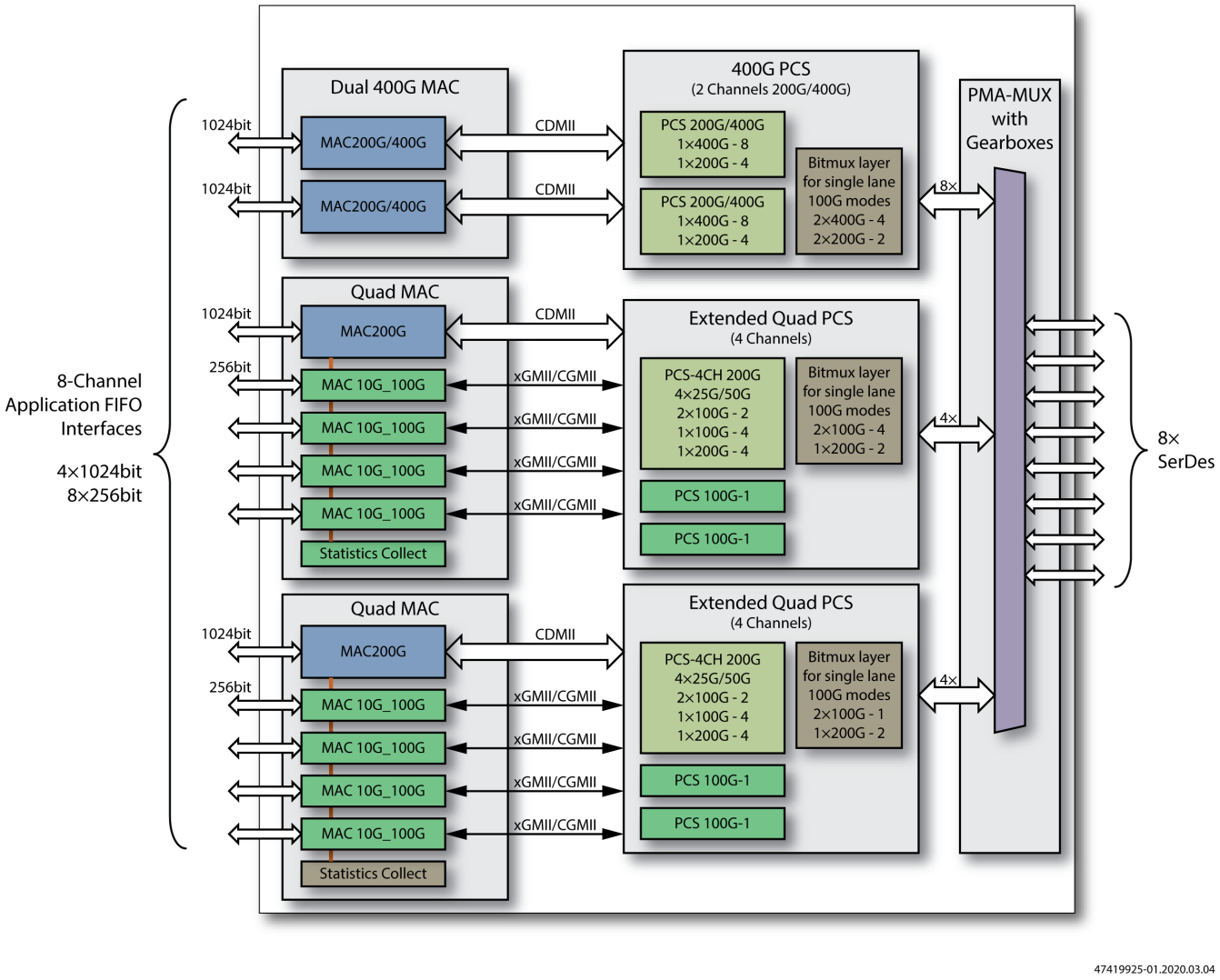


Figure 1: ***Ethernet Interface Block Diagram***

PMA (SerDes)

The physical media attachment (PMA) block consists of eight next-generation SerDes. Each SerDes can operate at up to 106.25 Gbps and down to 10.3125 Gbps. In normal operation the PMA block uses a dedicated mux to connect the SerDes directly to the PCS layer. If the user requires the SerDes for applications other than Ethernet (or to use their own Ethernet PCS and MAC), then the PMA mux can be set to connect the SerDes interface directly to the fabric.

> **Note**
>
> The PMA mux must be switched for all SerDes signals within an interface; therefore, if the user wishes to use any SerDes from an Ethernet interface directly, then all SerDes from that group are switched to the fabric, and the related Ethernet MAC and PCS are not longer available for use

PCS

The physical coding sublayer (PCS), connects between the SerDes and the MAC. The PCS consists of a dual-channel 400G PCS which can be configured to support either 2× 200G or 2× 400G operation, and twin Quad-PCS, each of which implements a combined four channel PCS, supporting up to a single channel of 200G operation, or four channels with up to 100G per channel. The PCS layer provides the coding functions for the various channel rates supported, including Reed-Solomon error correction of RS(528, 514) (KR) and RS(544, 514) (KP) code words. In addition, support for 25G (Clause 108) and 50G (Clause 134) error correction and coding are also supported.

MAC

The media access controller (MAC) is constructed from three blocks: one a dedicated dual channel 400G MAC for 400G/200G operation, which connects to the dual channel 400G PCS, and then two instances of a Quad-MAC, each connecting to a Quad-PCS. Each of the Quad-MAC can support a single channel of 200G, or four channels operating at 100G down to 10G per channel. The dedicated 400G MAC is optimized for higher data rates and the wider bus widths necessary for the faster interfaces; the Quad-MAC is equally optimized for multiple channels of lower data rates.

Specifications

Channels

Each Ethernet interface can support the following combinations of data rates

Table 1: *Channel configurations*

Mode – Lanes	SerDes Lanes (per Channel)	SerDes Rate per Lane	Possible No. of Channels	Description (with Coding Options)
400G – 8	8	53.125G	1	400G over 8 lanes (2:1 bitmux)
400G – 4	4	106.25G	2	400G over 4 lanes (4:1 bitmux)
200G – 4	4	53.125G	2	200G over 4 lanes (2:1 bitmux)
200G – 2	2	106.25G	4	200G over 2 lanes (4:1 bitmux)
100G – 4	4	25.78125G or 26.5625G	2	100G over 4 lanes (no FEC, RSFEC-KR4 or RSFEC-KP4)
100G – 2	2	53.125G	4	100G over 2 lanes (RSFEC-KP 2:1 bitmux)
100G – 1	1	106.25G	8	100G over 1 lane (RSFEC-KP 4:1 bitmux)
50G – 2	2	25.78125G or 26.5625G	4	50G single lane (RSFEC-KR or RSFEC-KP)
50G – 1	1	53.125G	8	50G single lane (RSFEC-KP 4:1 bitmux)

Mode – Lanes	SerDes Lanes (per Channel)	SerDes Rate per Lane	Possible No. of Channels	Description (with Coding Options)
40G – 4	4	10.3125G	2	40G over 4 lanes (optional Base-R (Firecode) FEC according to Clause 74 of IEEE 802.3)
40G – 2	2	20.625G	4	40G over 4 lanes (optional Base-R (Firecode) FEC according to Clause 74 of IEEE 802.3), 2:1 bitmux
25G – 1	1	25.78125G	8	25G single lane (66b or RSFEC-KR)
10G – 1	1	10.3125G	8	10G single lane (66b)

Bitmux

Bitmux refers to a layer of multiplexing that occurs between the PCS and the SerDes. This multiplexing provides the ability to widen the data interface to the PCS, (and MAC), while reducing the overall system frequency.

For example, 400G-8. requires 8 50G SerDes lanes. From the SerDes table (see page 11) below it can be seen that when configured as 50G, the SerDes is set to a 64-bit interface, operating at 83 0MHz. By using a 2:1 bitmux, the data bus is widened to 16 lanes of PCS, with each PCS operating at 415 MHz.

SerDes

The SerDes supports the following modes and interface width combinations

Table 2: ***SerDes Interface Configuration and Frequencies***

SerDes Speed	PMA Interface Width	Active SerDes Width(3)	PMA Interface Frequency
10.3125 Gbps (NRZ)	128	32/64	32 : 322.265635 MHz 64 : 161.1328125 MHz
25.78125 Gbps (NRZ)	128	32/64	32 : 805.6640625 MHz 64 : 402.83203125 MHz
26.5625 Gbps (NRZ)	128	32/64	32 : 830.078125 MHz 64 : 415.0390625 MHz
26.5625 Gbps (PAM4)	128	32/64	32 : 830.078125 MHz 64 : 415.0390625 MHz
53.125 Gbps (PAM4)	128	64	830.078125 MHz
106.25 Gbps (PAM4)	128	128	830.078125 MHz

Chapter - 2: System Architecture

Ethernet Interface System Architecture

Due to the fully integrated architecture of the Speedster7t device, the connections to and use of the Ethernet interface subsystem, are different from what a user may be used to. In previous FPGA architectures, a user would expect to generate an RTL wrapper which contained the Ethernet interface (perhaps including also models of the SerDes). This wrapper would then be instantiated within a design, and the user would directly connect to the clock, reset, status and data ports. The user would then be expected to ensure the correct clock and reset strategies were followed, and confirm the integrity of the data connections.

With a Speedster7t device, the usage methods and connections are very different. The full Ethernet interface, containing MAC, PCS and PMA (SerDes) is fully integrated within the Speedster7t device system components. The MAC connects via an Ethernet interface unit (EIU (see page 15)) directly to the network-on-chip (NoC). These components and the connections between them are not accessible to the user, and the user design cannot connect directly to the Ethernet interface components. This architecture is as shown below (see page 13)

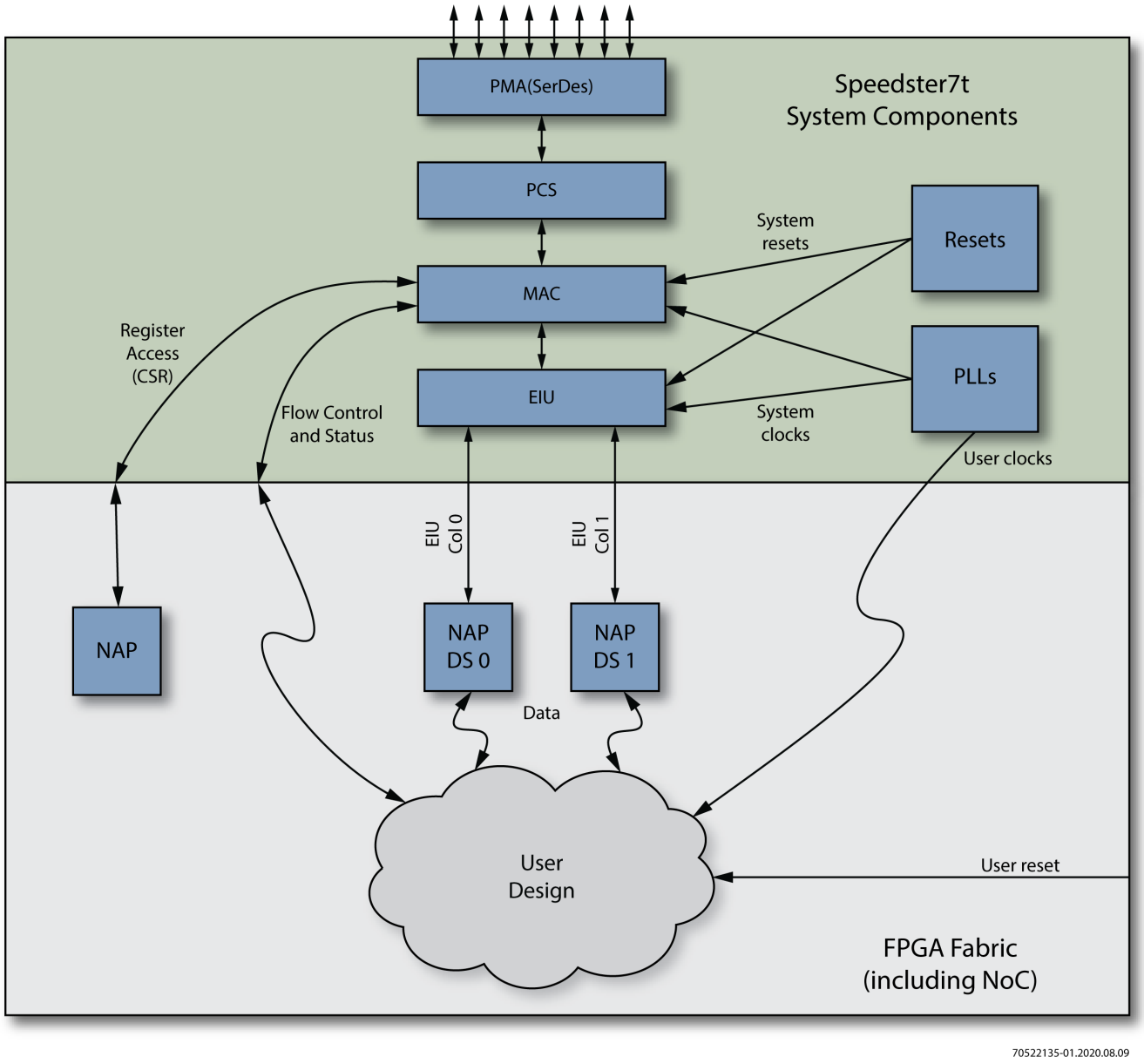


Figure 2: ***Ethernet Interface System Architecture***

Data Flow

The basic data flow of a user Ethernet design is as follows:

- Transmit data, in the form of whole packets, is input to a NAP
- The packet flows up the NoC column to the EIU (see page 15). Within the EIU (see page 15) the packet can be passed through, or stored and forwarded depending on the required mode. For the advanced modes(Packet and Quad Segmented Modes (see page)) the EIU (see page 15) will assemble and format the packets accordingly.
- The whole packet is transmitted in parallel form to the MAC, which processes the packet, adding preambles and FCS.
- The packet is passed through the PCS to the SerDes, and hence to the serial pins of the device.

Packet reception is a reverse of the above.

With reference to the above architecture (see page 13), the user has to make connections to the Ethernet interface with the following methods;

Data Connections

The user connects to the Ethernet data streams using network access points (NAPs) placed within the NoC. The use of the NoC greatly simplifies the access to the Ethernet interface, with simplified data streams presented to the user.

> **Note**
>
> For data streams, the user design only connects to the NAPs; it does not connect directly to the MAC or EIU.

For full details of the NAP data connections, refer to NAP Data Connections. (see page 17)

Status and Flow Control Signals

Although the data streams, and accompanying metadata are delivered directly from the NAP, flow control and MAC status are connected directly to the FPGA fabric using the Ethernet direct-connect (DC) interface. When the Ethernet interface is configured within ACE, the appropriate flow control and status signals are created and set to be connected via the DC Interface to the FPGA fabric. The user can then connect directly to these signals within their design. The description of these signals is detailed in the Direct-Connect Interface table (see page 24).

> **Note**
>
> The names and types of signals vary according to the configuration of the various MACs; each MAC type has a different set of control and status signals.

Control and Status Registers

Access to the internal Ethernet controller registers, is also performed via the NoC. The access is performed using a NAP, set for AXI-4 mode, which is used to access the control and status register (CSR) address space. This NAP can access any of the control registers in any of the interface subsystems on the Speedster7t device. Similar to the data streams, the user design does not connect directly to the MAC or PCS, instead access to the CSRs is only supported via the NoC.

> **Note**
>
> Correct configuration of the Ethernet interface is performed at power-up using the ACE-generated configuration. Typically a user design would only require CSR access to monitor status or to enable or disable extended features.

IP Generation and Configuration

Generation and configuration of the Ethernet interface is performed using ACE I/O Designer as detailed in Ethernet IP Software Support in ACE (see page 43). The configuration created within ACE sets the appropriate channel modes while ensuring the correct clocks and frequencies are provided to the Ethernet interface. ACE subsequently generates the following files:

- A bitstream file for the Ethernet interface. This file is merged with other bitstream files during final bitstream assembly. This bitstream programs the Ethernet interface upon FPGA power-up, ensuring the Ethernet interface is ready for operation once the device enters user mode.
- A simulation configuration file. This file is input to the simulation environment to set Ethernet interface configuration.

With the above generation and configuration flow, combined with the fully integrated nature of the Speedster7t device, the user no longer has the responsibility for ensuring the right clocking and reset strategies are followed. The Speedster7t device, combined with ACE, manage these low-level tasks leaving the user to concentrate on the core parts of their design.

EIU

The Ethernet interface unit (EIU) is unique to the Speedster7t family of devices. There is an EIU for each Ethernet interface, and each instance manages the connection of the MAC data interfaces to the NoC. The EIU adapts the traffic flow and performs clock domain crossings between the NoC and the MAC. It is further responsible for dividing the traffic when one of the 400G or 200G, packet or quad modes is selected. For all modes, the EIU is responsible for packetizing the traffic so it can be sent down the appropriate NoC column to the correct NAP endpoint.

NAP Columns

Each EIU supports two NoC columns which the Ethernet NAP endpoints can be located on. The EIU will only send and receive packets to NAPs placed within those two columns. The particular columns that an EIU is connected to for each Ethernet interface is detailed in the table below. (see page 15)

Table 3: *Ethernet NoC columns*

Device	Ethernet Subsystem		Total NoC Columns
	0	1	
ac7t1500	Columns 0 & 1	Columns 4 & 5	4

Memory Buffering

The EIU has memory buffering, primarily in order to do the higher line rate packet division and rearrangement. In addition the buffering is used for clock crossing and packetization. The EIU buffering operates in two modes, according to the selected line rate

- For 200 and 400G Packet and Quad Segmented Modes, (see page) the EIU operates in a store-and-forward mode. Each whole packet is buffered until it is fully received from the NoC, and then it is forwarded to the MAC. A similar scheme operates for received packets.
- In all other modes, no buffering is performed; therefore, the EIU should be considered as a transparent block with minimal buffering. This mode places frequency and throughput restrictions on the NAPs; these restrictions are discussed below.

Each NAP only has a shallow buffer in order to support clock domain crossing. For a system design the NAPs should also be considered transparent elements with minimal or no buffering capability.

Clocks

The EIU bridges between the NoC core operating frequency, 2 GHz, and the MAC `ff_clk` frequency domains. Each of these clock domains can be configured by the user. The NoC operating frequency is fixed, and the user must provide a 2 GHz clock as detailed in Ethernet IP Software Support in ACE (see page 43). The minimum MAC `ff_clk` frequencies are defined in the Reference and FIFO Clock frequencies. (see page 34)

Packet and Quad Segmented Modes

At higher data rates, 200G and 400G, it is impractical to transmit the full bandwidth through a single NAP (for 400 Gbps, this rate would require an operating frequency of 2 GHz for the associated fabric logic). Therefore, the EIU and NoC support spreading the bandwidth across four NAPs. There are then two operating modes for these four NAPs:

- **Quad segmented mode (QSI)**. The four NAPs are combined to form a single 1024-bit bus. The packet is rotated around the four NAPs such that a new packet will start on the lane after the previous packet ended. For example, if the NAPs are identified as NAP-1 to NAP-4, then if the end-of-packet (EoP) word is output on NAP-2, and the next packet asserts it's start-of-packet (SoP) word on NAP-3.
- **Packet mode**. The four NAPs each process a whole packet, providing four 100G streams to the fabric. Each NAP operates through its 256-bit data bus interface. The EIU transmits the next available packet to whichever NAP is not currently transmitting a packet.

For additional details on each of these modes, including diagrammatic representations of the packet structure, refer to Speedster7t Ethernet NoC Connectivity. (see page 28)

Mode Considerations

The two schemes each have advantages and disadvantages:

- For QSI, packet ordering is guaranteed as the NAPs present a single 1024-bit bus for both transmit and receive. However, the width of the bus can create routing challenges, or difficulties with interfacing the data to other interface subsystems, such as memory. The NAP's native data width is 256 bits, requiring buffering and pipelining for the 1024-bit bus to be written to another NAP. In addition the requirement to barrel shift the packet around the bus for different packet start lanes can add complexity to a design.
- Packet mode has the advantage of having whole packets contained within a single 256-bit bus, which can then be easily connected to other NAPs, and hence to other interface subsystems or other parts of the design. However, in packet mode the four streams are separate with no guaranteed packet ordering. In many systems, this lack of ordering is not an issue; however, if point-to-point communication is being used, and packet ordering is required, then packet ordering will need to be implemented.

Note

The Ethernet standard does not guarantee packet ordering due to the architecture of multiple routes through a network. Packet sequence ordering is supported in higher level protocols such as TCP/IP.

Warning!

For packet mode transmission, the Ethernet subsystem forwards a packet as it is delivered from the EIU. This ordering is dependent upon not only the order that packets are input to their respective NAPs, but also on the buffering within the NAP, NoC and EIU. As a result, it is not possible to implement transmit packet ordering in packet mode.

The choice as to which mode is most suitable will be based on many factors. These include the ease of interfacing to the 1024-bits of Quad Segmented mode, or the requirement for packet ordering performed at a hardware level. The user is directed to investigate the two approaches as implemented within the Speedster7t Ethernet Reference Designs

Configuration

The EIU configuration is controlled by parameters set on each NAP within the user design. These parameters set the NAP column location, channel speed, channel mode (packet or QSI), etc. These parameters are then used by both the simulation environment and ACE to ensure the EIU is correctly configured.

NAP Data Connections

The data streams to and from the Ethernet MACs are interfaced to the user design with the use of NAPs. The specific NAP component is an ACX_NAP_VERTICAL, full details of which can be found in the *Speedster7t IP Component Library User Guide* (UG086). When connected to the EIU (see page 15), the NAP connections are listed below.

Table 4: *Ethernet NAP Data Ports*

Name	Direction	Description	Ethernet NAP Usage
rstn	Input	Asynchronous reset input. This signal resets the NAP interface. This signal does not affect the NoC.	Same.
output_rstn	Output	(Do not use) Reset output from NAP to fabric logic. Intended for use with partial reconfiguration. Signal controlled by write to configuration space. Currently signal fixed to 1'b0.	
clk	Input	All operations are fully synchronous and occur upon the active edge of the clk input.	
tx_ready	Output	Asserted high when the NAP can accept data.	Same. Used to flow control transmit data.
tx_valid	Input	Assert high to issue a word of data to the NAP.	
tx_dest[3:0]	Input	The 4-bit destination ID of the row that this word of data should be sent to.	Must be set to 4'hf (EIU reserved address)
tx_sop	Input	Start of packet and end of packet indicators. These values are sent unmodified to the destination.	Same.
tx_eop			
tx_data[292:0]	Input	Data transmitted to the destination node.	tx_data[255:0] = Data
			tx_data[260:256] = Mod. Only used when `tx_eop` is asserted

Name	Direction	Description	Ethernet NAP Usage
tx_data[292:0]	Input	Data transmitted to the destination node.	tx_data[290:261] = Transmit flags or timestamp (see table below)
			tx_data[292:291] = Unused
rx_ready	Input	Asserted high by user logic to indicate it is ready to receive data.	Same. Used to flow control received data.
rx_valid	Output	Asserted high with each valid `rx_data` word.	
rx_src[3:0]	Output	The 4-bit transmission source ID indicating the row that originated the data.	Will always be 4'f , indicating packet is from the EIU
rx_sop	Output	Start of packet and end of packet indicators. These values are unmodified from the source.	Same.
rx_eop			
rx_data[292:0]	Output	Received data.	rx_data[255:0] = Data
			rx_data[260:256] = Mod. Only valid when `rx_eop` is asserted
			rx_data[290:261] = Receive flags or timestamp (see table below)
			rx_data[292:291] = Unused

Timestamp

For both transmit and receive, when SoP is asserted, the flag field, data[290:261], is set to equal the timestamp value:

- For transmission the user design can generate a free running counter and insert this value into the flag field aligned with `tx_sop`. This value can then be inserted into the outgoing Ethernet packet by enabling the appropriate timestamp mode.
- For reception the timestamp field is a free running value, updated by a 1 GHz clock within the Ethernet interface system. This value represents the time at which the packet was output from the MAC to the EIU (see page 15).

For full details of how the timestamp fields are generated and used, including IEEE-1588 1-Step updating, refer to Speedster7t Ethernet IEEE Timestamping. (see page 36)

Transmit Flags

In addition to the timestamp, during the remaining packet period, the flag field can contain a number of important flags to indicate packet status.

Table 5: *Ethernet NAP Transmit Flags*

Slice	Direction	Flag Name	Description
Flag[16:0]	Output	ID	One-step update control vector
Flag[17]	Output	Frame	Set to indicate IEEE 1588 event frame
Flag[18]	Output	Error	Frame error
Flag[19]	Output	CRC	CRC indicator
Flag[20]	Output	CRC invert	CRC inverted indicator
Flag[21]	Output	CRC overflow	CRC overflow indicator
Flag[22]	Output	Class A	Class A packet
Flag[23]	Output	Class B	Class B packet
Flag[29:24]	Output	Unused	Should be set to 6'b0

Table Note

If the user design does not wish to make use of any of the transmit flags, or the transmit timestamp field, then the appropriate fields should be set to 1'b0.

Table 6: *One-Step Update Control Vector Values*

Bit	Description
[3:0]	Frame Identifier. An arbitrary value that can be used to mark specific frames. This value is available at the transmit status pins when the frame has been transmitted. The usage of the identifier is transparent to the MAC. **Note** The ID vector is only valid when `frame=1'b1`.
4	• 1'b1 – A one-step field update for the frame should occur. All following bits are valid. • 1'b0 – No update occurs, and all following bits have no relevance.

Bit	Description
5	• 1'b1 – Add the peer_delay value to the correction field in addition to the transient time update. This setting is required for peer-to-peer transparent clocks. • 1'b0 – Only the transient time will be added.
6	• 1'b1 – Indicates that the 1588v2 message is using UDP/IP and the UDP checksum needs to be corrected. This setting modifies the last two bytes of the payload (two bytes before frame CRC). • 1'b0 – Indicates that no UDP checksum update should be done. See Speedster7t Ethernet IEEE Timestamping (see page 36) on usage and limitations of this function.
[14:7]	Start offset from begin of frame where the field to update is found (index to most significant byte). An 8-bit value given in steps of byte. The offset must respect all MAC headers, VLAN tags and other protocol headers accordingly. Offset Examples: • Ethernet L2 frame: 22 (14:MAC Header+8 correction field offset in header). • UDP/IPv4 frame: 50 (14:MAC header, 20: IPv4 header, no options; 8:UDP header; 8:correction field offset). • UDP/IPv6 frame: 70 (40 for IPv6 header instead 20 of IPv4). **Note** • The minimum offset for field update is 16. • VLAN tags require adding +4 accordingly.
15	Perform 64-bit seconds/nanoseconds time field update:. • 1'b1 – Bits 5 and 6 cannot be used and must be 0. The offset defines the start of a 64-bit field (byte granularity) within the frame. The field is updated with 8 bytes from `TX timestamp`. The first byte of the field is the most significant). The minimum offset for 64-bit field update is 16. • 1'b0 – No update will occur. **Note** This function sets the field, overwriting the previous frame contents.
16	Perform 48-bit correction field update from delta value (see pins). • 1'b1 – The update adds the application provided delta value to the current correction field value extracted from the frame. Bit 5 cannot be used and must be 0. • 1'b0 – No update will occur

Receive Flags

In addition to the timestamp, during the remaining packet period, the received flag field can contain a number of important flags to indicate packet status.

Table 7: *Ethernet NAP Received Flags*

Slice	Direction	Flag Name	Description
Flag[0]	Input	Error	Frame error indicator.
Flag[8:1]	Input	Error Status	Frame error status. Indicates the type of frame error.
Flag[13:9]	Input	Sequence ID	Frame sequence indicator. Details the order that the packet was output from the MAC. Used particularly in packet mode to reassemble original packet sequence.
Flag[29:14]	Input	Unused	Set to 16'b0.

Receive Sequence ID

The receive sequence ID is a 5-bit count which indicates the order in which packets were transmitted from the EIU (see page 15):

- For a single stream (all data rates up to and including 100G), a continuous count of receive sequence IDs will be received at the single NAP.
- For Quad Segmented Mode (see page) (200 and 400G), the same `sequence ID` will be present on all segments of the same packet.
- For Packet Mode (see page) (200 and 400G), the next `sequence ID` in order could be presented at any of the four possible NAP endpoints. If receive packet ordering is required, then each NAP must be checked to locate the next packet in sequence.

Transmit Clock Frequencies

At all data rates, the complete Ethernet transmission subsystem must ensure that an Ethernet packet is transmitted as a contiguous whole; no part of the system must be allowed to run empty between a start of packet (SoP) and an end of packet (EoP). This restriction places differing requirements on the NAPs based on the selected data rate.

10G/25G/50G/100G

In these modes the EIU is operating with no buffering; therefore, it can be considered that the packet is transmitted directly from the NAP to the MAC. In this case, the NAP must operate at a sufficient frequency that matches the required line rate in order, that when the longest potential frame is transmitted, the whole frame can be sent within the required time.

Assuming a system needs to support jumbo frames with a NAP data width of 256-bits, the following transmit frequencies are required.

Table 8: *NAP Lower Data Rate Transmit Minimum Frequencies*

Data Rate (Gbps)	Frequency (MHz)
10	40

Data Rate (Gbps)	Frequency (MHz)
25	98
50	195
100	390

Note

The transmit frequencies above will only achieve the full stated line rate if all packets are of a length that fits the NAP data width exactly; hence, all packet lengths are a multiple of 32 bytes. If packets of varying lengths are used, then the transmit frequencies need to increase to support the loss in bandwidth caused by having unused data bytes in the last word of a packet. For example, if a packet of 1488 bytes is transmitted (46 full words of 256-bits or 32 bytes, followed by one word of 16 bytes) then the 47 transmit cycles delivers 46.5 cycles of data. Hence, to achieve full line rate, the transmit frequency needs to be increased by a ratio of 47/46.5.

200G/400G

For these line rates the EIU operates in a store-and forward mode, as it has to process each packet based on the stream configuration of quad segmented or packet mode. Therefore, there are no requirements on the transmit frequency. The NAP can transmit at any frequency as the EIU will wait until it has received a whole packet, before transferring that packet to the MAC.

Receive Clock Frequencies

For received packets, there are the same requirements that a packet must not overflow during any stage as it is input to the MAC, and sent via the EIU to a NAP.

10G/25G/50G/100G

For these data rates the EIU is operating in transparent mode — a packet traverses directly from MAC to NAP. As the receive path is designed for packets of any length, including worst-case lengths of $n \times 32 + 1$ bytes, the receive operating frequency must be high enough to support these worst-case scenarios.

Table 9: ***NAP Lower Data Rate Receive Minimum Frequencies***

Data Rate (Gbps)	Frequency (MHz)
10	57
25	143
50	254
100	507

200G/400G

At these data rates, the EIU will transmit packets direct to the NAP. The NAP receive clock rate must be sufficient to match the EIU transmission rate

Table 10: ***NAP Higher Data Rate Receive Minimum Frequencies***

Data Rat0 (Gbps)	Frequency (MHz)
200	254
400	507

Combined TX and RX Frequencies

Although it is possible to configure the NAPs as either transmit or receive only, in most scenarios it is expected that the same NAP will be used for both transmit and receive. In these configurations the higher of the transmit or receive clock frequency will need to be met for the NAP single clock.

Transmit Rate Limiting

In the case where a NAP is both receiving and transmitting, the operating frequency will then be high enough to support worst-case length packets. However, with these higher frequencies, it is also possible that the NAP can exceed the transmit bandwidth, leading to overflows and data loss. For example, for a 100G configuration with a NAP frequency set at 507 MHz, each NAP has a theoretical bandwidth of 130 Gbps, which is in excess of what the EIU would support from a single NAP. Therefore, it is necessary to implement transmission rate limiting for an Ethernet NAP to ensure that the overall data rate is not exceeded.

In the example above, the maximum of 130 Gbps is 13/10 faster than the MAC and EIU can sustain. Therefore, in the Speedster7t Ethernet Reference Designs a transmit rate limiter control block is included which monitors the rate and maintains a 10/13 ratio maximum duty cycle for all transmissions. A version of this block or an equivalent should be used to ensure the correct transmit data rate is maintained.

> **Note**
>
> When using lower data rates with the EIU operating in transparent mode, it is suggested that any transmit flow control is performed between packets, rather than breaking up a single packet transmission. However, if very large packets are used, it may be necessary to insert flow control breaks at intervals within the packet.

Receive Flow Control

Similarly to transmission, the reception of Ethernet traffic should not rely on buffering within the EIU or NAP. Therefore, any design should be structured to receive the full packet from a NAP without de-assertion of the receive ready. In the Speedster7t Ethernet Reference Designs, where this receive flow control could occur, a FIFO is connected to the NAP to ensure that full packets can be received uninterrupted. Receive flow control is then enacted on the FIFO output as opposed to the NAP output.

The receive FIFO can be composed of either four Speedster7t LRAM2K_FIFO (each set to 72-bit width each) or two Speedcore7t BRAM72K_FIFO (set to 144-bit width each).

Flow Control and Status Signals

The flow control and status signals have a logical and consistent naming scheme, consisting of `<prefix>_<mac_identifier>_function`. The details of <prefix> and <mac_identifier> are listed below.

Prefix

Configuration of the Ethernet Interface is performed using ACE I/O Designer, as detailed in Ethernet IP Software Support in ACE (see page 43). Each Ethernet Interface is named during generation. The Ethernet Interface name is prefixed to each signal name to distinguish different Ethernet Interfaces and to aid users in having separate logical names for each instance. In the tables below, this prefix is shown as `<prefix>_` on each signal name.

MAC Identifier

The control and status lines use identifiers to logically group signals from the same MAC:

- `m[1:0]` is for 400G/200G MAC
- `mq0` is for QUAD0 MAC
- `mq1` is for QUAD1 MAC

This identifier follows the prefix field in the signal name table below.

Table 11: *Ethernet Controller Direct Connect Interface*

Pin Name(1,2)	Direction	Width	Comments	Clock	Raw Mode = 0 Data Going to Ethernet
Quad MAC					
<prefix>_mq<n>_tx_hold_req	Input	4	Per channel 100G MACs. Holds and preempts if needed the pMAC (optional, e.g., for test/debug or if higher layer function anticipates an eMAC frame being written soon and wants to prepare the MAC instead having the MAC doing it automatically from the eMAC FIFO being non-empty).	Synchronous to (Application0 Clock)/2	mq<n>_tx_hold_req
<prefix>_mq<n>_lpi_txhold	Input	4	Per channel 100G MACs. Prevents MAC transmit from transmitting a frame even if data is stored in FIFO.	Asynchronous Input Synchronized internally on Reference Clock	mq<n>_lpi_txhold
<prefix>_mq<n>_time_1ms_tgl	Input	4	Per MAC channel 1 ms time base. It is required for internal timing functions. The signal must toggle its value every 1 ms.	Asynchronous Input	mq<n>_time_1ms_tgl
<prefix>_mq<n>_mac_stop_tx	Input	4	Per channel control of MAC transmit. For each lane, when it's respective input is asserted (1'b1), the MAC transmit state machine stops after any ongoing frame has been sent completely (i.e., does not corrupt outgoing frames). If further frames are available in the transmit FIFO the MAC will not begin transmitting them until the respective mac_stop_tx is de-asserted (1'b0) again.	Synchronous to (ref_clk)/2	mq<n>_mac_stop_tx[3:0]
<prefix>_mq<n>_emac_xoff_gen	Input	32	Transmit flow control generate (8 bits per channel) to eMAC /pMAC. When PFC pause mode is enabled, an 8-bit input		mq<n>_emac_xoff_gen[31:0]

Pin Name[1,2]	Direction	Width	Comments	Clock	Raw Mode = 0 Data Going to Ethernet
<prefix>_mq<n>_pmac_xoff_gen	Input	32	vector is used to signal the creation of PFC control frames. When link pause mode is enabled, bit 0 (per channel, i.e., bits 0,8,16,24) is used only.	Asynchronous input synchronized to ref_clk	mq<n>_pmac_xoff_gen[31:0]
<prefix>_mq<n>_mac_peer_delay	Input	120	A peer delay value that can be added to the correction field for all one-step updates (30 bits per channel). Must be wired to 0 if unused.	Synchronous to ref_clk	mq<n>_mac_peer_delay_val[119:0]
<prefix>_mq<n>_mac_peer_delay_val	Input	4	Per channel valid strobe for mac_peer_delay (1 bit per channel). Must assert for 1 ref_clk clock cycle when the peer delay was updated to write the mac_peer_delay value to the MAC internal register. Once the value has been written (i.e., peer_delay_val 0 again) the peer_delay input is no longer relevant and can have arbitrary data. Must be wired to 0 if unused.	Synchronous to (ref_clk)/2.	mq<n>_mac_peer_delay_val[3:0]
<prefix>_mq<n>_pmac_pause_on	Output	32	Transmit paused/class congestion Indication (8-bit value per channel) from eMAC/pMAC. Bit 0 (per channel, i.e., bits 0,8,16,24) is also used to indicate link pause. When asserted to '1' indicates a running pause counter that has been started because a Xoff frame (pause/PFC) was received. In link pause mode, the transmitter is also stopped when the command_config configuration bit PAUSE_PFC_COMP is not set. When the PAUSE_PFC_COMP bit is set, the transmitter will not be stopped and it is the responsibility of the application to assert mac_stop_tx to implement proper flow control.	Synchronous to ref_clk/2	mq<n>_pmac_pause_on[31:0]
<prefix>_mq<n>_emac_pause_on	Output	32			mq<n>_emac_pause_on[31:0]
<prefix>_mq<n>_pmac_pause_en	Output	4	General-purpose indication that the eMAC/pMAC is configured to react on pause frames (1-bit per channel). It is a direct result of the inverted COMMAND_CONFIG (PAUSE_IGNORE) control bit.	Synchronous to reg_clk	mq<n>_pmac_pause_en[3:0]
<prefix>_mq<n>_emac_pause_en	Output	4			mq<n>_emac_pause_en[3:0]
<prefix>_mq<n>_pmac_enable	Output	4	General-purpose indication that the eMAC/pMAC datapaths have been enabled. It is a direct result of both the COMMAND_CONFIG(TX_EN & RX_EN) control bits. Per channel from 100G MACs. Can be left unconnected if not used. **Note** A corresponding signal from the 200G MAC does not exist.	Synchronous to reg_clk	mq<n>_pmac_enable[3:0]
<prefix>_mq<n>_emac_enable	Output	4			mq<n>_emac_enable[3:0]
<prefix>_mq<n>_mac_tx_ovr_err	Output	4	FIFO overflow truncation error indication. Asserts when the FIFO write control logic had to truncate a frame as either the ff_tx_rdy deassertion was not respected by the application, or the frame transferred is larger than the FIFO in store-and-forward mode of operation.	Synchronous to ref_clk/2	mq<n>_mac_tx_ovr_err[3:0]
<prefix>_mq<n>_ffp_tx_ovr	Output	4			mq<n>_ffp_tx_ovr[3:0]
<prefix>_mq<n>_ffe_tx_ovr	Output	4			mq<n>_ffe_tx_ovr[3:0]
<prefix>_mq<n>_mac_tx_underflow	Output	4	Transmit FIFO became empty during transmission. A frame has been corrupted.	Synchronous to ref_clk/2	mq<n>_mac_tx_underflow[3:0]
			Transmit FIFO Empty Indication from pMAC. When set to 1, indicates that the transmit FIFO is empty. When set to 0, indicates transmit FIFO has data. When the Quad operates		

Pin Name[1,2]	Direction	Width	Comments	Clock	Raw Mode = 0 Data Going to Ethernet
<prefix>_mq<n>_pmac_tx_empty	Output	4	in 200G, pmac_tx_empty[0] indicates empty for the 200G MAC FIFO.	Synchronous to ref_clk/2	mq<n>_pmac_tx_empty[3:0]
<prefix>_mq<n>_emac_tx_empty	Output	4	Transmit FIFO Empty Indication from eMAC. When set to 1, indicates that the transmit FIFO is empty. When set to 0, indicates transmit FIFO has data. When the Quad operates in 200G, emac_tx_empty is unused/not relevant.		mq<n>_emac_tx_empty[3:0]
<prefix>_mq<n>_mac_tx_isidle	Output	4	Transmit datapath is not transmitting when 1. Will toggle during normal operation, but is not necessarily frame accurate (i.e., may not always de-assert during minimum IPG).	Synchronous to ref_clk/2	mq<n>_mac_tx_isidle[3:0]
<prefix>_mq<n>_link_up	Input	4	Per channel indication from the PCS that the link is up. The application must drive this signal from the PCS link_status, and potentially from loc_fault and rem_fault. It is used to detect a link loss in the MAC's transmit merge sub-layer to disable pre-emption.	Synchronous to (ref_clk)/2	mq<n>_link_up[3:0]
<prefix>_mq<n>_mac_tx_ts_val	Output	4	Timestamp valid. Asserted for one clock cycle to indicate that mac_tx_ts_id and mac_tx_tsN are valid. The signal is not asserted for internally generated pause frames.	Synchronous to ref_clk/2	mq<n>_mac_tx_ts_val[3:0]
<prefix>_mq<n>_mac_tx_ts_id	Output	16	Frame identifier return (4-bit value per channel). The value that was provided by the application at ff_tx_id(3:0) for the frame.	Synchronous to ref_clk	mq<n>_mac_tx_ts_id[4*4-1:0]
<prefix>_mq<n>_mac_fault_ored4l	Output	1	Local, remote, and link interruption faults. ORed into a single output.	Synchronous to ref_clk/2	|mq<n>_mac_loc_fault[3:0]) || (|mq<n>_mac_rem_fault[3:0]) || (|mq<n>_mac_li_fault[3:0]
<prefix>_mq<n>_mac_tx_ts	Output	256	Frame timestamp value return (64-bit value per channel). Transmit timestamp value for the frame sent with the frame identifier set on mac_tx_ts_id. Returns the value sampled from mac_frc_i_tx.	Synchronous to ref_clk	mq1_mac_tx_ts
400G/200G MAC[3]					
<prefix>_m80_c<m>_xoff_gen	Input	8	Transmit flow control generate. When PFC pause mode is enabled, an 8-bit input vector is used to signal the creation of PFC control frames. When link pause mode is enabled, bit 0 is used only.	Asynchronous input synchronized to ref_clk	m80_c<m>_xoff_gen[7:0]
<prefix>_m80_c<m>_tx_smhold	Input	1	Instructs the MAC to stop reading further data from the transmit FIFO at the next possible frame boundary.	Synchronous to (ref_clk)/2	m80_c<m>_tx_smhold
<prefix>_m80_c<m>_peer_delay	Input	30	Current value of the link delay measured at the ingress port that receives SYNC messages from a master. This input is a global value which is updated from time to time by the PTP software when implementing peer-to-peer transparent clock systems.	Synchronous to ref_clk	m80_c<m>_peer_delay[29:0]
<prefix>_m80_c<m>_peer_delay_val	Input	1	Indicates validity of peer_delay(). Must be a pulse for one cdmii_txclk cycle whenever the peer_delay() input was updated.	Synchronous to (ref_clk)/2.	m80_c<m>_peer_delay_val
			Transmit paused/class congestion indication, one bit per priority class. When asserted to '1' indicates a running		

Pin Name[1,2]	Direction	Width	Comments	Clock	Raw Mode = 0 Data Going to Ethernet
<prefix>_m80_c<m>_pause_on	Output	8	pause counter has been started because an Xoff frame (pause/PFC) was received. In link pause mode, the transmitter is also stopped (if not disabled by COMMAND_CONFIG.PAUSE_PFC_COMP configuration setting).	Synchronous to ref_clk/2	m80_c<m>_pause_on[7:0]
<prefix>_m80_c<m>_tx_ovr_err	Output	1	FIFO overflow truncation error indication. Asserts when the FIFO write control logic had to truncate a frame as either the ff_tx_rdy de-assertion was not respected by the application, or the frame transferred is larger than the FIFO in store-and-forward mode of operation.	Synchronous to ref_clk/2	m80_c<m>_tx_ovr_err
<prefix>_m80_c<m>_tx_underflow	Output	1	Transmit FIFO became empty during transmission. A frame has been corrupted.	Synchronous to ref_clk/2	m80_c<m>_tx_underflow
<prefix>_m80_c<m>_fault	Output	1	Rx receives a local or remote fault.	Synchronous to ref_clk/2	m80_c<m>_loc_fault || m80_c<m>_rem_fault
<prefix>_m80_c<m>_tx_ts	Output	64	Frame timestamp value return (64-bit value per channel). Transmit timestamp value for the frame sent with the frame identifier set on mac_tx_ts_id. Returns the value sampled from mac_frc_i_tx.	Synchronous to ref_clk	m80_c<m>_tx_ts[63:0]
<prefix>_m80_c<m>_tx_ts_id	Output	4	Frame identifier return (4-bit value per channel). The value that was provided by the application at ff_tx_id(3:0) for the frame.	Synchronous to ref_clk	m80_c<m>_tx_ts_id[3:0]
<prefix>_m80_c<m>_tx_ts_val	Output	1	Timestamp valid. Asserted for one clock cycle to indicate that mac_tx_ts_id and mac_tx_tsN are valid. The signal is not asserted for internally generated pause frames.	Synchronous to ref_clk/2	m80_c<m>_tx_ts_val
<prefix>_m80_c<m>_tx_empty	Output	1	Transmit FIFO empty.	Synchronous to ref_clk/2	m80_c<m>_tx_empty
<prefix>_m80_c<m>_tx_isidle	Output	1	Indicates (when 1) transmit state machine is not transmitting a frame currently.	Synchronous to ref_clk/2	m80_c<m>_tx_isidle
<prefix>_m80_c<m>_frm_drop	Output	1	Frame drop indication. A frame drop occurs when a frame contains less than 64 data bytes or the application was not ready to accept a frame (ff_rx_rdy=0) at begin of the frame. In both cases, the frame is not delivered to the application.	Synchronous to ref_clk/2	m80_c<m>_frm_drop
Common					
<prefix>_ref_clock_divby2	Output	1	Reference clock divided by 2		
<prefix>_m0_ff_clk_divby2	Output	1	Application0 clock divided by 2		
<prefix>_m1_ff_clk_divby2	Output	1	Application1 clock divided by 2		

Table Notes

1. The Ethernet interface name (shown as `<prefix>_`) is prefixed to each signal name, to distinguish different Ethernet interfaces and to aid users in having logical names for each interface.
2. The variable <n> indicates the MAC quad number: 0 for QUAD0 MAC, and 1 for QUAD1 MAC.
3. The variable <m> indicates the channel number: 0 or 1.

Chapter - 3: NoC Connectivity

As previously detailed, the data from the Ethernet IP is delivered via the EIU and NOC to the NAPs in data streaming mode.

The Ethernet subsystem connects directly to specific columns on the NoC and can communicate to FPGA fabric logic connected to vertical NAPs along those specific columns using Ethernet packets. Each Ethernet subsystem has two dedicated columns and can send transactions to NAPs placed only on those two specific columns. The table below lists the specific columns connected to the Ethernet subsystems.

Table 12: ***NoC Columns for Ethernet Subsystems***

Ethernet Subsystem location	Ethernet Subsystem 0 (West)	Ethernet Subsystem 1 (East)
NoC Column 1	1	4
NoC Column 2	2	5

Table Note

NoC Columns are numbered 1 at the west-most column and increment going east.

There are a few modes available, depending on how the user wishes to handle the Ethernet packets in the FPGA fabric. For interfaces using 100GE or slower, the Ethernet sends 256-bit packets down the columns directly to NAPs. For interfaces running 200GE or 400GE, there are two modes to choose from: packet mode or quad-segmented mode.

Packet Mode

The NoC rearranges the 1024-bit data bus into four narrower data paths, funneling a separate packet to each of four NAPs and splitting the full 1024-bit data bus into four 256-bit (32-byte) data paths. This solution results in less congestion in the fabric because the user logic can reside in four separate engines distributed down the NoC columns rather than a single large engine immediately next to the Ethernet subsystem. This mode also reduces the needed frequency in the FPGA fabric design and makes the design easier because each NAP can have its own individual packet processing engine.

Packet mode can result in larger latency as each packet can take more cycles to transfer. Importantly, packets can arrive out of order, with the NoC sending a sequence number along with each packet. The user logic is responsible for reordering the packets (if necessary), in order to retrieve the original data sequence. The figure below shows how the Ethernet subsystem data bus is rearranged into four separate 256-bit wide data buses. Each packet can take multiple cycles to complete.

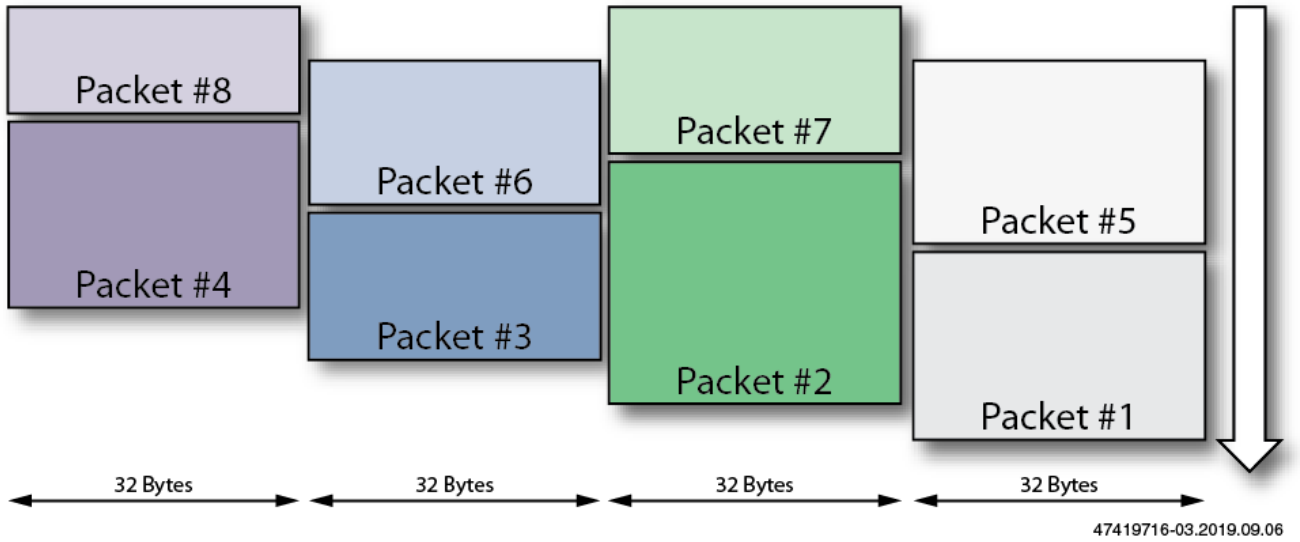


Figure 3: ***Data Bus Rearrangement for Packet Mode***

The four packets shown above are sent to four separate NAPs distributed down the designated NoC columns. Each NAP communicates to an individual packet processing engine. This arrangement allows each NAP and processing engine to be run at a lower frequency than that required of a single processing engine with the full 1024-bit bus, thus simplifying the system design. For example, a single processing engine for a 400GE solution requires a 1024-bit bus running at about 728 MHz, whereas the packet mode for 400GE uses four NAPs and requires four 256-bit buses running at 507 MHz. The NoC automatically handles the load balancing, sending the next available packet to the next free NAP. For more details on Ethernet packet mode, refer to the *Speedster7t Ethernet User Guide (see page 6)* (UG097).

In the figure below, the four NAPs are distributed in different locations along two columns. The specific placement of the NAPs is a design choice. It is equally possible to have all four NAPs be located on a single column, or grouped closer together.

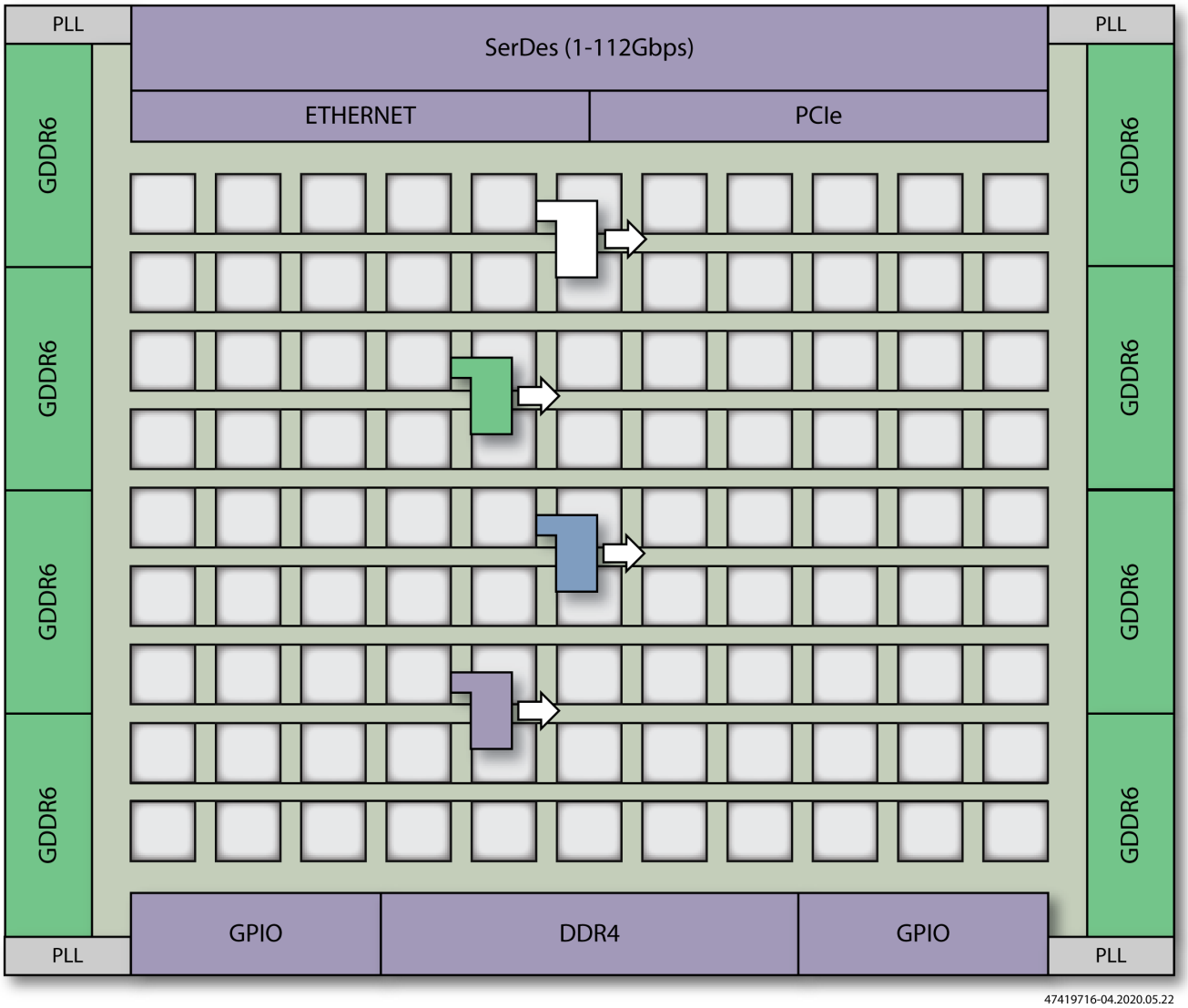


Figure 4: ***Ethernet Packet Mode on the NoC***

Quad-Segmented Mode

In quad-segmented mode, the NoC sends a 1024-bit bus that is segmented across four NAPs. This mode makes the user logic a little more complex as the design logically is one large packet processing engine distributed across the four NAP locations. This mode does guarantee in-order packet arrival, and larger packets arrive with less latency than in packet mode described above. Because the bus is segmented, packets can potentially start at any of the four NAPs, and up to two packets can arrive in a single fabric clock cycle.

Similar to the packet mode above, the FPGA logic can be spread across the space of four NAPs on the designated columns, rather than having to be placed immediately next to the Ethernet subsystem. This arrangement helps ease congestion, and because the design can be split across four NAPs, the frequency can be reduced similar to the packet mode. For example, a single processing engine for a 400GE solution requires a 1024-bit bus running at 728 MHz, whereas the quad-segmented mode for 400GE uses four NAPs and requires four 256-bit buses running at 507 MHz. The figure below shows how the packets are arranged and segmented for the quad-segmented mode.

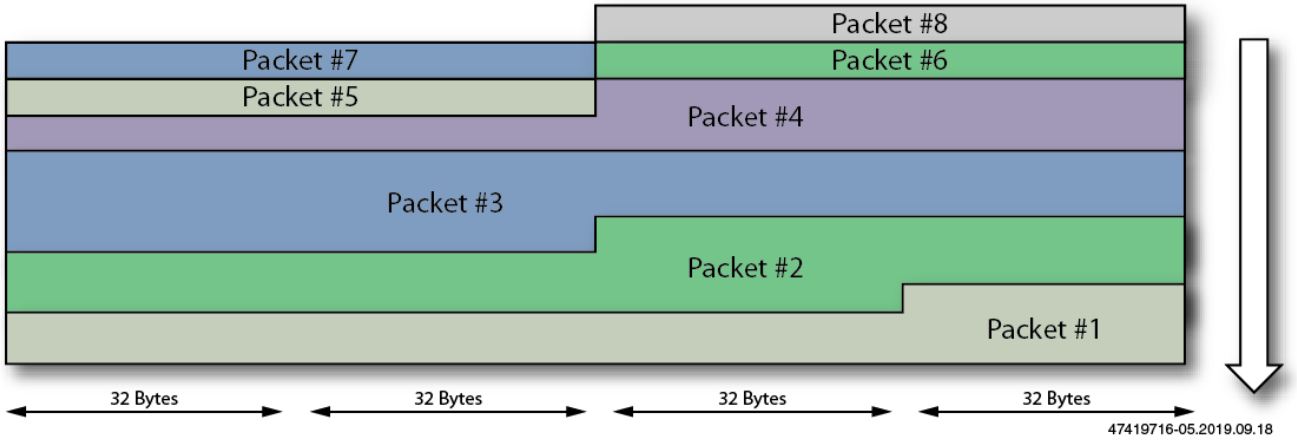


Figure 5: ***Packet Segmentation for Quad-Segmented Mode***

Each packet is distributed across four NAPs located on the designated columns of the NoC. Each 32-byte segment is dedicated to a specific NAP in the group of four. The packet processing engine should be located close to the four NAPs. The figure below shows the four NAPs distributed in two columns, but placed close together. The specific placement of the NAPs is a design choice. It is equally possible to have all four NAPs be located on a single column, or grouped farther apart.

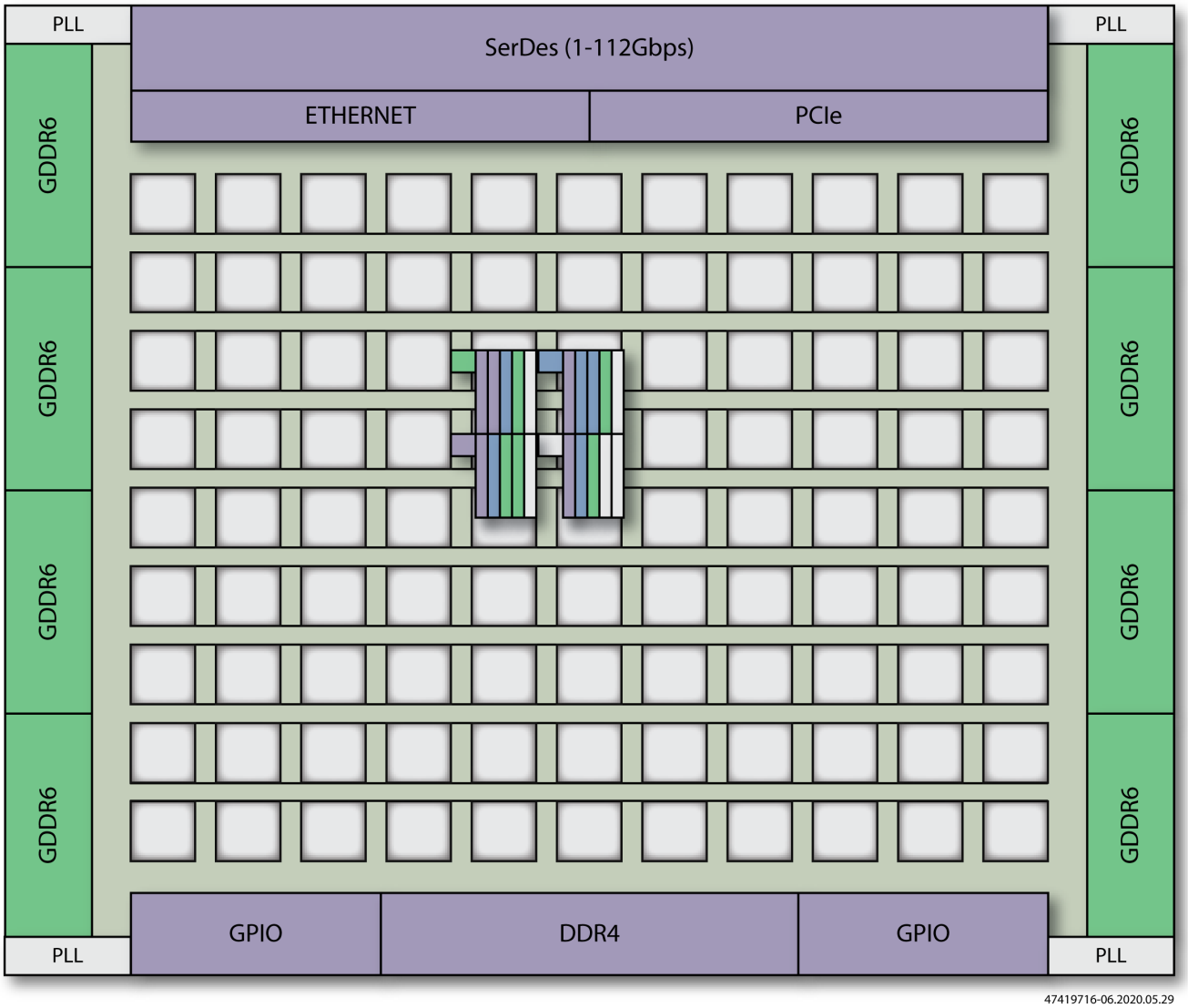
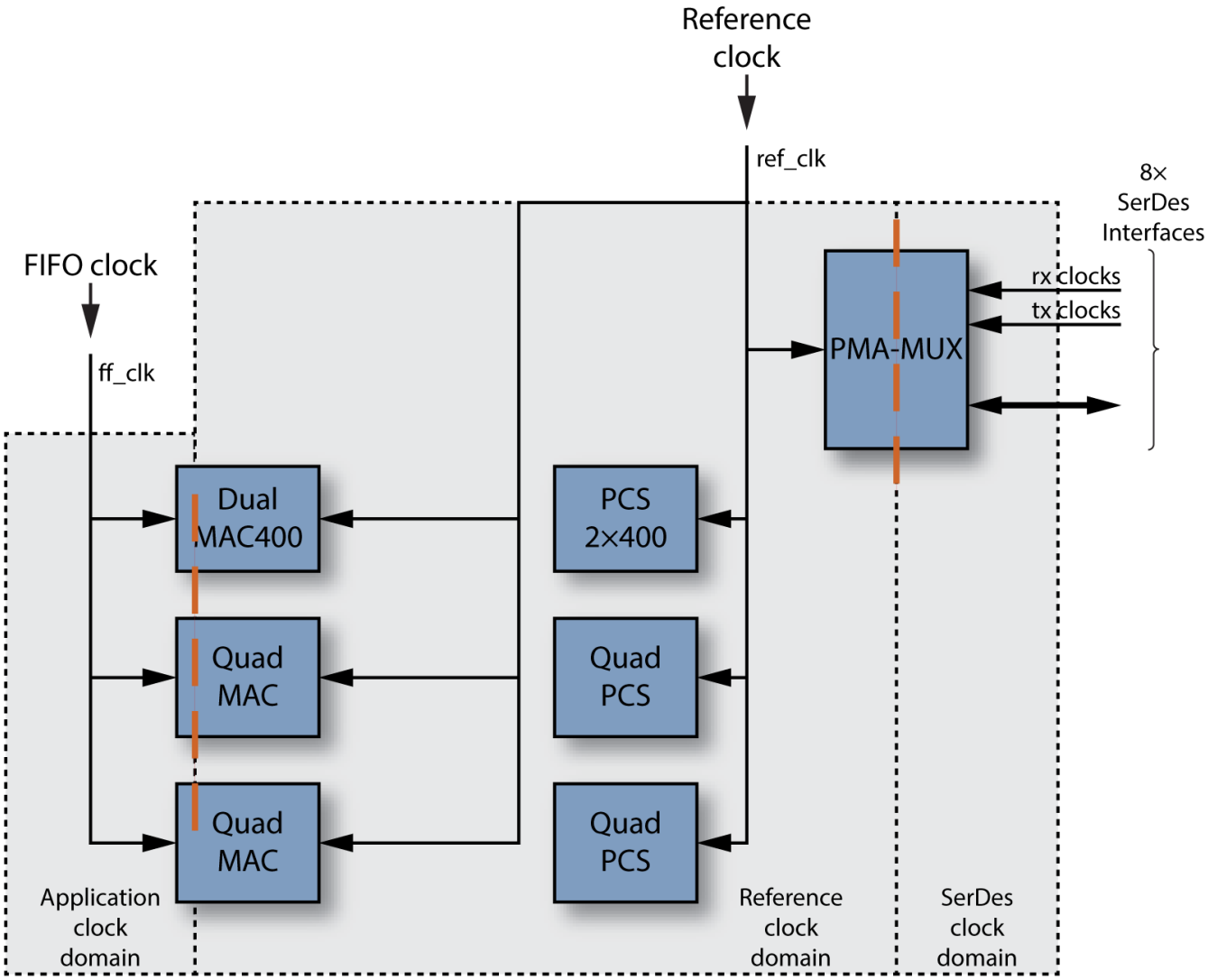


Figure 6: ***Quad -segmented Mode on the NoC***

Chapter - 4: Ethernet Clocks

Clock Domains

Within an Ethernet Interface there are four clock domains (two `ff_clk` domains) as shown in the diagram below.



47420509-01.2020.03.04

Figure 7: *Ethernet Interface Clock Domains*

The clocks driving these clock domains are as follows

- `ff_tx_clk` – FIFO transmit clock. This clock is driven from the EIU and NoC and is used to write the data into the transmit FIFOs. This clock must be configured to be fast enough to ensure that it can supply data at the required data rate to avoid the transmit FIFO running empty during packet transmission. The

minimum frequencies for `ff_tx_clk` based on the data rate are detailed in Table: Reference and FIFO Clock Frequencies (see page 34).

- `ff_rx_clk` – FIFO receive clock. This clock is driven from the EIU and NoC and is used to receive the data from the receive FIFOs. This clock must be configured to be fast enough to ensure that it can receive data at the required data rate in order to avoid the receive FIFO overflowing and dropping packets. The minimum frequencies for `ff_rx_clk` based on the data rate are detailed in Table: Reference and FIFO Clock Frequencies (see page 34).
- `ref_clk` – This clock drives the internal logic of the MAC and PCS. This clock domain interfaces to the FIFO clock domains from the EIU and NoC, through to the PMA interface driving the SerDes. This clock must be configured to be fast enough to allow the PCS and MAC to process the packets. The minimum frequencies for `ref_clk` based on the data rate are detailed in Table: Reference and FIFO Clock Frequencies (see page 34).
- `serdes_clk` – This clock drives the SerDes and is related to the SerDes line rate. The SerDes clock requirements are detailed in Table: SerDes Clock Frequencies (see page 35).

The four clock domains operate independently, using asynchronous FIFOs to decouple the data flows between them. There are no frequency or phase requirements between the clocks; however, they must meet the minimum frequencies detailed in the tables below. All the clocks are specified using ACE I/O Designer tool, as detailed in Ethernet IP Software Support in ACE. (see page 43)

Clock Frequencies

Reference and FIFO Clocks

The frequency requirement for the reference clock `ref_clk` , and the user FIFO clocks, `ff_tx_clk` and `ff_rx_clk`, depend on the mode of operation. The highest mode active in any of the channels defines the minimum requirement. The clock frequencies ranges are detailed in the table below.

Table 13: *Reference and FIFO Clock Frequencies*

Mode	ref_clk		ff_clk (Recommended)	
	Min	MAX	Min Theoretical	Min Recommended
10G	323 MHz	900 MHz	59 MHz	160 MHz
25G	782 MHz (25G no RSFEC) 831 MHz (25G with RSFEC)	900 MHz	147 MHz	210 MHz
50G	831 MHz	900 MHz	293 MHz	
100G	831 MHz	900 MHz	586 MHz	
200G	831 MHz	900 MHz	392 MHz	
400G	831 MHz	900 MHz	782 MHz	

Warning!

The given minimum frequencies must be respected. For example, for 25G, 781.25 MHz is not sufficient, it must be 782 MHz or higher.

SerDes Clocks

SerDes clock frequency is directly proportional to the SerDes speed and interface width, as detailed in the table below.

Table 14: *SerDes Clock Frequencies*

SerDes Speed	Active SerDes Width	PMA Interface Frequency
10.3125 Gbps (NRZ)	32/64	32 : 322.265635 MHz 64 : 161.1328125 MHz
25.78125 Gbps (NRZ)	32/64	32 : 805.6640625 MHz 64 : 402.83203125 MHz
26.5625 Gbps (NRZ)	32/64	32 : 830.078125 MHz 64 : 415.0390625 MHz
26.5625 Gbps (PAM4)	32/64	32 : 830.078125 MHz 64 : 415.0390625 MHz
53.125 Gbps (PAM4)	64	830.078125 MHz
106.25 Gbps (PAM4)	128	830.078125 MHz

Selecting Clock Frequencies

The selection of clock frequencies is provided by the ACE I/O Designer tool, as detailed in Ethernet IP Software Support in ACE (see page 43). Upon selecting the desired Ethernet data rates, I/O Designer specifies what the required minimum clock frequencies are. In addition, I/O Designer actively checks that the clock sources are connected to suitable PLLs, and that those PLLs are correctly configured to generate the desired frequencies. Using I/O Designer greatly simplifies the task of ensuring the correct frequencies are defined at the start of a design.

Chapter - 5: IEEE Timestamping

Overview

The MAC Core supports IEEE 1588 receive and transmit timestamping by providing an optional timestamp for every frame and, on transmit, one-step frame update.

Receive Timestamping

When a packet is received, the user design should latch the value of the receive timestamp field. The timestamp is presented with the start-of-packet (SoP) pulse.

Transmit Timestamping

On transmit it is only necessary to time-stamp IEEE 1588 event frames. When the user design is transmitting an IEEE 1588 event frame, the `TX frame` flag must be set; `frame = 1'b1`, and the frame identifier, `TX ID vector`, set to indicate updated control information.

The frame identifier can be used by the application to mark specific frames and then use the returned timestamp for those marked frames as needed (e.g., 1588 event frames). For example, the MAC could store timestamps and generate interrupts only for frames with a specific ID and ignore all timestamps for frames with other IDs.

When frame = 1'b1, the MAC additionally stores the timestamp lower 32-bit for the frame in the register TS_TIMESTAMP. The status bit STATUS(TS_AVAIL) is set to indicate that a new timestamp is available. SA user application (software) would implement a handshaking procedure by setting the transmit frame flag when it transmits a frame that it needs a timestamp for. This application would then wait on the STATUS(TS_AVAIL) status bit to know when the timestamp is available, then read the timestamp from the TS_TIMESTAMP register. This process would be done for all 1588 event frames; other frames will not set the transmit frame flag, and hence, will not interfere with the timestamp capture.

> **Note**
>
> The TS_TIMESTAMP register can store up to 32 bit of a timestamp. The NAP timestamp field is only 30-bits wide.

IEEE 1588 TX One-Step Frame Field Update

Overview

The IEEE 1588 field update functions are controlled by the `TX ID vector`, when the `TX frame` flag is set. The IEEE 1588 functions are selected when `TX ID vector[15] = 1'b0.`

> **Note**
>
> The one-step functions are available only when the control register bit XIF_MODE.ONE_STEP_ENA is set. Otherwise, the one-step controls of the `TX ID vector` have no effect.

IEEE 1588 Message Header Structure

A 1588 message can be transported with various encapsulation protocols for example within an Ethernet frame (L2) payload or an UDP/IP payload. The following table describes the message header, which is common to all 1588 version 2 messages. The MAC allows some flexibility on updating the header allowing it to be compatible with other standards as well.

All fields follow the network byte order which is the first byte transmitted is the most significant byte (leftmost) of a multi-byte field. The following tables provide a short overview. For details of any of the fields please refer to the IEEE 1588 specification document.

Table 15: *IEEE 1588v2 Message Header (PTPv2)*

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
transport specific				messageID				1	0
reserved				versionPTP = 0x2				1	1
messageLength								2	2
domainNumber								1	4
reserved								1	5
flags								2	6
correctionField								8	8
reserved								4	16
sourcePortIdentity								10	20
sequenceID								2	30
control								1	32
logMeanMessageInterval								1	33
additional fields as required								n	34

The type of message is encoded in the field `messageId` as follows (informal only):

Table 16: *PTPv2 Message Type Identification*

messageID	Message Name	Message
0x0	SYNC	Event message
0x1	DELAY_REQ	

messageID	Message Name	Message
0x2	PATH_DELAY_REQ	Event message
0x3	PATH_DELAY_RESP	
0x4 - 0x7		Reserved
0x8	FOLLOW_UP	General message
0x9	DELAY_RESP	
0xa	PATH_DELAY_FOLLOW_UP	
0xb	ANNOUNCE	
0xc	SIGNALING	
0xd	MANAGEMENT	

Correction field updates only occur in event messages (i.e., `messageId` < 4). As additional information, the MAC function does not inspect for these messages as the application must define if it wants an update or not by specifying the correct control bits with the `TX ID vector` when writing the frame to the NAP.

Update Function Overview

The MAC offers to change the correction field of outgoing IEEE 1588 event frames while the frame is being transmitted (one-step). This capability enables its use in transparent clocks where the correction field is updated with the transient time, as well as end-node systems that need to transmit one-step event frames (i.e., omitting the follow-up frames). For both usage variants (transparent or end-node), it is sufficient to update the correction field of the frame (as opposed to updating the `originTime` – see below for a detailed discussion). The `TX ID vector` is used to identify and control the update function.

Warning

As a requirement for one-step support, all timestamps must be given in true nanoseconds values. The user design must use an accurate clock (not 507 MHz) that can give precise nanosecond values. When this requirement is met the correction field update can consider delays up to one second.

Transparent Clock Concepts

Overview

A transparent clock receives frames at a port and forwards them to other ports, updating the frames to corrected for the time spent traversing that network equipment instance. The transient time from reception to transmission needs to be added to the correction field of the outgoing frame to compensate for the delay. The IEEE 1588 standard distinguishes two different modes of operation for transparent clocks: end-to-end and peer-to-peer transparent clocks.

The main difference, with respect to frame update is that an end-to-end clock only adds the transient time to the correction field. In contrast, a peer-to-peer transparent clock additionally needs to add the ingress path delay from the master at the port where the frame was received (link delay on ingress port).

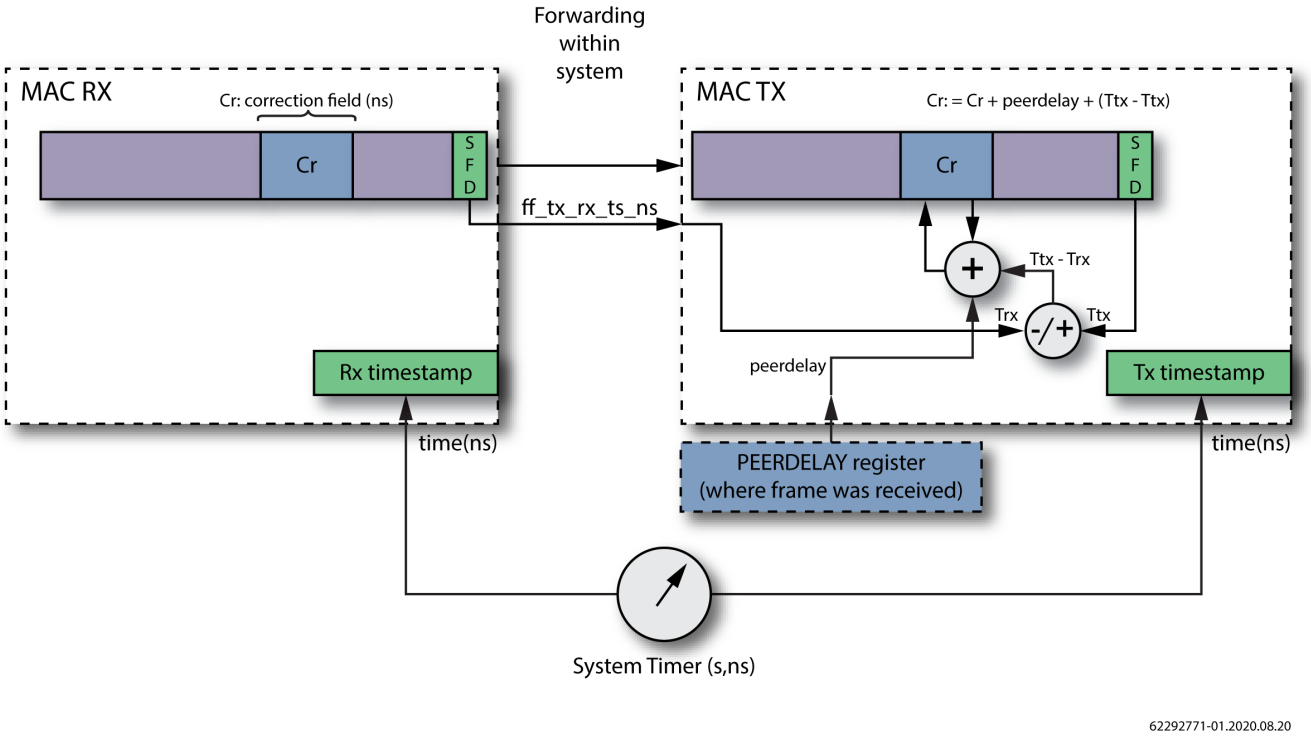


Figure 8: ***Transparent Clock Concepts Overview***

Hence, the designer needs to consider the following system functions and concepts when implementing transparent clocks:

End-to-End Transparent Clock

The receive timestamp (available with every frame at `ff_rx_ts`) needs to be forwarded throughout the system and provided to the `ff_tx_rx_ts_ns()` input together with the frame. At the same time the application must parse the frame to determine if it is a 1588 event frame that needs updating as well as the transport layer to be able to provide this information to the transmit ID vector.

> **Note**
>
> In an end-to-end clock network, the peer_delay() value is not used and should be set to 0.

Peer-to-Peer Transparent Clock

In this mode, the forwarding process operates identically to end-to-end clocks and must forward the received timestamp with the frame to present it to the `TX timestamp` field at the outgoing NAP and parse the frame for controlling the `TX ID vector`. However, in addition to the transient time update, the link delay (peer delay) of the port where the frame was received must be added to the correction field. This update can be performed either by the system (at the receiving port), or alternatively the MAC offers an option to add this value when it updates the correction field.

The link delay is acquired by the PTP software by periodically transmitting PDELAY_REQ messages at the port where the master is connected (it may do it also on all other ports but only the one that connects to the master is relevant). This value is then presented to the top-level pins peer_delay(). As there can only be one master active at a time, a single top-level peer delay value can be given, which is always the one from the port where SYNC messages from the master are received.

> **Note**
>
> For peer-to-peer transparent clocks, the only forwarded message is a SYNC message from the master. All other messages (PDELAY_REQ/RESP) are locally generated and terminated.

Locally Generated Event Messages

When implementing a 1588-capable node, the local system may need to generate event frames on its own. For example, the implementation of peer-to-peer transparent clocks requires transmission of PDELAY_REQ and PDELAY_RESP event messages at every port individually. Implementation of an end-node may need to create DELAY_REQ/DELAY_RESP frames when connected to an end-to-end transparent clock environment.

If any event frame needs a precise `originTime` field within the frame, the following concepts can be used to take advantage of the MAC's one-step update feature.

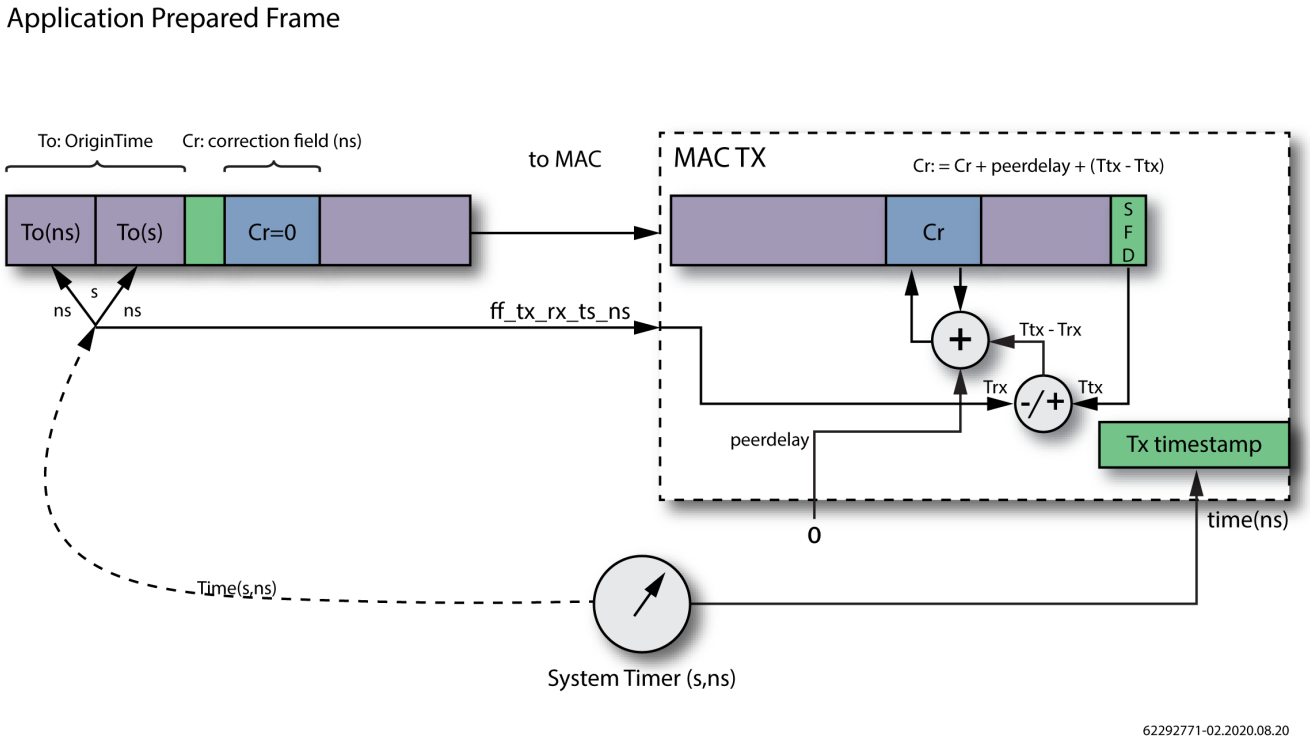


Figure 9: ***One-Step Update Concepts for Locally Generated Event Frames***

Locally generated one-step event frames (i.e., SYNC frames) have their transmission timestamp in the `originTime` field of the message. As the correction field is present in all messages, it is valid to use it for correcting the `originTime` as the receiver always adds it to the `originTime` before processing it (see also IEEE1588 v2, Clause 9.5.9.3). Hence, when the local system generates event frames, it is sufficient to set the `originTime` field of the frame to the current time when the frame is prepared. It then presents the same nanoseconds value that was used to set `originTime` (that is hence also found in the frame) to the TX `timestamp` field with the frame for transmit. The MAC updates the correction field with the difference between the current time and frame generation time, making the final timestamp value for the frame precise.

> **Note**
>
> This correction requires that the message generation (i.e., setting of `originTime` field) and its actual transmission must happen within one second as the correction field update function can only respect latencies up to one second (as it only operates on the nanoseconds field of all values).

Transport Layer Identification

To use one-step updates for a frame, either a parser or another system-level function (e.g., driver) needs to classify the frame and control the MAC's update function accordingly through the `TX ID vector` when it writes the frame into the NAP.

The following are typical examples used for transporting 1588 messages. The interface is generic to support any type of transport protocol. Depending on whether the system role is an endnode or a transparent clock, as well as depending on the type of transport layer used for 1588 messages, various options exist. The following table lists the control bits for the `TX ID vector` for typical examples.

- 1588v2 Layer 2
- 1588v2 UDP/IPv4
- 1588v2 UDP/IPv6
- Other transport protocol

Table 17: *One-Step Transport Layer Identification Examples*

Transport Type	Frame Type	Correction Update Needed Tx ID Vector[4]	Add Peer-Delay Tx ID Vector[5]	Do UDP/IP Checksum Tx ID Vector[6]	Field Offset Tx ID Vector [14:7]	Description
Any	Non-1588	0	0	0	0	Normal traffic
	1588 general message	0	0	0	0	Non-event frames do not need field updates
Layer 2 (frame type 0x88f70)	1588 event message	1	The value is 0 or 1 depending on system role or origination of frame: • 0 – local or e-t-e transparent • 1 – PTP transparent and SYNC frame)	0	14+8=22	Layer 2 event frames either locally generated or forwarded from another port if node is a transparent clock
UDP/IP4		1		1 (do UDP checksum update)	42+8=50	IPv4 without any options
UDP/IP6		1			62+8=70	IPv6 without extension headers
Other (non-IP)		1	System specific	0	<size of headers>	Generic supporting any type of transport layer

Transport Type	Frame Type	Correction Update Needed Tx ID Vector[4]	Add Peer-Delay Tx ID Vector[5]	Do UDP/IP Checksum Tx ID Vector[6]	Field Offset Tx ID Vector [14:7]	Description

Note

It is the responsibility of the application to determine these values. As they need to be available at start-of-frame (SoP), it must be done before the frame can be presented to the NAP.

UDP Payload Checksum Correction Field Update

The Ethernet interface subsystem supports correction of UDP/IP payload checksum by modifying the last two bytes within the payload (the two bytes following 1588 event frame message). The application can request this function by setting the corresponding control bit with the `TX ID vector.` The subsystem modifies the two bytes of the frame to adapt the UDP checksum to compensate for the changes caused by the correction field update (see also IEEE 1588 v2, Annex E).

The UDP checksum update function does not perform a complete UDP checksum calculation. It only modifies the UDP payload to compensate for the changes done by the correction field update. The frame must have had a correct UDP checksum when it was written into the NAP. The function is supported for IEEE 1588 frame update functions only (i.e., `TX ID vector[15] = 1'b0`).

Warning!

The UDP update can work *only* on frames where the Ethernet payload and the modified field begin on a 16-bit boundary and the frame payload has an even number of bytes.

Correction Field Update from Delta Value

When transmit ID vector(16) = 1'b1, the update function only adds a delta value provided on `frc_delta()` to the current correction field value extracted from the frame. The CRC value of the frame is calculated after the update. Optionally a UDP checksum update can be performed on the updated frame when `TX ID vector[6] = 1'b1`.

Generic Transmit One-Step 64-bit Frame Field Overwrite

Alternatively to the IEEE 1588 frame update functions, a 64-bit field overwrite function is available, allowing the update of one 64-bit field within the frame. The overwrite function is selected when `TX ID vector[15]=1'b1` . The difference to the IEEE 1588 update function is that this overwrite function ignores the frame's contents and instead replaces/overwrites the frame contents with the timestamp value. No UDP checksum update is available in this mode.

Chapter - 6: Ethernet IP Support in ACE

Overview

Ethernet Interface IP generation in ACE provides a GUI to generate and integrate the Ethernet Interface instances based on the user specified inputs. The I/O Designer in ACE supports the integration of all the chosen IP for the user design and also allows the user to select the placement and visualize package routing. Once the desired IP is configured via the I/O Designer GUI, the tool generates a bitstream for the entire IP interface which is independent of the bitstream generated for the core fabric. The tool then integrates both these bitstreams into a single configurable bitstream targeting a Speedster7t device.

The following steps provide a brief description on creating an Ethernet IP interface design:

Step 1 – Creating a Project

Create a project in ACE, and then in the 'Project perspective', select the target device **AC7t1500ES0** which ensures that the appropriate IP options are available in the IP Perspective window in ACE.

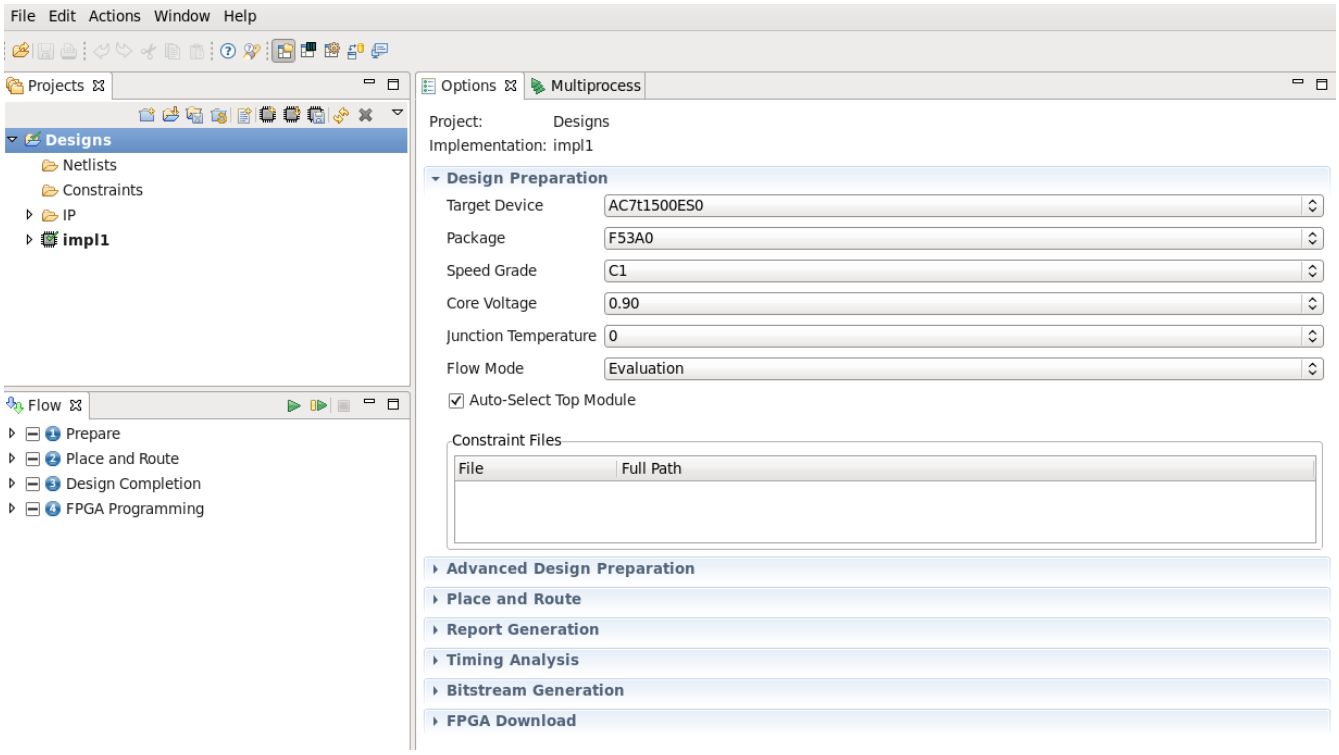


Figure 10: ***Design Preparation Options in the ACE Project***

Step 2 – Configure and Generate ACE IP

System Clock Input

Switch to the 'IP Configuration' perspective and under **Speedster7t** select **IO Ring** then select **Clock I/O Bank.** Select a suitable name for the instance (such as `clock_io.acxip`), and a target directory (the default location is in the ACE working directory; however, all reference designs place these files in a separate `/acxip` directory alongside the `/ace` directory).

In the **Clock I/O Bank** wizard, select the clock source that you require; for this example we will use a single ended reference clock, so will select `clock_io_msio_p`, and will rename it to `sys_clk`. Leave the clock bank placement as CLKIO_NE. In the **IP Problems** tab, ensure that there are no errors or warnings; then select **Generate** to save this `acxip` file, and generate the necessary IP files.

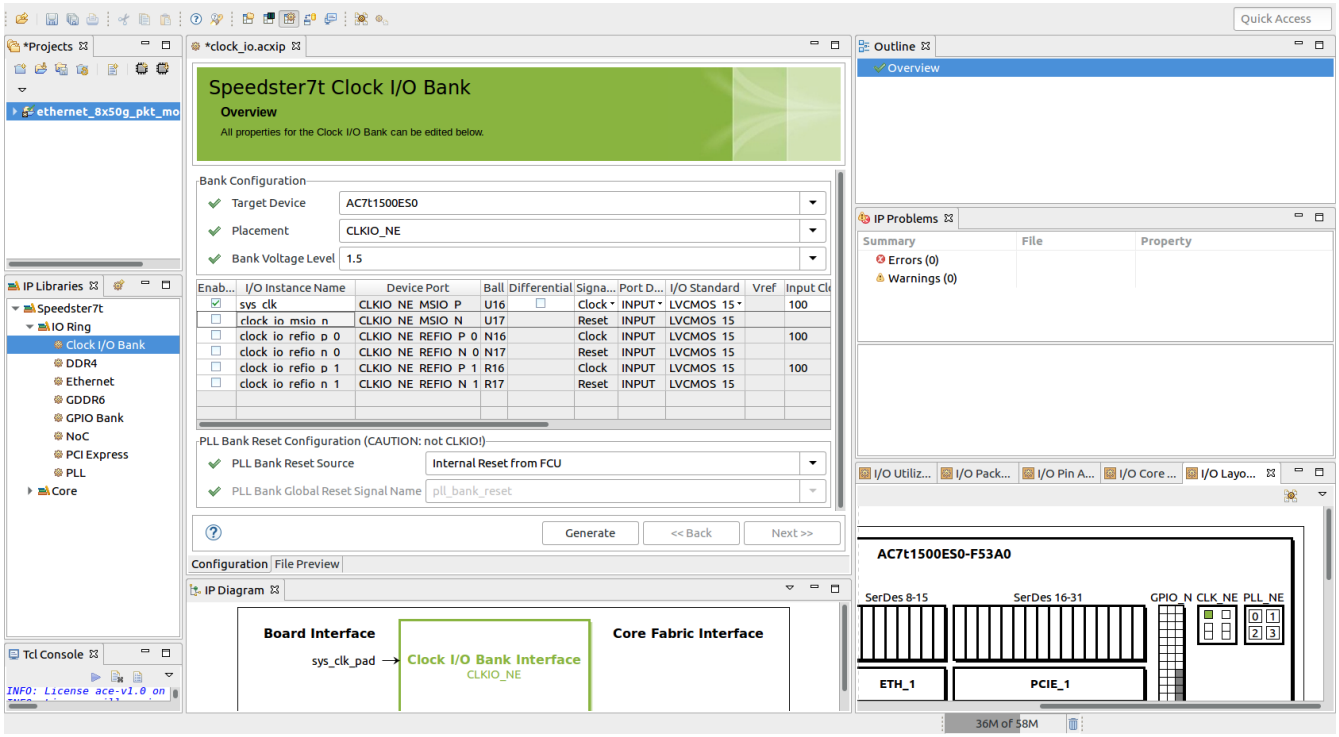


Figure 11: ***Clock I/O Configuration in ACE I/O Designer***

NoC

NoC PLL

Next, in the same IP Libraries, select **PLL**. Firstly the NOC PLL needs to be configured with the desired placement in the same corner as the input clock and the appropriate clock output frequencies (200 MHz). Name the file as `pll_noc.acxip`.

Table 18: *Settings for noc_pll*

Setting	Value
Placement	PLL_NE_3
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	On
clkout0 Desired Frequency	200
Clkout Output 0 Port Name	noc_clk
Expose Clock Output to Core Fabric	Off

Ensuring that there are no errors or warnings, the `pll_noc.acxip` can be saved, and further IP files generated.

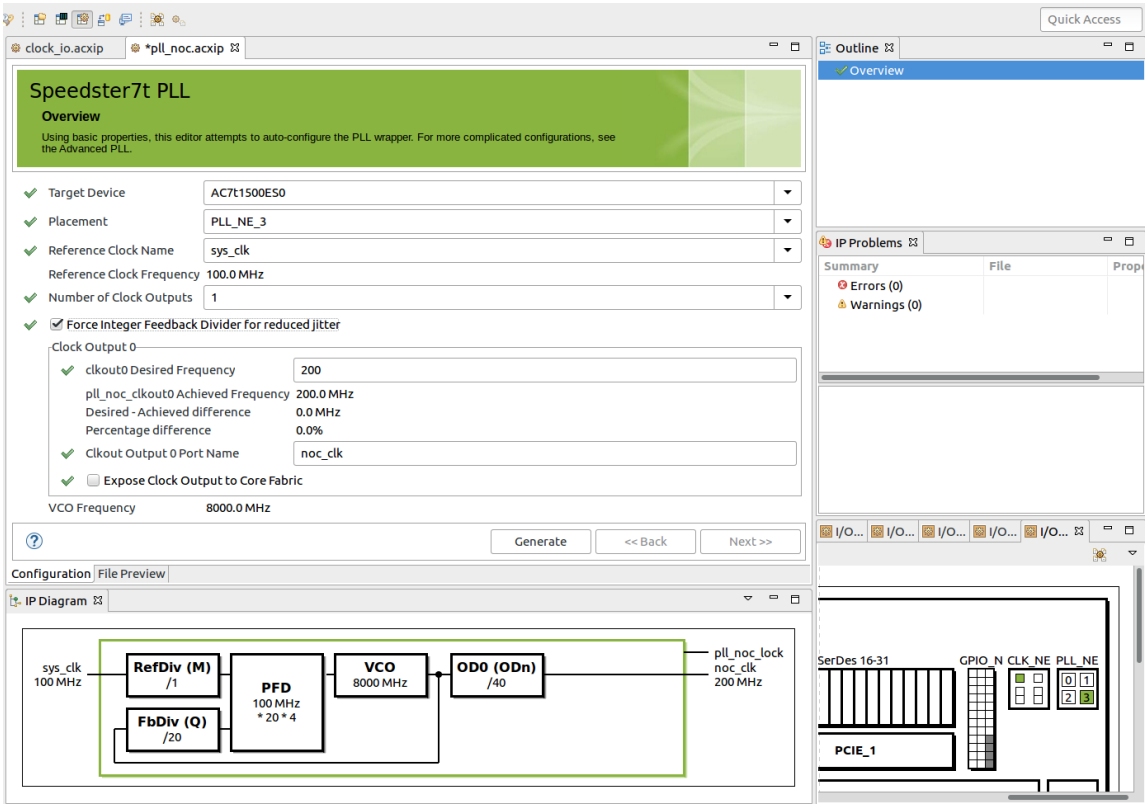


Figure 12: *Configuration of NoC PLL*

NoC Clock

Having configured the NoC PLL, it is then necessary to connect the clock to the NoC itself. From the **IP Libraries** tab select the NoC IP. and name this file `noc_1.acxip`.

There are only two settings required to configure the NoC:

- The Target Device, which should already be set to AC7t1500ES0.
- The NoC Reference Clock Name. For this field, select noc_clk from the available drop down list. Confirm that the NoC Reference Clock Frequency is indicated to be 200.0 MHz.

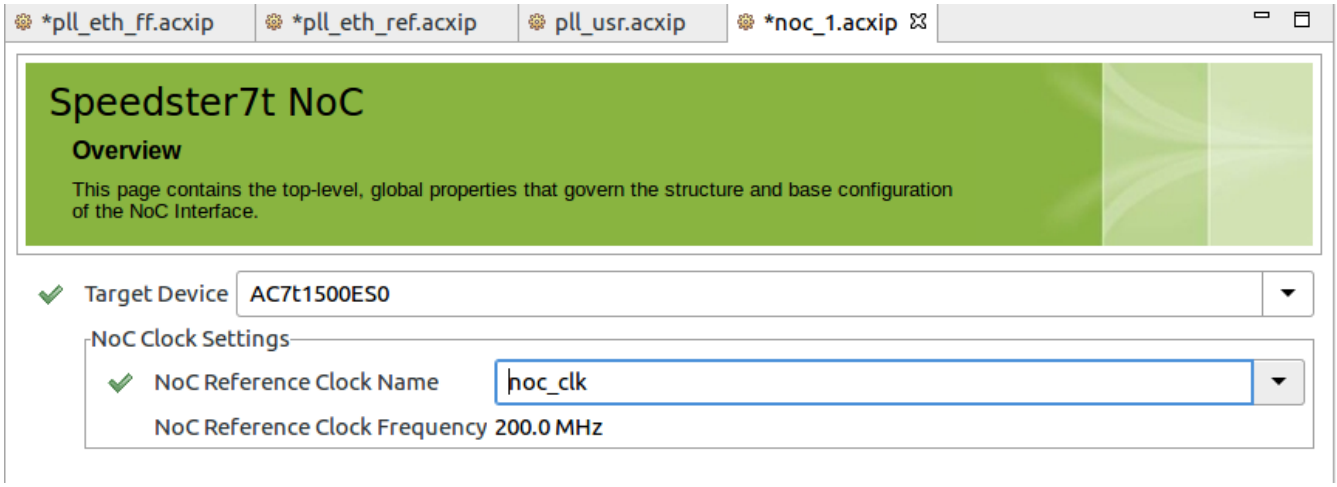


Figure 13: ***NoC IP Configuration in ACE I/O Designer***

Ethernet PLLs

For this example, configure the Ethernet Interface to operate at 400G. For this configuration, the following further clocks are required:

- 900 MHz as the Ethernet reference clock – name this signal `eth_ref_clk`. This signal is an internal clock not used by the user design.
- 800 MHz as the Ethernet `ff_tx_clk` and `ff_rx_clk`. – name this signal `eth_ff_clk`. This signal is an internal clock not used by the user design.
- 507 MHz as the user design clock – name this signal `i_eth_clk`.

In order to provide the above clock frequencies, three further PLLs must be instantiated. Their names and settings are detailed below.

Table 19: ***Settings for eth_ff_pll***

Setting	Value
Placement	PLL_NE_1
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	On
clkout0 Desired Frequency	800
Clkout Output 0 Port Name	eth_ff_clk

Setting	Value
Expose Clock Output to Core Fabric	Off

Table 20: ***Settings for eth_ref_pll***

Setting	Value
Placement	PLL_NE_0
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	On
clkout0 Desired Frequency	900
Clkout Output 0 Port Name	eth_ref_clk
Expose Clock Output to Core Fabric	Off

Table 21: ***Settings for usr_pll***

Setting	Value
Placement	PLL_NE_2
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	Off
clkout0 Desired Frequency	507
Clkout Output 0 Port Name	i_eth_clk
Expose Clock Output to Core Fabric	On

Note

When configured as above, the VCO frequency should equal 6083.99 MHz

Again once all these PLLs are correctly configured, the **IP Problems** tab should show no errors or warnings; the `acxip` files can be saved, and the output IP files generated.

Ethernet Interface

Next, the user must configure the Ethernet interface. From the **IP Libraries** select **Ethernet**. Select the desired **Ethernet MAC Placement** for the interface. In the **Lane Configurations** table, select the desired Ethernet channel operating modes. For this example, chose the following configuration:

Table 22: *Ethernet Interface IP Settings*

Setting	Value
Placement	ETH_1
Ethernet Lane [7:0]	400Gx8
Ethernet Reference Clock Name	eth_ref_clk
MAC FIFO Clock 0 Name	eth_ff_clk
MAC FIFO Clock 1 Name	eth_ff_clk
Ethernet Reset Source	Internal Reset from FCU

Table Note

The clocks selections should all appear as drop-down menu items if the preceding Ethernet PLLs have been correctly configured

Once the lanes are correctly specified, then the clocks from the PLL need to be connected to the **MAC Clock Settings**. The clock names use those specified in the previous PLL configuration, `eth_ref_clk, eth_ff1_clk, eth_ff2_clk.`

.

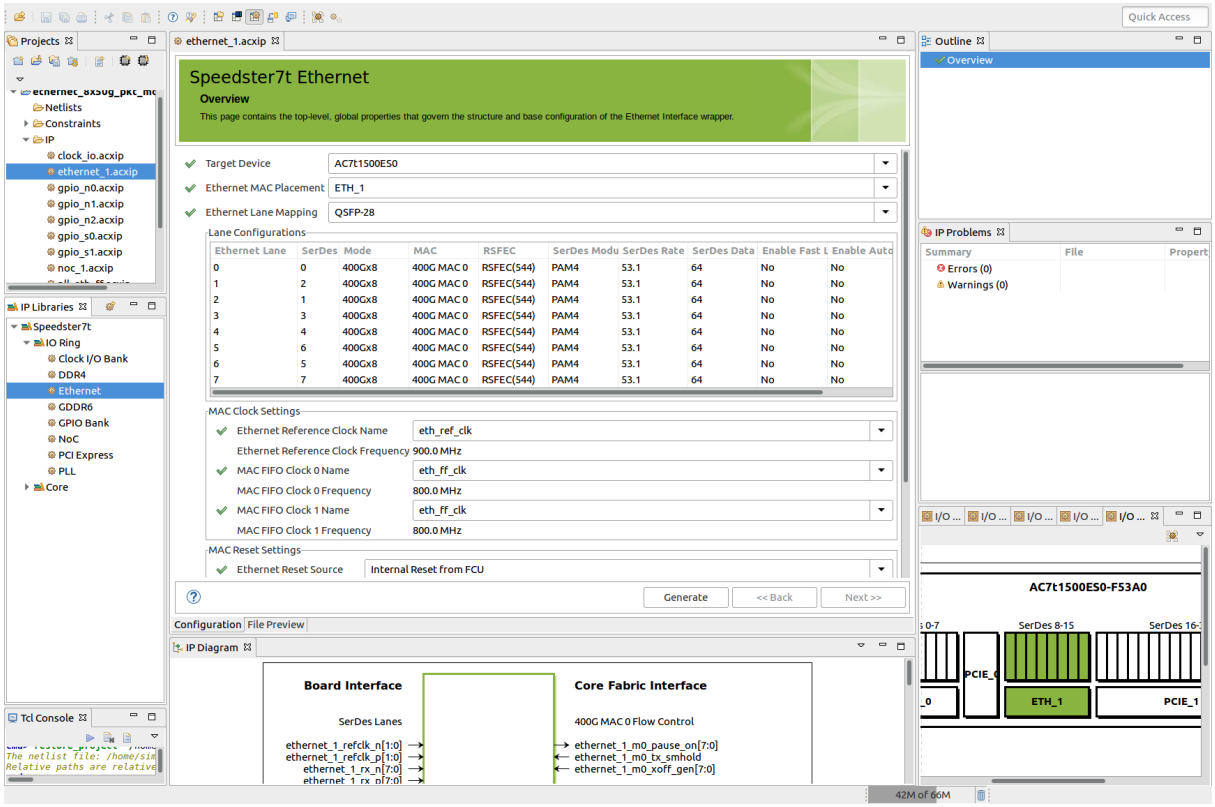


Figure 14: *Ethernet Interface Configuration*

If the user intends to build a design with multiple Ethernet interfaces, then the existing Ethernet interface can be cloned. In the Project pane, under IP, select the Ethernet IP `.acxip` file. Right-clicking this file provides a **Clone IP** option. The cloned IP instance(s) can subsequently be configured individually.

Check for Errors and Generate the Bitstream

After all the configuration options are selected, the **IP Problems** pane reports any errors or warnings that occurred with the configuration. If there are no errors or warnings reported, the user can be assured that the entire I/O interface with all the required IP are integrated properly. Once these checks are done, on any of the acxip IP file tabs, select **Generate.** This operation generates all the necessary files including constraint files, simulation configuration files, and the bitstream for the entire I/O ring.

Generated Constraint files

As part of the generate process, ACE creates an `.sdc` file with all the clock defined from any PLL. In addition a `.pdc` file is generated listing the I/O from each generated IP. In particular the Ethernet IP contains a number of flow control and status outputs which then specified by this `.pdc` file. The user is required to instantiate this I/O in their top-level design file (even if they are not using the signal). This instantiation ensures alignment during the place and route process between the I/O specified within the generated `.pdc` file, and the user design.

Once the above steps are completed, the user has a fully configured I/O ring and is able to connect to the Ethernet Interface in their design using NAPs and the direct-connect status and flow control signals.

Chapter - 7: Extended Features

The Ethernet Interface system supports additional extended features for particular usage scenarios.

Low Latency

Allowing lower latency with 10G and 25G modes when not using any FEC the Quad-PCS supports a fast one-lane mode. In this mode, the PCS provides the SerDes interface data on its internal data path and the PMA layer must bypass all gearboxes to allow minimized delay. The fast one-lane mode is only available for a SerDes width of 32 bit. For further details on low-latency mode and how to configure the SerDes for this mode, please contact Achronix support.

Loopback

For verification and validation purposes several, two loopback modes are implemented in the PMA:

- A direct (local) loopback returning all data from the PCS transmit back to the PCS receive. Transmit data is transmitted normally to the SerDes. Received data from the SerDes is ignored.
- A reverse (remote) loopback re-transmitting all data from SerDes receive back to SerDes transmit. Received data from the SerDes is passed through to the PCS. Transmitted data from the PCS is ignored.

The loopbacks operate all within the system reference clock domain, hence avoiding need for specific buffering.

Note

The loopback modes can be enabled for all lanes only — either all channels operate in a loopback mode or their normal data path.

Direct (Local) PMA Loopback Mode

The direct PMA loopback mode is enabled by setting the appropriate configuration register. Even if the direct loopback mode is enabled, the transmit SerDes clock must be run at a frequency corresponding the selected operational mode.

Reverse SerDes Loopback Mode

The reverse SerDes loopback mode is enabled by setting the appropriate configuration register. Even if the reverse loopback mode is enabled, the transmit/receive SerDes clocks must be run at a frequency corresponding the selected operational mode (and the transmit and receive SerDes bandwidths must the same).

Loopback Limitations

The direct (local) PMA loopback can only be used when RS-FEC mode is active (i.e., 10G/25G plain 64/66b modes cannot use loopback). The root cause for this limitation is that for non-RS FEC mode, the PCS output is 66-bit block based (0 to 65 bits). The final translation from the 66-bit internal bus to 40/80-bit SerDes interface is done by the PMA block in the SerDes clock domain. For the non-RS FEC. the receive PCS input mode always expects 40-bit data. In order to loopback the 66-bit output to 40-bit input, an extra gear box with memory is required.

Note

When using fast_1lane_mode=1, the loopback is functional also for 10G/25G modes, but operates slower than the line rate by a factor of 32/40 (80% of the original rate).

Revision History

Version	Date	Description
1.0	02 Sep 2020	Initial Achronix release.
1.1	16 Oct 2020	Added the chapter, Speedster7t Ethernet IEEE Timestamping (see page 36).