Speedster7t Ethernet User Guide (UG097)

Speedster FPGAs

Achronix
Data Acceleration

Copyrights, Trademarks and Disclaimers

Copyright © 2026 Achronix Semiconductor Corporation. All rights reserved. Achronix, Speedcore, Speedster, and ACE are trademarks of Achronix Semiconductor Corporation in the U.S. and/or other countries All other trademarks are the property of their respective owners. All specifications subject to change without notice.

Notice of Disclaimer

The information given in this document is believed to be accurate and reliable. However, Achronix Semiconductor Corporation does not give any representations or warranties as to the completeness or accuracy of such information and shall have no liability for the use of the information contained herein. Achronix Semiconductor Corporation reserves the right to make changes to this document and the information contained herein at any time and without notice. All Achronix trademarks, registered trademarks, disclaimers and patents are listed at http://www.achronix.com/legal.

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 1

Features 1

400G/200G PCS Layer 1

100G PCS Layer 1

10/25/40/50G PCS Layers 1

Reed-Solomon FEC (RS-FEC) 2

MAC 2

System Multirate and Multichannel 2

Architecture 2

PMA (SerDes) 3

PCS 4

MAC 4

Specifications 4

Channels 4

Bitmux 6

SerDes 6

SerDes Pad Numbering 6

SerDes Quad Assignment 6

SerDes Lane Mapping 7

System Latency 7

Chapter 2 : System Architecture 9

Ethernet Interface System Architecture 9
Data Flow 10
Data Connections 11
Status and Flow Control Signals 11
Control and Status Registers 11
IP Generation and Configuration 12
EIU 12
NAP Columns 12
Memory Buffering 13
Clocks 13
Quad Segmented Mode (QSI) 13
Configuration 14
NAP Data Connections 14
Timestamp 15
Transmit Flags 15
Transmit FCS Flags 16
Receive Flags 17
Receive Sequence ID 18
Receive FCS Control 19
Transmit Clock Frequencies 19
10G/25G/50G/100G 19
200G/400G 21
Receive Clock Frequencies 21
10G/25G/50G/100G 21
200G/400G 22
Combined Transmit and Receive Frequencies 22

Express and Preemptive MAC Channels 22
Enabling PMAC 23
Direct Connect Interface Signals 24
Prefix 24
MAC Identifier 24

Chapter 3 : 2D NoC Connectivity 28

Quad-Segmented Mode 28

Chapter 4 : Ethernet Clocks 31

Clock Domains 31
Clock Frequencies 32
Reference and FIFO Clocks 32
SerDes Clocks 33
Selecting Clock Frequencies 34
Flow Control 34
Transmit 100G and Below 34
Latency and Overrate 34
Multiple NAPs 34
Traffic Shaper 35
2D NoC Arbitration Parameters 35
Transmit 200G/400G Quad Segmented Mode 37
Receive Flow Control 38

Chapter 5 : Design Considerations 39

Setting Mode, MAC ID, and EIU Channels for the Ethernet Node 39

ACX_ETHERNET_NODE 39
Core ACX_ETHERNET_NODE Configuration Parameters: 40
EIU Channel Mapping 41
EIU Channels 41
Ethernet Configuration Overview 42
Example: Configuring the 100G (4 × 25G) Ethernet Rate 42
IP Configuration 42
Instantiating ACX_ETHERNET_NODE Configured for 100G (4 × 25G) using the Wrapper: 44
Example: Configuring Two Channels of 200G (4 × 50G) Ethernet Rate 44
IP Configuration 45
Instantiating ACX_ETHERNET_NODE Configured for Two Channels of 200G (4 × 50G) using the Wrapper: 46
Example: Configuring Four Channels of Mixed Ethernet Rates 47
IP Configuration 47
Instantiating ACX_ETHERNET_NODE Configured for the four channels using the Wrapper: 48

Chapter 6 : Register Maps 51

Global Address Space 51
CSR Addressing 51
Target ID Addressing 51
IP ID Addressing 52
Address Dictionary 52
System Control 53
Base Address 53
400G MAC Register Map 54
Base Address 54

Register Definitions 54
Frame Truncation *58*
400G PCS Register Map 62
Base Address 62
Register Definitions 62
Quad MAC Register Map 68
Base Address 68
Register Names 68
Register Definitions 69
Frame Truncation *74*
Quad PCS Register Map 81
Base Address 81
Register Names 82
Register Definitions 82
SerDes Register Map 92
Base Address 92
Register Names 93
Chapter 7 : IEEE Timestamping 96
Overview 96
Enabling Timestamping within ACE 96
Designing With TSN Signals *97*
Frame Timestamping 99
Receive Timestamping *99*
Transmit Timestamping *99*
Transmit Identifier *99*

Capturing Transmit Timestamp *100*

IEEE 1588 TX One-Step Frame Field Update 100

Overview 100

One-step Control Vector *100*

IEEE 1588 Message Header Structure 101

Update Function Overview 103

Transparent Clock Concepts 103

Overview *103*

End-to-End Transparent Clock *104*

Peer-to-Peer Transparent Clock *105*

Locally Generated Event Messages 105

Transport Layer Identification 106

UDP Payload Checksum Correction Field Update 107

Generic Transmit One-Step 64-bit Frame Field Overwrite 108

Chapter 8 : Ethernet IP Support in ACE 109

Step 1 – Creating a Project 109

Step 2 – Configure and Generate ACE IP 109

System Clock Input 109

NoC 110

NoC PLL *110*

NoC Clock *112*

Ethernet PLLs 113

Ethernet Interface 115

Check for Errors and Generate the Bitstream 116

Generated Constraint files 116

Chapter 9 : Extended Features 118

Low Latency 118

Loopback 118

Direct (Local) PMA Loopback Mode 118

Reverse SerDes Loopback Mode 118

Loopback Limitations 118

Chapter 10 : Revision History 120

Chapter 1 : Introduction

Speedster®7t FPGAs 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 the latest generation SerDes which can individually support 100 Gbps data rates. With eight of these SerDes per Ethernet interface, on AC7t1500/AC7t1400/AC7t700 devices, each interface can support 1 × 400 Gbps Ethernet IP channel. On AC7t800 devices, each interface can support up to 2 × 400 Gbps Ethernet IP channels.

The number of Ethernet interfaces varies according to the device. AC7t1500/AC7t1400 devices have two Ethernet interfaces. AC7t800/AC7t700 devices only have one Ethernet interface each.

> **Note**
>
> Throughout this document, references to the AC7t1500 also apply to the AC7t1400, and references to the AC7t800 also apply to the AC7t700. The only exception is that the AC7t1400 and AC7t700 are each limited to an aggregate SerDes bandwidth of 500 Gbps across all active lanes.

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. 16× 25G or 8× 25G configurations are not available.

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 the 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

- 10G–400G operation: Dedicated MAC for 200G/400G, plus two Quad-MAC blocks supporting multiple lower-rate channels
- IEEE 1588 receive and transmit timestamping
  - One-step timestamp correction for 10G up to 100G, 200G, and 400G. Accuracy of 2 ns
  - Event frames support
- 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, 25G, 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 four 50 Gbps lanes each
- Up to four (AC7t800) or two (AC7t1500) 200G Ethernet channels using two 100 Gbps lanes each
- Up to two (AC7t800) or one (AC7t1500) 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.

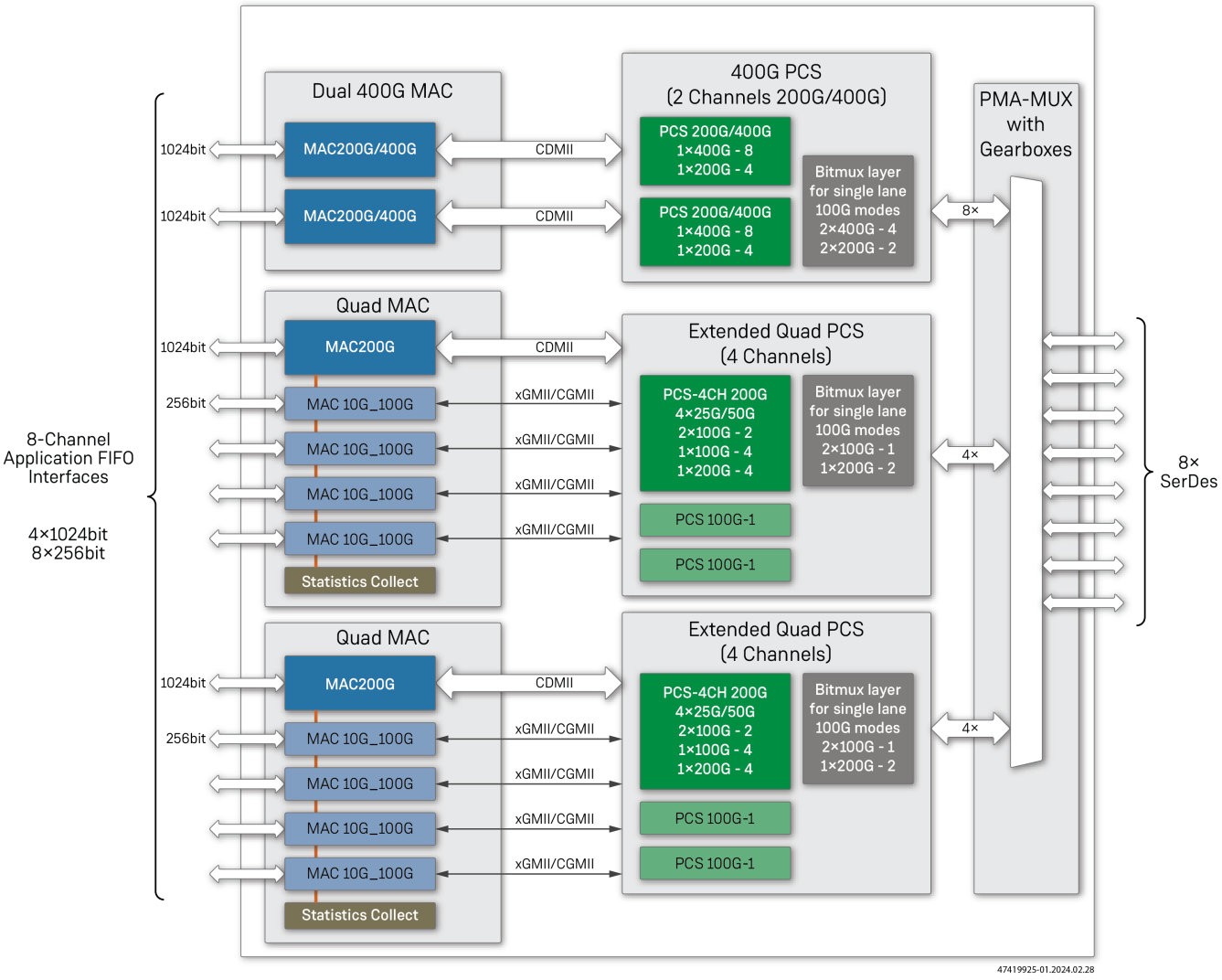


Figure 1 • Ethernet Interface Block Diagram

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 SerDes is required for applications other than Ethernet (or for user-provided 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 any SerDes from an Ethernet interface is used directly, then all SerDes from that group are switched to the fabric and the related Ethernet MAC and PCS are no longer available for use.

PCS

The physical coding sublayer (PCS), connects between the SerDes and the MAC. Consisting of a dual-channel 400G PCS configurable to support either 2× 200G or 2× 400G operation, and a twin Quad-PCS, the components can be configured as 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 dedicated dual channel 400G MAC for 400G/200G operation which connects to the dual channel 400G PCS, and two instances of a Quad-MAC, each connecting to a Quad-PCS. Each 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.

> **Warning!**
>
> On AC7t1400/AC7t1500 devices, the 400G MAC 1 (lanes 4 to 7) does not support 200G × 2 or 400G × 4. For those configurations, please use MAC 0 (lanes 0 to 3). This restriction limits the AC7t1400/AC7t1500 to only one 400G channel and two 200G channels per Ethernet interface.

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)	PMA Width	PMA Interface Frequency	PMA Electrical
400G × 8	8	53.125G	1	400G over 8 lanes (2:1 bitmux).	64	830.078125 MHz	PAM4
400G × 4	4	106.25G	1 (AC7t1500) 2 (AC7t800)	400G over 4 lanes (4:1 bitmux).	128	830.078125 MHz	PAM4

Mode × Lanes	SerDes Lanes (per Channel)	SerDes Rate per Lane	Possible No. of Channels	Description (with Coding Options)	PMA Width	PMA Interface Frequency	PMA Electrical
200G × 4	4	53.125G	2	200G over 4 lanes (2:1 bitmux).	64	830.078125 MHz	PAM4
200G × 2	2	106.25G	2 (AC7t1500) 4 (AC7t800)	200G over 2 lanes (4:1 bitmux).	128	830.078125 MHz	PAM4
100G × 4	4	25.78125G or 26.5625G	2	100G over 4 lanes (no FEC, RSFEC-KR4 or RSFEC-KP4).	32	805.6640625 MHz or 830.078125 MHz	NRZ or PAM4
100G × 2	2	53.125G	4	100G over 2 lanes (RSFEC-KP 2:1 bitmux).	64	830.078125 MHz	PAM4
100G × 1	1	106.25G	8	100G over 1 lane (RSFEC-KP 4:1 bitmux).	128	830.078125 MHz	PAM4
50G × 2	2	25.78125G or 26.5625G	4	50G over 2 lanes (RSFEC-KR or RSFEC-KP).	64	402.83203125 MHz or 415.0390625 MHz	NRZ or PAM4
50G × 1	1	53.125G	8	50G single lane (RSFEC-KP 4:1 bitmux).	64	830.078125 MHz	PAM4
40G × 4	4	10.3125G	2	40G over 4 lanes (optional Base-R (Firecode) FEC according to Clause 74 of IEEE 802.3).	32	322.265635 MHz	NRZ
25G × 1	1	25.78125G	8	25G single lane (66b or RSFEC-KR).	32	805.6640625 MHz	NRZ
10G × 1	1	10.3125G	8	10G single lane (66b).	32	322.265635 MHz	NRZ
200G × 2 + 100G × 1	2 + (1×1)	106.25G	1 set (AC7t1500) 2 sets (AC7t800)	One channel of 200G over 2 lanes, (4:1 bitmux) plus one channel of 100G over 1 lane (RSFEC-KP 4:1 bitmux).	128	830.078125 MHz	PAM4
200G × 2 + 100G × 2	2 + (2×1)	106.25G and 53.125G	1 set (AC7t1500) 2 sets (AC7t800)	One channel of 200G over 2 lanes, (4:1 bitmux) plus one channel of 100G, over 2 lanes (RSFEC-KP 2:1 bitmux).	128 and 64	830.078125 MHz	PAM4

Bitmux

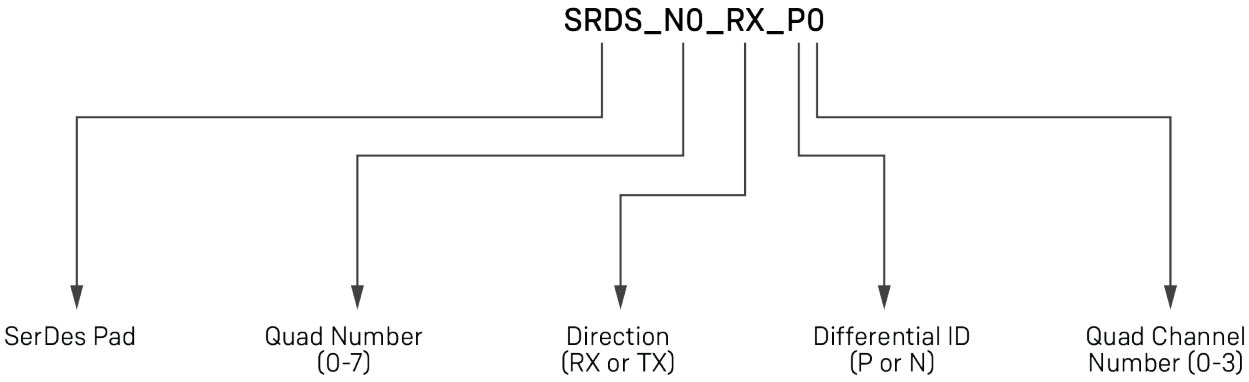
Bitmux refers to a layer of multiplexing that occurs between the PCS and the SerDes. This multiplexing provides the ability to widen the data interface to the PCS (and MAC) while reducing the overall system frequency.

For example, 400G-8 requires 8 50G SerDes lanes. From the **Channel Configurations** (page 4) table above, it can be seen that when configured as 50G, the SerDes is set to a 64-bit interface, operating at 830MHz. By using a 2:1 bitmux, the data bus is widened to 16 lanes of PCS, with each PCS operating at 415 MHz.

SerDes

SerDes Pad Numbering

The SerDes pads have a consistent numbering scheme to assist with board layout and simulation.



47419925-02.2022.25.11

Figure 2 • SerDes Pad Numbering

In total, for the Speedster AC7t1500, there are 32 SerDes lanes, configured as eight quads, each of four lanes. Only a subset of these SerDes are connected to the Ethernet subsystems on a device. These assignments are detailed in **SerDes Quad Assignment** (page 6) below.

SerDes Quad Assignment

Each Ethernet subsystem uses two SerDes Quads, giving eight SerDes lanes in total. The SerDes Quads associated with each Ethernet subsystem are detailed in the following table.

Table 2 • SerDes Quads per Ethernet Subsystem

Device	Register Name			
	Ethernet Subsystem 0		Ethernet Subsystem 1	
	Lower Quad (Lanes 0-3)	Upper Quad (Lanes 4-7)	Lower Quad (Lanes 0-3)	Upper Quad >(Lanes 4-7)
	SERDES_0	SERDES_1	SERDES_0	SERDES_1
AC7t800	–	–	SRDS_N4	SRDS_N5
AC7t1500	SRDS_N6	SRDS_N7	SRDS_N4	SRDS_N5

SerDes Lane Mapping

In order to provide additional flexibility for board layouts, the internal SerDes lanes can be remapped to the SerDes pads. This mapping flexibility allows a single board design to support a number of common connector standards. The eight internal SerDes channels are mapped into a different sequence of SerDes pads, numbered as SerDes Lane ID 0-7, as shown in the SerDes Lane Mapping (page 7) table below. This mapping is independent of the SerDes speed. For any single design, the mapping is static. Mapping is selected during the Ethernet subsystem configuration and is set by the generated bitstream. See the Ethernet Interface section in Speedster7t Ethernet IP Software Support in ACE (page 115).

Table 3 • SerDes Lane Mapping

SerDes Lane ID	0	1	2	3	4	5	6	7
Linear	0	1	2	3	4	5	6	7
QSFP	0	2	1	3	4	6	5	7
QSFP-DD	0	6	1	7	2	4	3	5

In the SerDes Lane Mapping (page 7) table, above, the top row represents the device pads. Thus, SerDes Lane ID 0 would equate to pins SerDes Quad 0, Lane 0 (SRDS_N0_TX_P0 etc.). SerDes Lane ID 1 would be pins SerDes Quad 0, Lane 1 (SRDS_N0_TX_P1 etc.). The lower rows represent which internal SerDes lane is mapped to the corresponding SerDes Lane ID. For example, if QSFP is selected, then SerDes Lane ID 1 is driven by the internal SerDes channel 2. SerDes Lane ID 1 pads equate to pad pairs SRDS_N0_RX_N1/P1 and SRDS_N0_TX_N1/P1.

System Latency

When considering latency, consider the whole system end-to-end latency. Although SerDes (PMA) latency values are often published, these neglect the significant latency introduced by the MAC and PCS layers.

For the Speedster7t FPGA, the following transmit and receive latencies have been measured between the SerDes pad and the NAP ingress and egress interface.

Table 4 • Speedster7t SerDes Transmit and Receive Latencies

Rate	Configuration	Transmit latency	Receive latency
10G	1 × 10G Single NAP	204 ns	204 ns
25G	1 × 25G Single NAP	360 ns	360 ns
50G	1 × 50G Single NAP	215 ns	215 ns
50G	2 × 25G Single NAP	235 ns	235 ns
100G	2 × 50G Single NAP	177 ns	177 ns
100G	4 × 25G Single NAP	150 ns	150 ns
400G	8 × 50G QSI mode	165–187 ns	167–189 ns

Chapter 2 : System Architecture

Ethernet Interface System Architecture

Due to the fully integrated architecture of the Speedster7t FPGA, the connections to and use of the Ethernet interface subsystem are different from what might be expected. In previous FPGA architectures, an RTL wrapper which contained the Ethernet interface (perhaps including models of the SerDes) would be generated. This wrapper would then be instantiated within a design and directly connected to the clock, reset, status and data ports. It is expected that the correct clock and reset strategies are followed, and the integrity of the data connections are confirmed.

With a Speedster7t FPGA, the usage methods and connections are very different. The full Ethernet interface, containing MAC, PCS and PMA (SerDes) is fully integrated within the Speedster7t FPGA system components. The MAC connects via an Ethernet interface unit (EIU (page 12)) directly to the 2D network-on-chip (NoC). These components and the connections between them are not user accessible and the user design cannot connect directly to the Ethernet interface components. This architecture is detailed **below** (see figure 3).

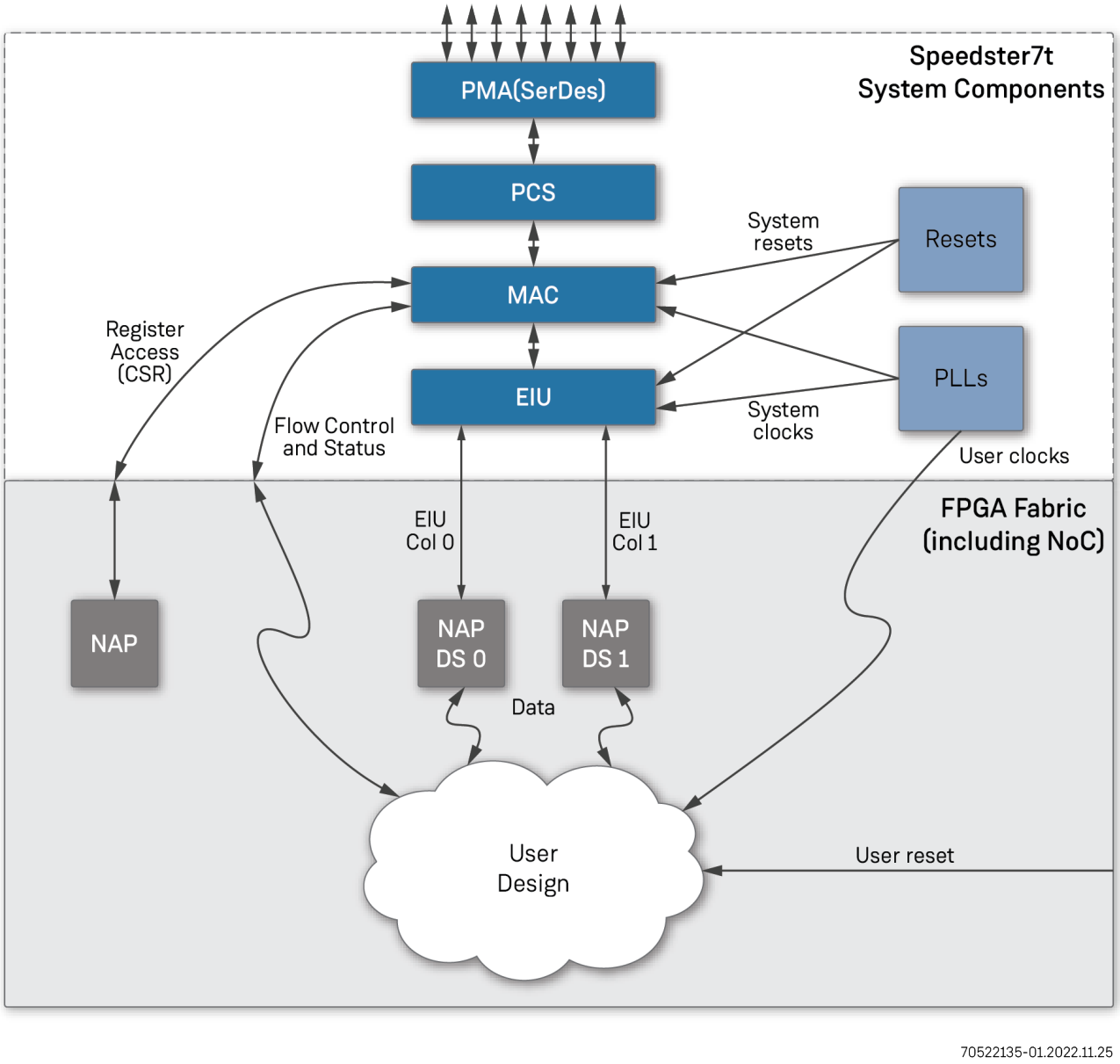


Figure 3 • Ethernet Interface System Architecture

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 2D NoC column to the EIU (page 12)
  - Within the EIU, the packet can be passed through or stored and forwarded depending on the required mode

    - For the advanced modes **Quad Segmented Mode** (page 0) the EIU assembles and formats the packets accordingly
- The whole packet, from MAC header to the end of the payload, is transmitted in parallel form to the MAC. The MAC processes the packet, adding preamble, SFD and FCS. The MAC also applies any interpacket gap, (IPG).
- The packet is passed through the PCS to the SerDes, and on to the serial pins of the device

Packet reception is a reverse of the above sequence.

With reference to the **above architecture** (see figure 3), connections must be made to the Ethernet interface using the following methods.

Data Connections

Connections to the Ethernet data streams are made using network access points (NAPs) placed within the 2D NoC. The 2D NoC greatly simplifies access to the Ethernet interface with simplified data streams presented to the user design.

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

For full details of the NAP data connections, refer to **NAP Data Connections** (page 14).

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. Direct connections can then be made to these signals within the user design. These signals are detailed in the **Direct-Connect Interface table** (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 2D 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 FPGA. 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 2D 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 Speedster7t Ethernet IP Software Support in ACE (page 109). 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:

- **Ethernet Interface Bitstream File** – This file is merged with other bitstream files during final bitstream assembly and programs the Ethernet interface upon FPGA power-up, ensuring the Ethernet interface is ready for operation when the device enters user mode.
- **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 FPGA, ensuring that the correct clocking and reset strategies are followed is no longer the responsibility of the user. The Speedster7t FPGA, combined with ACE, manages these low-level tasks freeing users to concentrate on the core parts of their design.

EIU

The Ethernet interface unit (EIU) is unique to the Speedster7t family of FPGAs. There is an EIU for each Ethernet interface and each instance manages the connection of the MAC data interfaces to the 2D NoC. The EIU adapts the traffic flow and performs clock domain crossings between the 2D NoC and the MAC. It is further responsible for dividing the traffic when 400G or 200G quad segmented mode is used. For all modes, the EIU is responsible for packetizing the traffic so it can be sent down the appropriate 2D NoC column to the correct NAP endpoint.

NAP Columns

Each EIU supports two 2D NoC columns on which the Ethernet NAP endpoints can be located. The EIU only sends and receives packets to NAPs placed within those two columns. The particular columns on which an EIU is connected for each Ethernet interface is detailed in the table below (page 12). Columns are numbered from 1, starting on the West side.

Table 5 • Ethernet 2D NoC columns

Device	Ethernet Subsystem		Total 2D NoC Columns
	0	1	
AC7t700/800		Columns 1 and 2	2
AC7t1400/1500	Columns 1 and 2	Columns 4 and 5	4

Memory Buffering

The EIU has memory buffering primarily to handle 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 200G and 400G **Quad Segmented Mode** (page 13) (see below), the EIU operates in a store-and-forward mode. Each whole packet is buffered until it is fully received from the 2D 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 2D NoC core operating frequency up to 1.7 GHz and the MAC `ff_clk` frequency domains. Each of these clock domains are user configurable. Details on configuring the 2D NoC clock can be found in **Ethernet IP Software Support in ACE** (page 109). The minimum MAC `ff_clk` frequencies are defined in the **Reference and FIFO Clock frequencies** (page 32).

Quad Segmented Mode (QSI)

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 2D NoC support spreading the bandwidth across four NAPs.

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 starts on the lane after the previous packet ended. For example, with the NAPs identified as NAP-1 to NAP-4, then if the end-of-packet (EoP) word is output on NAP-2, the next packet asserts its start-of-packet (SoP) word on NAP-3.

For additional details, including diagrammatic representations of the packet structure, refer to **Speedster7t Ethernet 2D NoC Connectivity.** (page 28) 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 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.

> **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.

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_ETHERNET` primitive, full details of which can be found in the "Speedster7t Network on Chip Primitives" chapter of the ***Speedster7t Component Library User Guide*** (UG086)[1]. When connected to the EIU (page 12), the NAP connections are as listed below.

Table 6 • Ethernet NAP Data Ports

Name	Direction	Description
`rstn`	Input	Asynchronous reset input. This signal resets the NAP interface. This signal does not affect the 2D NoC.
`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 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.
`tx_valid`	Input	Assert high to issue a word of data to the NAP.
`tx_dest[3:0]`	Input	The 4-bit destination ID for the Ethernet packets. This value must be fixed to `4'hf` to indicate that packets must be sent to the EIU reserved address.
`tx_sop`	Input	Start of packet and end of packet indicators. These signals are required to be set for each packet transmitted to the Ethernet subsystem.
`tx_eop`		
`tx_data[292:0]`	Input	Ethernet packet to be transmitted to the EIU. • `tx_data[255:0]` : Packet data • `tx_data[260:256]`: Mod data. Valid when `tx_eop` is set • `tx_data[290:261]`: Transmit flags or timestamp. Timestamp is valid when `tx_sop` is set. Transmit flags are valid on other packet cycles. • `tx_data[292:291]`: Unused
`rx_ready`	Input	Asserted high by user logic to indicate it is ready to receive data.

1 https://www.achronix.com/documentation/speedster7t-ip-component-library-user-guide-ug086

Name	Direction	Description
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. This value is always 4'hf to indicate that the packet was received from the EIU. This output is generally unused.
rx_sop rx_eop	Output	Start of packet and end of Ethernet packet indicators.
rx_data[292:0]	Output	Received Ethernet Packet. • rx_data[255:0] : Packet data • rx_data[260:256]: Mod data. Valid when rx_eop is asserted. • rx_data[290:261]: Receive flags or timestamp. Timestamp is valid when rx_sop is set. Receive flags are valid on rx_eop. • 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 insert a value into the flag field aligned with tx_sop. This value is then captured by the outgoing MAC. If IEEE-1588 one-step correction is enabled, this timestamp value is used to update the correction field.
- For reception, if the user design is providing a timestamp count to the MAC, then the packet reception timestamp is captured (at the time the packet arrives within the MAC) and this value is provided into the flag field coincident with rx_sop.

For full details of how the timestamp fields are generated and used, including IEEE-1588 1-Step updating, refer to IEEE Timestamping. (page 96)

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 7 • Ethernet NAP Transmit Flags

Slice (1)	Direction (relative to NAP)	Flag Name	Description
Flag[16:0]	Input	ID	One-step update control vector. See IEEE Timestamping (page 96)
Flag[17]	Input	Frame	Set to indicate an IEEE 1588 event frame. See IEEE Timestamping (page 96)

Slice [1]	Direction (relative to NAP)	Flag Name	Description
Flag[18]	Input	Tx Error	Force transmit frame error. • `1'b1` – Frame is transmitted with an incorrect CRC (no error control code is inserted). If CRC insert is set to `1'b0`, (CRC already in frame from application) then one bit within the last 4 bytes of the frame is changed to corrupt the CRC. • `1'b0` – No change to transmitted frame or CRC.
Flag[19]	Input	CRC insert	Add FCS field to transmitted frame. • `1'b1` – Transmit MAC appends a CRC (FCS) field of 4 bytes to the transmitted frame • `1'b0` – Transmit MAC does not append an FCS to the transmitted frame.
Flag[20]	Input	CRC invert	Invert any inserted FCS field • `1'b1` – When set to `1'b1`, the CRC of the frame is inverted whenever CRC (flag[19]) or CRC override (flag[21]) are set. • `1'b0` – CRC is not inverted.
Flag[21]	Input	CRC override	CRC override. Only effective when CRC (flag[19]) = `1'b0`. • `1'b1` – The CRC field of the frame passed to the MAC is overridden. • `1'b0` – The CRC field of the frame passed to the MAC is not modified. If CRC (flag[19]) = `1'b1`, then CRC override has no effect.
Flag[22]	Input	Class A	Class A packet. Used for AVB rate limiting.
Flag[23]	Input	Class B	Class B packet. Used for AVB rate limiting.
Flag[29:24]	Input	Unused	Must be set to `6'b0`.

Table Notes

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

Transmit FCS Flags

There are three flags that control CRC (FCS) insertion on transmitted frames. The insertion and modification is performed by the relevant transmit MAC. The CRC-32 polynomial is as specified in the 802.3 specification.

The following table lists the effects of these three flags in combination.

Table 8 • MAC FCS Transmit Flags

CRC Insert	CRC Override	CRC Invert	Description
0	0	x	Pass-thru. The frame is not modified.
1	x	0	Append a CRC-32 field to the frame.
1	x	1	Append an inverted CRC-32 field to the frame.
0	1	0	Overwrite the CRC-32 field in the frame (calculates a new CRC-32 based on the frame data).
0	1	1	Overwrite the CRC-32 field in the frame (calculates a new CRC-32 based on the frame data and inverts it).

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. These flags are valid when `eop`is received as part of a valid transfer, (`valid` and `ready` both asserted)

Table 9 • Ethernet NAP Received Flags

Slice	Direction (Relative to NAP)	Flag Name	Description
Flag[0]	Output	Error	Frame error indicator.
Flag[1] (1)	Output	Length Error	Set to `1'b1` if the frame has an invalid length. This can be either a too short frame (less than 64 bytes), or a too long frame (length greater than the value programmed in register `FRM_LENGTH`), or a frame which has a different amount of payload than specified in the frame payload length field. Frames below 64 bytes in length might not be delivered to the NAP.
Flag[2]	Output	CRC Error	Set to `1'b1` if the frame was received with a CRC field that did not match the CRC calculated by the receive MAC. The frame is invalid.
Flag[3]	Output	Decode Error	Set to `1'b1` if the frame was received with an unknown control character. The frame is invalid.

Slice	Direction (Relative to NAP)	Flag Name	Description
Flag[4] (1)	Output	FIFO Overflow	Set to `1'b1` if a FIFO overflow was detected during frame reception. The received frame is truncated by the receive MAC and is invalid.
Flag[5] (1)	Output	Short Frame	Set to `1'b1` if a fault sequence (`CDMII 0x9c`) was detected while receiving the frame, or if the frame is a short frame (had less than 64 bytes on the line). If set due to a short frame the `Length Error` field is also set.
Flag[6] (2)	Output	Inverted CRC	Set to `1'b1` if the frame was received with a CRC that matches the inverted CRC (inverted via XOR'ing the CRC with the value in the register `CRC_INV_MASK`) calculated by the receive MAC. When such frame is received by the MAC, the frame is still considered erroneous (`CRC Error` is also set).
Flag[7]	Output	Transmit Error	Set to `1'b1` if a transmit error control character (`0xFE`) has been received from the line. The frame is invalid.
Flag[8]	Output	VLAN	Set to `1'b1` to indicate that the frame is a VLAN frame (containing at least one VLAN tag).
Flag[13:9]	Output	Sequence ID	Frame sequence indicator. Details the order that the packet was output from the MAC. Used particularly in packet mode to reassemble the original packet sequence.
Flag[29:14]	Output	Unused	Set to `16'b0`.

Table Notes

1. A frame with length error due to `FRM_LENGTH` violation or `FIFO Overflow` is truncated and the last bytes of the frame might contain arbitrary data.
2. The user application must monitor `Inverted CRC` and ignore the error indication in order to accept these frames if so required.

Receive Sequence ID

The receive sequence ID is a 5-bit count which indicates the order in which packets were transmitted from the EIU (page 12):

- For a single stream (all data rates up to and including 100G), a continuous count of receive sequence IDs is received at the single NAP.
- For Quad Segmented Mode (page 0) (200 and 400G), the same `sequence ID` is present on all segments of the same packet.

Receive FCS Control

In the same way that the transmit MAC can insert CRC (FCS) fields into the outgoing frame, the receive MAC can optionally forward the received FCS field to the application, or it can remove it. Error checking of the frame against the FCS is still performed regardless of whether the FCS itself is forwarded or not.

Receive forwarding of the FCS is controlled by the `COMMAND_CONFIG` register, bit[6], in each MAC. For full details of the programming interface refer to the Register Map (page 51)

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 so that when the longest potential frame is transmitted, the whole frame can be sent within the required time.

Minimum Ideal Packet Size Frequencies

Assuming a system needs to support jumbo frames (9000 bytes with a NAP data width of 256 bits), the following minimum transmit frequencies are required to not starve the MAC during packet transmission.

Table 10 • NAP Ideal Packet Sizes Minimum Transmit Frequencies

Data Rate (Gbps)	Frequency (MHz)
10	40
25	98
50	195
100	390

Note

The transmit frequencies above only achieve the full stated line rate if all packets are of a length that fits the NAP data width exactly. Thus, all packet lengths are a multiple of 32 bytes.

The table above lists the minimum frequencies required for packets that are exact multiples of the NAP data width (32-bytes). For general operation over any packet size, it is necessary to increase the transmit frequency in order allow for loss of throughput caused when the packet width is not a multiple of 32-bytes. In order to calculate the worst-case frequency requirements, it is also necessary to consider any extra bytes, or byte intervals that are added to a packet before transmission.

Packet Overhead

The following bytes (or byte intervals) are added to an Ethernet packet on transmission:

- Preamble = 8 bytes
- Interpacket Gap (IPG) = Originally 12 bytes for lower Ethernet rates. For 10G and 25G the IPG is reduced to 5 bytes. For 40G, 50G, 100G, 200G and 400G it is reduced to a minimum of 1 byte.
- Frame CheckSum (FCS) = 4 bytes. This can be generated as part of the packet to the NAP or added by the MAC, (as in the Achronix supplied designs). Similarly, it can be removed on reception by the MAC or passed through to the NAP. Any FCS added by the MAC is expressed as `MAC_FCS` in the calculations below.

Minimum Non-Ideal Packet Size Frequencies

The general equation to calculate the required rate is:

Freq = Number of cycles to write packet to NAP × Data Rate / ((Packet Length + Preamble + IPG + MAC_FCS) × 8)

The worst case length packet to transmit from the NAP is 65 bytes. This requires 3 cycles of the NAP `tx_clk`. The equation for a 65-byte packet, at a transmit rate of 100Gbps, with an IPG of 1 and the FCS created by the user design is then:

Freq = 3 × 100G / ((65 + 8 + 1 + 0) × 8) = 506.7MHz.

The table below details the minimum non-ideal packet size transmit frequencies for both FCS from the MAC and FCS from the NAP.

Table 11 • NAP Non-Ideal Packet Sizes Minimum Transmit Frequencies

Data Rate (Gbps)	IPG (bytes)	Frequency (MHz)	
		FCS from NAP	FCS from MAC
10	5	48	46
25	5	121	115
50	1	254	241
100 (1)	1	507	481

Data Rate (Gbps)	IPG (bytes)	Frequency (MHz)	
		FCS from NAP	FCS from MAC

Table Notes

1. The maximum rate at which a single NAP operates is 100G. For 200G and 400G line rates, the throughput is distributed across multiple NAPs.

If the user application determines to use larger IPG values, then the required frequency can be calculated using the formula above.

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 mode. Therefore, there are no requirements on the minimum transmit frequency. The NAP can transmit at any frequency (up to a maximum of 507MHz) as the EIU waits until it has received a whole packet (including jumbo frames) before transferring that packet to the MAC. To achieve maximum throughput, the NAPs should operate at 507MHz as detailed in the table above (page 20)

Receive Clock Frequencies

For received packets, the the same requirements exist 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 12 • NAP Lower Data Rate Receive Minimum Frequencies

Data Rate (Gbps)	FCS to NAP	FCS removed by MAC
	Frequency (MHz)	Frequency (MHz)
10	48	46
25	121	115
50	254	241

Data Rate (Gbps)	FCS to NAP	FCS removed by MAC
	Frequency (MHz)	Frequency (MHz)
100	507	481

200G/400G

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

Table 13 • NAP Higher Data Rate Receive Minimum Frequencies

Data Rate (Gbps)	FCS to NAP	FCS removed by MAC
	Frequency (MHz)	Frequency (MHz)
200	507	481
400	507	481

> **Note**
>
> For 200G quad mode (using four NAPs) the receive frequency requirement of 507 MHz gives a total potential bandwidth of greater than 400G. However, this receive frequency is required so that all NAPs in an EIU column can operate at the full bandwidth of up to 507MHz.

Combined Transmit and Receive 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 is used for both transmit and receive. In these configurations, the higher of the transmit or receive clock frequencies must be met for the NAP single clock.

Express and Preemptive MAC Channels

To support the IEEE 802.3br standard, the Quad-MACs support both express and preemptive MAC channels for all Ethernet channel rates of up to 100G. For each of the four Quad-MAC channels, there are two interfaces, named `pmac` and `emac`. These names apply to both the direct connect signals and the register names. The preemptable

MAC (pmac) interface is the low priority interface where frames can be preempted. The express MAC (emac) interface is the high priority interface where frames are sent ahead of preemptable frames.

The selection of the appropriate MAC channel is made in the instantiation of the ACX_NAP_ETHERNET with the tx_eiu_channel and rx_eiu_channel parameters (see the ***Speedster7t Component Library User Guide (UG086)***[2]). Referring to the ACX_NAP_ETHERNET description table, **EIU Channels**, observe that EIU lanes 16-19, and 24-27 are used for preemptive MAC channels, and lanes 20-23 and 28-31 are used for express MAC channels.

> **Note**
>
> If the preemptive EIU channels are selected for a NAP, ensure that the pmac has been enabled as detailed below. If the channel is not enabled, no Ethernet traffic is sent or received via the NAP.

If only one interface on a MAC channel is required, the express MAC (emac) should be used. If both interfaces are used, express frames are prioritized and preempts normal frames using a MAC merge sublayer.

Enabling PMAC

To enable the pre-emptive MAC (pmac), in the ACE Ethernet IP configuration wizard, set the **Enable PMAC** option to **Yes**. This is shown below for a 100G (4 × 25G) channel:

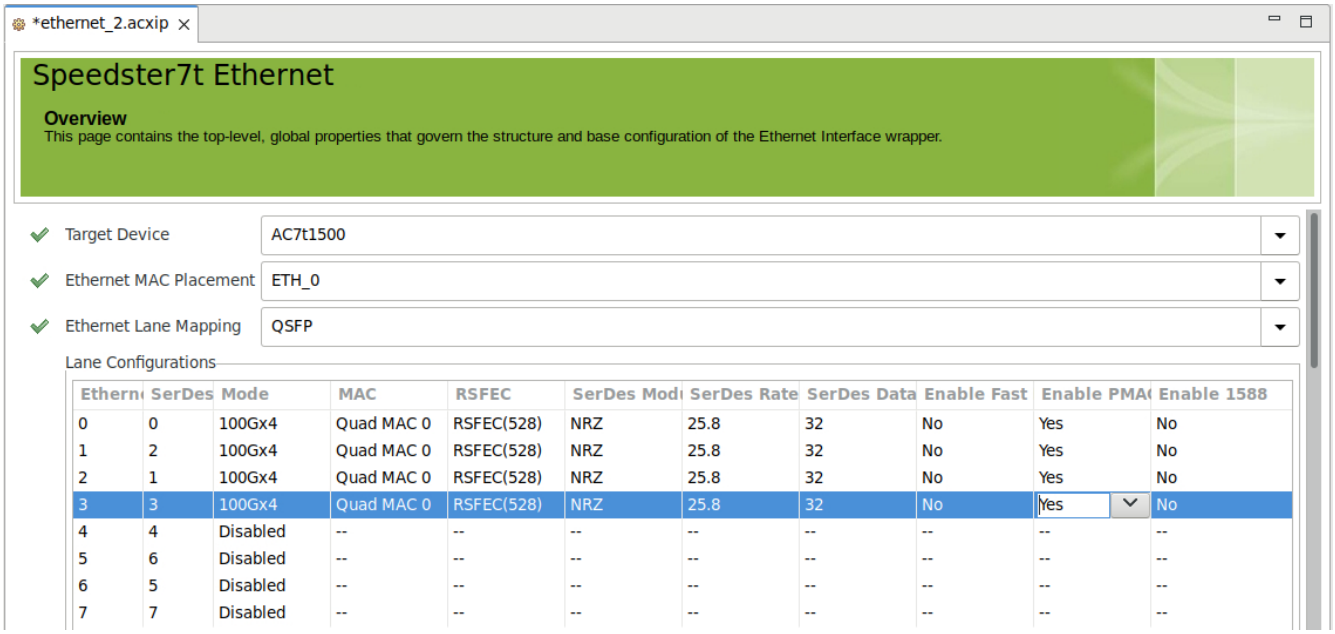


Figure 4 • Enabling PMAC Within ACE

> **Note**
>
> Preemptive channels are only available for Ethernet channel rates of 100G or less. Enable PMAC cannot be set for channel rates of 200 or 400G.

2 https://www.achronix.com/sites/default/files/docs/Speedster7t_Component_Library_User_Guide_UG086.pdf

Direct Connect Interface Signals

The DCI 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 (page 109)**. 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 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
- `quad0` is for QUAD0 MAC
- `quad1` is for QUAD1 MAC

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

Table 14 • Ethernet Controller Direct Connect Interface

Pin Name (1)	Direction (4)	Width	Comments (6)	Clock
Quad MAC (2)				
`<prefix>_quad<n>_tx_hold_req`	Input	4	Per channel 100G MACs. Holds and preempts the `pmac` transmission, if needed, allowing `emac` traffic to be transmitted. Can be used either for test and debug or when a higher layer function anticipates an `emac` frame being written and must prepare the MAC instead of requiring the MAC to prepare automatically based on the non-empty status of the eMAC FIFO.	Synchronous to `ff_clk_divby2`.
`<prefix>_quad<n>_lpi_txhold`	Input	4	Per channel 100G MACs. Prevents MAC transmission of a frame even if data is stored in the FIFO.	Asynchronous Input. Synchronized internally to `ref_clock`.
`<prefix>_quad<n>_mac_stop_tx`	Input	4	Per channel control of MAC transmit. When the respective input for each lane is asserted (1'b1), the MAC transmit state machine stops after any outgoing frame has been sent completely (does not corrupt outgoing frames). If further frames are available in the transmit FIFO, the MAC does not begin transmitting them until the respective `mac_stop_tx` is de-asserted (1'b0).	Synchronous to `ref_clock_divby2`.
`<prefix>_quad<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 vector is used to signal the creation of PFC control frames. When link pause mode is enabled, only the 0 bit per channel (bits 0,8,16,24) is used.	Asynchronous input synchronized to `ref_clock`.
`<prefix>_quad<n>_pmac_xoff_gen`	Input	32		

Pin Name [1]	Direction [4]	Width	Comments [6]	Clock
<prefix>_quad<n>_pmac_pause_on	Output	32	Transmit paused/class congestion indication (8-bit value per channel) from emac/pmac. Bit 0 per channel (bits 0,8,16,24) is also used to indicate link pause. When asserted to 1'b1, indicates a running pause counter has been started because an Xoff frame (pause/PFC) was received. In link pause mode, the transmitter is also stopped when the command_config bit, PAUSE_PFC_COMP, is not set. When the PAUSE_PFC_COMP bit is set, the transmitter is not stopped and it is the responsibility of the application to assert mac_stop_tx to implement proper flow control.	Synchronous to ref_clock_divby2.
<prefix>_quad<n>_emac_pause_on	Output	32		
<prefix>_quad<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_clock.
<prefix>_quad<n>_emac_pause_en	Output	4		
<prefix>_quad<n>_pmac_enable [5]	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 and RX_EN) control bits. Per channel from 100G MACs. Can be left unconnected if not used.	Synchronous to reg_clock
<prefix>_quad<n>_emac_enable	Output	4		
<prefix>_m<n>_ffp_tx_ovr	Output	4	FIFO overflow truncation error indication (1 bit per channel). Asserted when the FIFO write control logic truncates a frame as either the FIFO had overflowed, or the frame transferred is larger than the FIFO when operating in store-and-forward mode.	Synchronous to ff_clock_divby2.
<prefix>_m<n>_ffe_tx_ovr	Output	4		
<prefix>_m<n>_mac_tx_underflow	Output	4	Indicates transmit FIFO became empty during transmission. A frame has been corrupted. 1 bit per channel.	Synchronous to ref_clock_divby2.
400G/200G MAC [3]				
<prefix>_m<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, only bit 0 is used.	Asynchronous input synchronized to ref_clock.
<prefix>_m<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_clock_divby2.
<prefix>_m<m>_pause_on	Output	8	Transmit paused/class congestion indication, one bit per priority class. When asserted to 1'b1, indicates a running 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_clock_divby2.
<prefix>_m<m>_tx_ovr_err	Output	1	Indicates a FIFO overflow truncation error. Asserted when the FIFO write control logic truncates a frame because either the ff_tx_rdy de-assertion was not respected by the application, or the frame transferred is larger than the FIFO when operating in store-and-forward mode.	Synchronous to ref_clock_divby2.

Pin Name [1]	Direction [4]	Width	Comments [6]	Clock
<prefix>_m<m>_tx_underflow	Output	1	The transmit FIFO became empty during transmission. A frame has been corrupted.	Synchronous to ref_clock_divby2.
<prefix>_m<m>_rx_buffer0_at_threshold	Output	4	EIU RX buffer channel 0 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit RX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, RX_THRESHOLD[7:0] sets the threshold value for rx_buffer0_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_rx_buffer1_at_threshold	Output	4	EIU RX buffer channel 1 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit RX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, RX_THRESHOLD[7:0] sets the threshold value for rx_buffer1_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_rx_buffer2_at_threshold	Output	4	EIU RX buffer channel 2 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit RX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, RX_THRESHOLD[7:0] sets the threshold value for rx_buffer2_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_rx_buffer3_at_threshold	Output	4	EIU RX buffer channel 3 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit RX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, RX_THRESHOLD[7:0] sets the threshold value for rx_buffer3_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_tx_buffer0_at_threshold	Output	4	EIU TX buffer channel 0 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit TX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, TX_THRESHOLD[7:0] sets the threshold value for tx_buffer0_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_tx_buffer1_at_threshold	Output	4	EIU TX buffer channel 1 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit TX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, TX_THRESHOLD[7:0] sets the threshold value for tx_buffer1_at_threshold[0].	Synchronous to ff_clk_divby2.
<prefix>_m<m>_tx_buffer2_at_threshold	Output	4	EIU TX buffer channel 2 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit TX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, TX_THRESHOLD[7:0] sets the threshold value for tx_buffer2_at_threshold[0].	Synchronous to ff_clk_divby2.

Pin Name [1]	Direction [4]	Width	Comments [6]	Clock
<prefix>_m<m>_tx_buffer3_at_threshold	Output	4	EIU TX buffer channel 3 threshold indication. The four threshold indicators are each set individually by their respective byte in the 32-bit TX_THRESHOLD parameter applied to the relevant ACX_NAP_ETHERNET. Thus, TX_THRESHOLD[7:0] sets the threshold value for tx_buffer3_at_threshold[0].	Synchronous to ff_clk_divby2.
Common				
<prefix>_ref_clock_divby2	Output	1	Reference clock divided by 2.	
<prefix>_m0_ff_clk_divby2	Output	1	MAC FIFO clock 0 divided by 2.	
<prefix>_m1_ff_clk_divby2	Output	1	MAC FIFO clock 1 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 in having logical names for each interface.
2. The variable <n> indicates the quad MAC number: 0 for QUAD0 MAC, and 1 for QUAD1 MAC.
3. The variable <m> indicates the 400G/200G MAC channel number: 0 or 1.
4. Direction is with respect to the interface subsystem. The direction is reversed for the signals from the perspective of the programmable fabric design.
5. A corresponding signal from the 200G quad MAC does not exist.
6. For quad MAC signals that are 1 bit per channel, all four MACs are not enabled, only the bits corresponding to any enabled MAC are active. For example, if 100G x 4 is selected, then only MAC 0 is active, and thus only bits[0] of any of the appropriate control signals are active.

Chapter 3 : 2D NoC Connectivity

As previously detailed, the data from the Ethernet IP is delivered via the Ethernet Interface Unit (EIU) and the 2D NoC to the NAPs in data streaming mode.

The Ethernet subsystem connects directly to specific columns on the 2D 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 following table lists the specific columns connected to the Ethernet subsystems.

Table 15 • 2D NoC Columns for Ethernet Subsystems

Device	AC7t800	AC7t1500	
Ethernet Subsystem Location	**Ethernet Subsystem 1**	**Ethernet Subsystem 0 (West)**	**Ethernet Subsystem 1 (East)**
2D NoC Column 1	1	1	4
2D NoC Column 2	2	2	5

Table Notes

- 2D NoC Columns are numbered 1 at the westernmost column and increment going east.

For interfaces using 100G or slower, the Ethernet sends 256-bit packets down the columns directly to NAPs. For interfaces running 200G or 400G, quad-segmented mode is used.

Quad-Segmented Mode

In quad-segmented mode, the 2D NoC sends a 1024-bit bus that is segmented across four NAPs. This applies to both 200G and 400G modes, quad mode always uses 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 as previously described. 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.

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. For example, a single processing engine for a 400GbE solution would require a 1024-bit bus running at approximately 728 MHz, whereas the quad-segmented

mode for either 400G or 200G uses four NAPs and requires four 256-bit buses running at 507 MHz. The following figure shows how the packets are arranged and segmented for the quad-segmented mode.

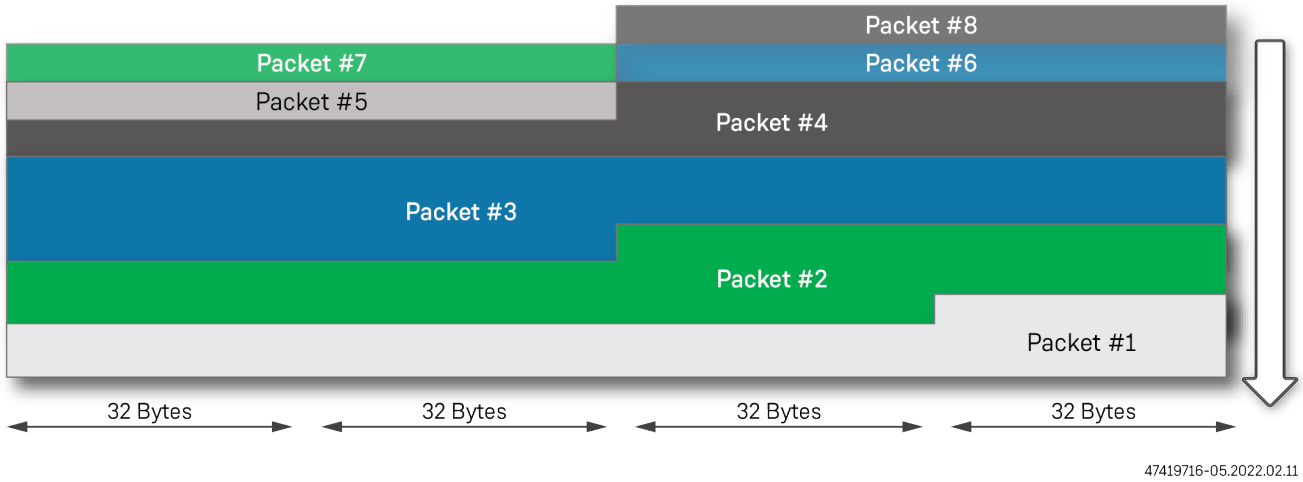


Figure 5 • Packet Segmentation for Quad-Segmented Mode

Each packet is distributed across four NAPs located on the designated columns of the 2D 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 following figure 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 located on a single column, or grouped farther apart.

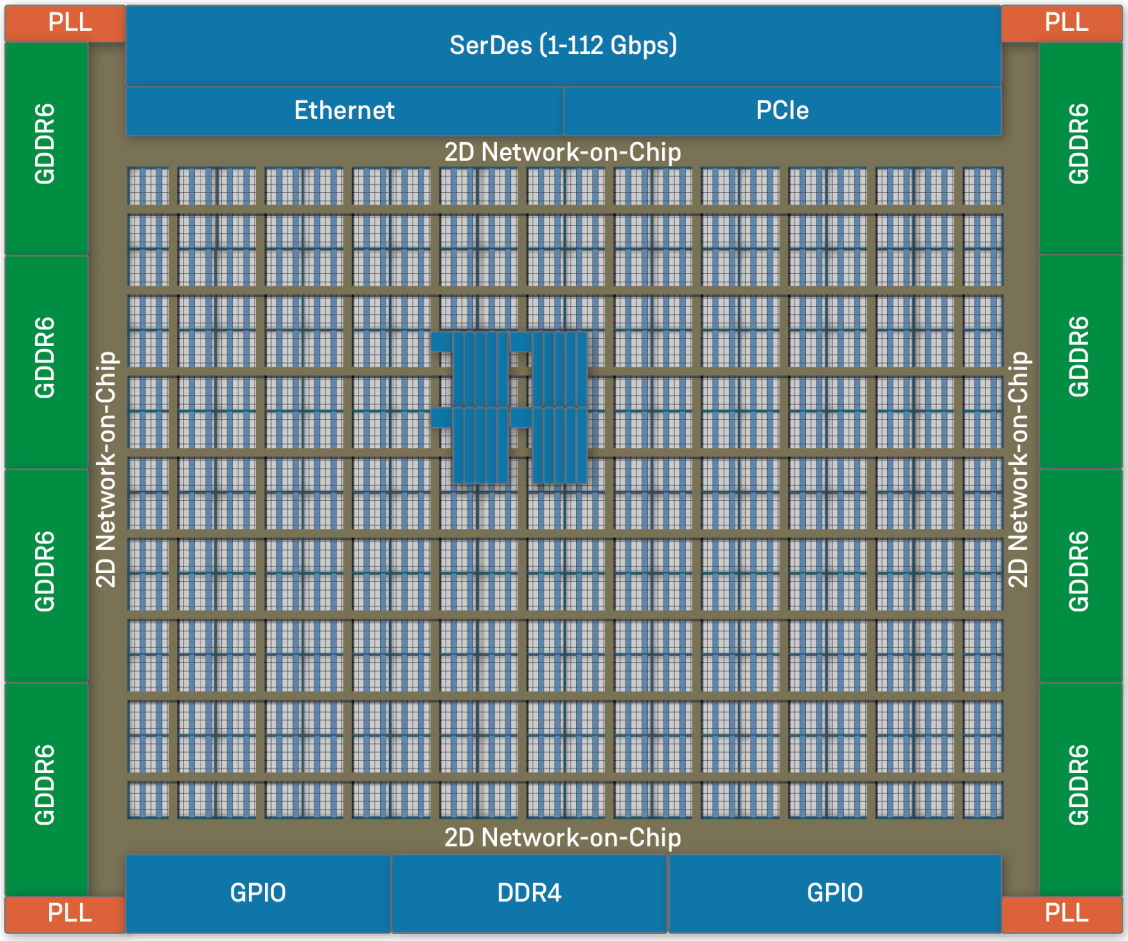


Figure 6 • Quad-Segmented Mode on the 2D NoC

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.

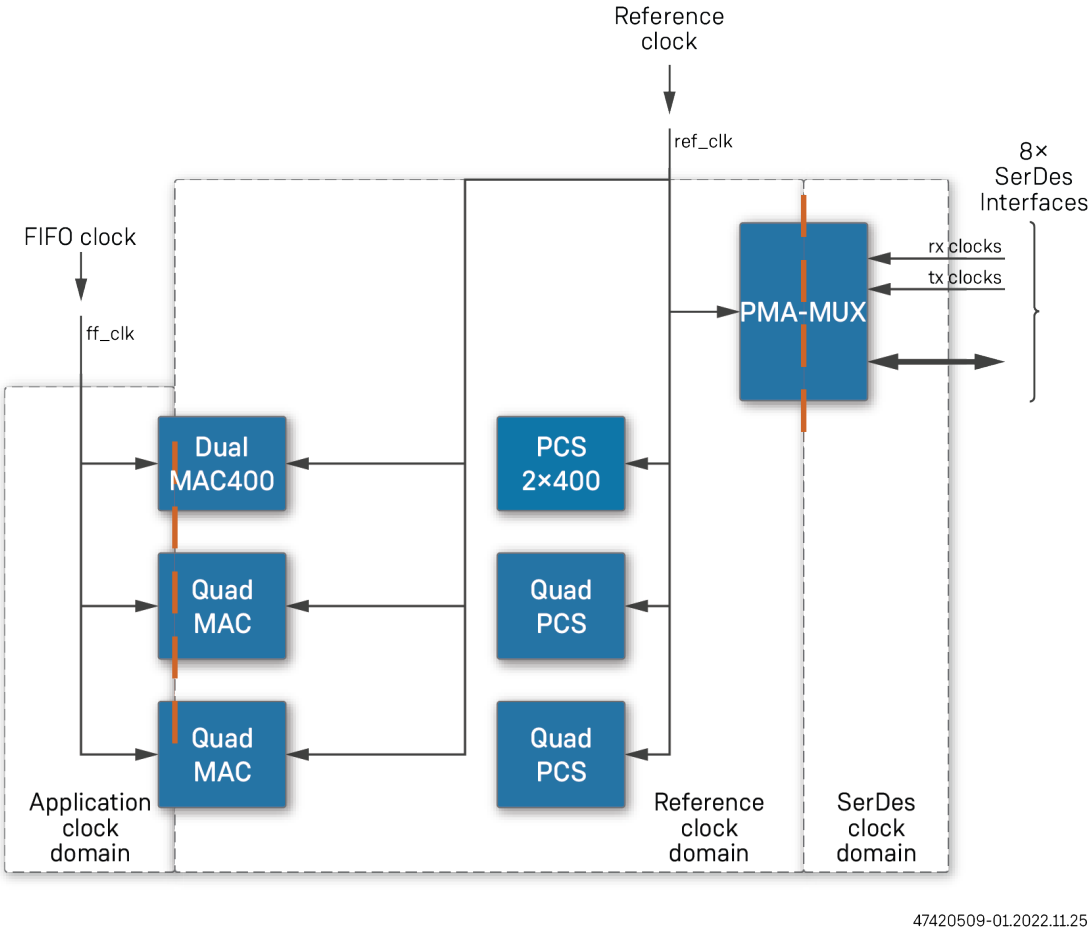


Figure 7 • Ethernet Interface Clock Domains

The clocks driving these clock domains are as follows:

- `ff_tx_clk` – FIFO transmit clock. This clock is driven from the EIU and 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 the Reference and FIFO Clock Frequencies (page 32) table.

- ff_rx_clk – FIFO receive clock. This clock is driven from the EIU and is used to read 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 the Reference and FIFO Clock Frequencies (page 32) table.
- 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, 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 the Reference and FIFO Clock Frequencies (page 32) table.
- serdes_clk – This clock drives the SerDes and is related to the SerDes line rate. The SerDes clock requirements are detailed in the SerDes Clock Frequencies (page 33) table.

The four clock domains operate independently, using asynchronous FIFOs to decouple the data that 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 the ACE I/O Designer tool as detailed in Ethernet IP Software Support in ACE. (page 109)

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 frequency ranges are detailed in the table below.

Table 16 • Reference and FIFO Clock Frequencies

Mode	MAC Reference Clock (ref_clk)		FIFO Clocks (ff_clk)		
	Min	Max	Min Theoretical	Min Recommended	Max
10G	323 MHz	900 MHz	59 MHz	160 MHz	782 MHz
25G	782 MHz (25G no RSFEC) 831 MHz (25G with RSFEC)	900 MHz	147 MHz	210 MHz	. 782 MHz.
50G	831 MHz	900 MHz	293 MHz		. 782 MHz.
100G	831 MHz	900 MHz	586 MHz		. 782 MHz.
200G(1) (2)	840 MHz	840 MHz	840 MHz		. 840 MHz.
400G(1) (2)	840 MHz	840 MHz	840 MHz		. 840 MHz.

Mode	MAC Reference Clock (ref_clk)		FIFO Clocks (ff_clk)		
	Min	Max	Min Theoretical	Min Recommended	Max

Table Notes

1. Clock frequencies of 850 MHz or above result in PCS errors. `ref_clk` MUST be set below 850 MHz.
2. . When in 200G/400G mode, the reference clock and the FIFO clock must have the same frequency..

Warning!

ACE 10.5.4 and earlier versions permit setting 200G/400G `ref_clk` values up to 900 MHz. `ref_clk` for these modes must be set below 850 MHz. This issue is to be resolved in future ACE releases.

The given minimum frequencies must also be respected. For example, for 25G (no RSFEC), a `ref_clk` of 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 17 • 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** (page 109). Upon selecting the desired Ethernet data rates, I/O Designer specifies the required minimum clock frequencies. 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.

Flow Control

Flow control is a key factor in any Ethernet system design to ensure that at no stage in the packet data path are any packets dropped or lost. The usual cause of these issues are buffers over or under-flowing. Due to the integrated nature of the Ethernet subsystem, with data delivered by the **EIU** (page 0) and 2D NoC, there are several buffers in the path from NAP to SerDes. The following describes what is required from a user design at a system level to ensure there are no packet losses.

Transmit 100G and Below

For all Ethernet channel speeds of 100G and below, a single NAP is used. This NAP can be operated at any frequency necessary in order to meet the Ethernet channel bandwidth. Clock frequency calculations for different data rates and configurations are detailed in **System Architecture** (page 9).

Latency and Overrate

For rates of 100G and below, the NAP is transmitting directly through the 2D NoC and EIU to the MAC. There is inherent latency in this data path meaning that it is possible for a NAP to overflow the MAC before the MAC is able to indicate that it is full. In addition, if the NAP is configured for worst-case length packets, it is also possible that an overrate condition can occur where the NAP can exceed the transmit bandwidth of the MAC, leading to overflows and data loss. Thus, 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 can support from a single NAP.

Multiple NAPs

When multiple NAPs share a column, each NAP retains the same latency characteristics as a single NAP operating in that column. However, if any NAP overrates, its associated MAC channel asserts backpressure to the EIU on a per-channel basis. The EIU then applies backpressure to the column, halting transmission from all NAPs in that column. As a result, NAPs that have not overrated also stop transmitting, causing their associated MAC channels to underflow. Therefore, when multiple NAPs share a column and operate at 100G or below, each NAP's transmit rate must be properly regulated to prevent overrating.

> **Warning!**
>
> Based on the above, it is necessary to implement transmission rate limiting for the Ethernet NAPs to ensure that the individual NAP overall data rate is not exceeded when using 100G or below Ethernet NAPs.

Traffic Shaper

To regulate the NAP transmission for NAPs supporting up to 100G, Achronix supplies a traffic shaper block built into the ACX_ETHERNET_NODE macro. For this macro, the clock frequency at which the NAP is operating is specified along with the desired channel rate. The traffic shaper then ensures that the NAP does not overrate. The traffic shaper block is freely available in the Speedster_Ethernet_ref_design_RD19 available via the KBA, **How do I Download Demonstration and Reference Designs?**[3] (login required).

2D NoC Arbitration Parameters

When there are multiple Ethernet NAPs on a column, it is necessary to ensure that the 2D NoC balances the throughput from each NAP so that all NAPs obtain equal bandwidth on the column. If not balanced, the NAPs at the top of a column, nearer the EIU, would get priority with the potential of causing NAPs lower down the column to have their packets stalled, leading to packet starvation at the MAC.

To control the NAP vertical arbitration, referring to the ACX_NAP_ETHERNET macro in the ***Speedster7t Component Library User Guide*** **(UG086)**[4], the s2n_arbitration_schedule parameter is used. The value of the parameter is set according to how many NAPs are in the column, and the position of the NAP relative to the other NAPs in the column. The values are shown in the following table.

Table 18 • s2n_arbitration_schedule Parameter Values

NAP Position in Column	s2n Value
8th from bottom	32'hc0808080
7th from bottom	32'h48102040
6th from bottom	32'h20820820
5th from bottom	32'h21084210
4th from bottom	32'h48888888
3rd from bottom	32'h24924924
2nd from bottom	32'h2aaaaaaa
Lowest	32'h2aaaaaaa

3 https://support.achronix.com/hc/en-us/articles/38065028969236-How-do-I-Download-Demonstration-and-Reference-Designs

4 https://www.achronix.com/documentation/speedster7t-component-library-user-guide-ug086

Note

1. The `s2n_arbitration_schedule` value is based on the relative position of the NAP compared to other NAPs on the column. The value is not based on the absolute row position. Thus, a NAP on row 7 may have the lowest setting of 32'h2aaaaaaa if there are only two NAPs on the column in rows 7 and 8. Equally, a NAP on row 7 could have a value of 32'h48102040 if there are 6 other NAPs in rows 1–6 below it.
2. The lowest NAP should always have the value of 32'h2aaaaaaa assigned to it regardless of the row on which it is placed. The NAPs above should have their values assigned working up the table above.
3. As the `s2n_arbitration_schedule` parameter is assigned based on position, it is strongly recommended that this parameter is assigned in the placements constraints file (.pdc) rather than the RTL. This placement file assigns the placement of all of the NAPs in a column, and can simultaneously assign the `s2n_arbitration_schedule` parameter.
4. It is not necessary to assign the `s2n_arbitration_schedule` parameter if there is only a single NAP in the column.
5. It is not necessary to assign the `n2s_arbitration_schedule` parameter. This parameter may be left at the default value of 32'hxxxxxxxx. All received traffic that is sent North to South (n2s) is from the EIU which is the single source of the column traffic.

The following code example demonstrates how to assign the `s2n_arbitration_schedule` in the placements constraints file (`.pdc`).

```
# ----------------------------------------------------------------------------
# How to place 8 ACX_ETHERNET_NAPs in two columns
# ----------------------------------------------------------------------------
# Automatically check if instance is present; if it is, then place
# Support Ethernet subsystem 1, hence columns 4 & 5
for {set ii 0} {$ii < 8} {incr ii} {
    # Row and Column calculations match testbench
    if { $ii > 3 } {set col 5} else {set col 4}
    switch [expr {$ii % 4}] {
        0       { set row 1; set arb_param 32'h2aaaaaaa }
        1       { set row 4; set arb_param 32'h2aaaaaaa }
        2       { set row 5; set arb_param 32'h24924924 }
        default { set row 8; set arb_param 32'h48888888 }
    }
    # Create a string for the NAP instance name
    set inst_str [subst
gb_all_ch_$ii\__gb_eth_ch_i_eth_ch.i_nap_eth.i_ethernet_node.u_nap_ethernet]
    # Create a string for the proposed site
    set site_str [subst s:x_core.NOC[$col][$row].logic.noc.nap_m]
    # If the instance exists, then place on the proposed site
    if { [find $inst_str -insts] != "" } {
        set_placement -fixed i:$inst_str $site_str
        # Set the s2n_arbitration_schedule on the NAP
        set_property s2n_arbitration_schedule $arb_param $inst_str
```

```
        # Display message in ACE console and log confirming placement and parameter
setting
        message "placed $inst_str on $site_str with s2n_arbitration_schedule $arb_param"
    }
}
```

The following code demonstrates how the same placement, and arbitration parameter would be set in the corresponding testbench.

```
// Bind the Ethernet naps and assign the arbitration parameters
generate for (genvar ii=0; ii<MAX_ETH_CHANNELS; ii++) begin : gb_all_ch
    if (ETH_CHANNEL_EN[ii]) begin : gb_eth_ch
        localparam ROW = ((ii % 4) == 0) ? 1 :
                         ((ii % 4) == 1) ? 4 :
                         ((ii % 4) == 2) ? 5 :
                                           8 ;

        localparam COL = (ii > 3) ? 5 : 4;

        // Ethernet 1 channels are in column 4 and 5
        // NAPs are numbered from 1,1

`ACX_BIND_NAP_ETHERNET(DUT.gb_all_ch[ii].gb_eth_ch.i_eth_ch.i_nap_eth.i_ethernet_node.u_
nap_ethernet,COL,ROW);

        // Set arbitration schedule is a function.  Delay to allow initial setup in
AC7t1500
        initial
            #10 ac7t1500.set_s2n_arbitration_schedule( COL, ROW,
                                                       ((ROW == 1) ? 32'h2aaaaaaa :
                                                        (ROW == 4) ? 32'h2aaaaaaa :
                                                        (ROW == 5) ? 32'h24924924 :
                                                                     32'h48888888)
                                                     );

    end
end
endgenerate
```

Transmit 200G/400G Quad Segmented Mode

When operating at 400G and 200G rates, the EIU is in Quad Segmented Mode (QSI) to support the bandwidth. In QSI, the four NAPs transmit, (and receive), in unison, with different NAPs containing the start and end of packet. The EIU operates in cut-through mode, combing the packet from the four NAPs into a single stream that is passed to the MAC. There is no EIU buffer storage in this mode, the only buffering being a small reassembly buffer in the EIU and the shallow FIFO at the front of the MAC, (32 entries deep).

The four NAPs should operate largely in lock-step, transmitting the same words of the same Ethernet frame on the same cycle. Some variance is allowed in the EIU reassembly buffer, however this must be limited to a few cycles of deviation. Therefore it is recommended to combine the four NAPs `tx_ready` signals and to use these to control transmission to all 4 NAPs.

As four NAPs are required in QSI mode, then there will be more than one NAP in any one column, with the recommended NAP layout is a 2x2 square for optimum timing closure. As there is more than one NAP per column and all NAPs should have equal access to transmit, the **2D NoC Arbitration Parameters** (page 0) should be applied to any QSI NAPs.

Receive Flow Control

For reception, the NAP must operate at a sufficient speed to receive the longest expected packet. If a receiving NAP provides back-pressure to the EIU (by de-asserting `rx_ready`), the packet is then stored within the MAC shallow buffers which are 32-words deep. If the NAP continues to issue back-pressure to the column, ultimately these MAC RX buffers overflow. This overflow is indicated to the user application via the **NAP Receive Flags** (see **System Architecture** (page 9)) asserting the `receive error` and `FIFO overflow` flags . For QSI mode, a receive FIFO is recommended for each NAP to allow for the small cycle to cycle variations in delivery of the four words that form a whole packet. The receive FIFO output can then be flow controlled to input the full packet into the user design.

In the *Speedster7t Ethernet Reference Designs Guide* (RD019) (available via **How do I Download Demonstration and Reference Designs?**[5]), where this receive flow control occurs, 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 ACX_LRAM2K_FIFO (each set to 72-bit width each) or two Speedcore7t BRAM72K_FIFO (set to 144-bit width each).

> **Note**
>
> It is strongly recommended, for all Ethernet data rates, that receive flow control is performed between packets, rather than during a single packet transmission.

5 https://support.achronix.com/hc/en-us/articles/38065028969236-How-do-I-Download-Demonstration-and-Reference-Designs

Chapter 5 : Design Considerations

Setting Mode, MAC ID, and EIU Channels for the Ethernet Node

When starting an Ethernet design using the ACX_ETHERNET_NODE, the initial step involves determining the Ethernet modes and the number of SerDes lanes the user requires for their design. This requirement influences the configuration of the ACX_ETHERNET_NODE's parameters, which is important for the user to consider when creating their designs. While the SerDes lane count does not directly affect the node parameters, it affects which MACs will be available, particularly after considering any previous MAC allocations.

ACX_ETHERNET_NODE

The ACX_ETHERNET_NODE is a key module in Ethernet designs, which is crucial in configuring the EIU, among other functionalities. Below is the module definition.

For detailed configuration of the ACX_ETHERNET_NODE, refer to the ***Speedster7t Component Library User Guide (UG086)***[6]

```
module ACX_ETHERNET_NODE #(
    // Configuration parameters.  These must be set to a value
    parameter               tx_enable           = 1'hx,
    parameter               rx_enable           = 1'hx,
    parameter [3:0]         tx_mode             = 4'hx,
    parameter [3:0]         rx_mode             = 4'hx,
    parameter [1:0]         tx_mac_id           = 2'hx,
    parameter [1:0]         rx_mac_id           = 2'hx,
    parameter [4:0]         tx_eiu_channel      = 5'hx,
    parameter [4:0]         rx_eiu_channel      = 5'hx,
    parameter [31:0]        tx_threshold        = 32'hxxxx_xxxx,
    parameter [31:0]        rx_threshold        = 32'hxxxx_xxxx,
    parameter real          nap_tx_frequency    = 507.0,
    parameter integer       fcs_length          = 4         // Default is for MAC to add
and remove FCS
                                                            // If fabric is adding and
removing FCS, set this to 0 in the instantiation
)
(
    input  wire             clk,
    input  wire             rstn,
    output wire             output_rstn,

    // TX port
```

6 https://www.achronix.com/documentation/speedster7t-component-library-user-guide-ug086

```
    output logic            tx_ready,
    output logic            ts_enable,
    input  wire             tx_valid,
    input  wire             tx_sop,
    input  wire             tx_eop,
    input  wire [293 -1:0]  tx_data,
    input  wire [4   -1:0]  tx_dest,

    // RX port
    input  wire             rx_ready,
    output logic            rx_valid,
    output logic            rx_sop,
    output logic            rx_eop,
    output logic [293 -1:0] rx_data,
    output logic [4   -1:0] rx_src
);
```

Core ACX_ETHERNET_NODE Configuration Parameters:

In setting up the ACX_ETHERNET_NODE for Ethernet designs, consider the following parameters:

- **Ethernet Mode (tx_mode[3:0] and rx_mode[3:0]):** These ports designate the transmission and reception modes, respectively.
- **MAC ID (tx_mac_id[1:0] and rx_mac_id[1:0]):** These ports determine the MAC ID for transmission and reception.
- **EIU Channel Mapping (tx_eiu_channel[4:0] and rx_eiu_channel[4:0]):** These ports allocate specific EIU channels for transmission and reception. The channels range from 0 to 31.

> **Note**
>
> Some channels might require an offset to these base addresses to get the exact channel number.

To help assign the required values to these parameters in the ACX_ETHERNET_NODE, users can refer to the eth_utils.svh file. This file provides a templated parameter definition for the various ACX_ETHERNET_NODE configurations.

From eth_utils.svh:

```
// ----------------------------------------------------------------------------
// NAP_ETHERNET parameter defines
// ----------------------------------------------------------------------------
// TX and RX modes
`define ACX_ETH_MODE_10G        4'b0000
`define ACX_ETH_MODE_25G        4'b0001
`define ACX_ETH_MODE_40G        4'b0010
`define ACX_ETH_MODE_50G        4'b0011
`define ACX_ETH_MODE_100G       4'b0100
`define ACX_ETH_MODE_200G_QSI   4'b0110
```

```
`define ACX_ETH_MODE_400G_QSI   4'b1000

// MAC IDs
`define ACX_ETH_400G_MAC0       2'b00
`define ACX_ETH_400G_MAC1       2'b01
`define ACX_ETH_QUAD0           2'b10
`define ACX_ETH_QUAD1           2'b11

// EIU channels
// Define base addresses only
// For multichannel,  as will use addition to calculate the intermediate values
// NOTE : Values are in decimal not hex
`define ACX_ETH_CH_SUB0         5'd0
`define ACX_ETH_CH_SUB1         5'd8
`define ACX_ETH_CH_QUAD0_PRE0   5'd16
`define ACX_ETH_CH_QUAD0_EXP0   5'd20
`define ACX_ETH_CH_QUAD1_PRE0   5'd24
`define ACX_ETH_CH_QUAD1_EXP0   5'd28
```

For the full `eth_utils.svh` file containing additional Verilog defines essential for Ethernet interface configurations, users can refer to the Ethernet reference designs via the KBA, **How do I Download Demonstration and Reference Designs?**[7] (login required).

After selecting the Ethernet Mode and MAC ID based on design requirements, it's important to determine the EIU channel mappings which depend on the selected mode and MAC ID.

EIU Channel Mapping

When the user is working with the `ACX_ETHERNET_NODE` in their FPGA designs, a crucial aspect to understand and configure is the Ethernet Interface Unit (EIU) channel mapping. Proper channel mapping ensures that traffic between the NAPs and Ethernet MACs is routed without conflicts.

EIU Channels

Each Ethernet subsystem contains an EIU, which links the traffic from the NAPs to the Ethernet MACs. Each EIU consists of 32 channels. Each channel must be exclusively assigned to a single NAP. The channels are detailed below.

Ethernet Modes	Name	Lane Type	Function	Channel Number	Comments
200G_QSI, 400G_QSI	Buffer SubSystem0	-	400G/200G buffering and packet structuring	0 to 7	400/200G MAC 0, or 200G Quad MAC 0 NAPs

7 https://support.achronix.com/hc/en-us/articles/38065028969236-How-do-I-Download-Demonstration-and-Reference-Designs

Ethernet Modes	Name	Lane Type	Function	Channel Number	Comments
	Buffer SubSystem1			8 to 15	400/200G MAC 1, or 200G Quad MAC 1 NAPs
10G to 100G pass-through	Quad MAC 0	Pre-emptive	10G to 100G pass-through	16 to 19	Each Ethernet channel has a single NAP and must be assigned to a single EIU channel.
		Express		20 to 23	
	Quad MAC 1	Pre-emptive	10G to 100G pass-through	24 to 27	
		Express		28 to 31	

Table 19 • EIU Channel Functions

Based on the user's choice of Ethernet Mode, Buffer Subsystem (200G and 400G modes) or MAC number, and Pre-emptive or Express lanes, assign the corresponding Channel Number to the node configuration parameter.

Ethernet Configuration Overview

This guide provides the major steps for users on how to configure various Ethernet rates using the `ACX_ETHERNET_NODE`. Within each example, users will learn how to select the desired Ethernet rate in the ACE IP editor, understand the MAC-to-lane mapping, utilize the `nap_ethernet_wrapper` for instantiation, and correctly set parameters. For a comprehensive understanding of all parameter definitions, users should always refer to the `eth_utils.svh` file.

Example: Configuring the 100G (4 × 25G) Ethernet Rate

IP Configuration

When setting up a 100G (4 × 25G) Ethernet configuration, the user will start by selecting the desired Ethernet rate within the ACE IP configuration editor.

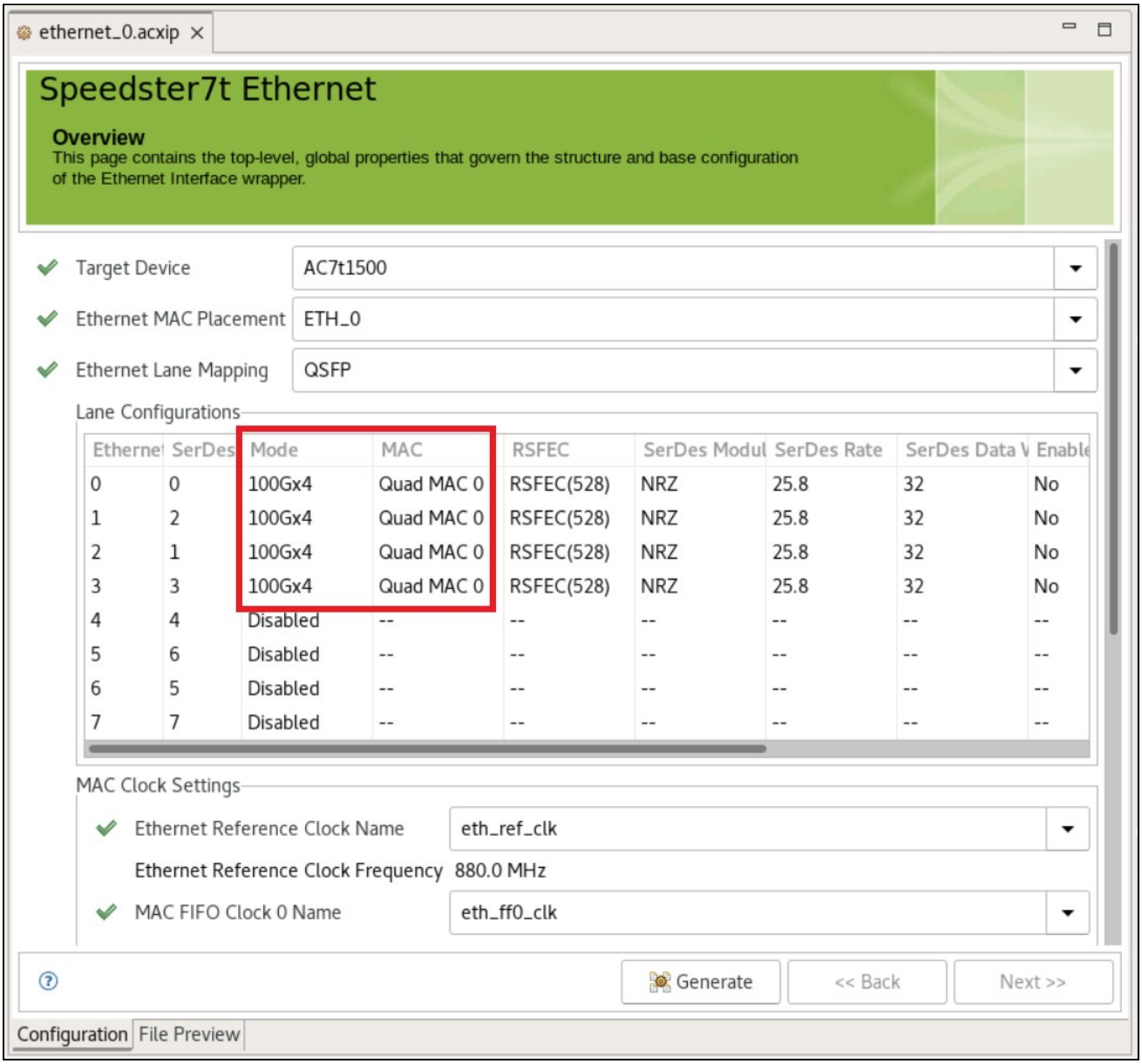


Figure 8 • Ethernet IP Mode Selection

After specifying the 100G rate, the MAC column in the image above will indicate which of the four internal MACs are mapped to this Ethernet lane. It's essential to note that this MAC column is read-only, as a reference to guide the user on the MAC-to-lane mapping. With this mapping knowledge, the user can then proceed with customizing and configuration of the ACX_ETHERNET_NODE.

Instantiating *ACX_ETHERNET_NODE* Configured for 100G (4 × 25G) using the Wrapper:

For users looking to instantiate the ACX_ETHERNET_NODE, the nap_ethernet_wrapper offers a streamlined process. Using this wrapper, the user can easily set the mode, ID, and channel parameters. Additionally, this wrapper is a part of the Achronix reference designs, making integration more straightforward. The user can refer to the provided Achronix reference designs on via the KBA, **How do I Download Demonstration and Reference Designs?**[8] (login required).

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_100G),
    .RX_MODE                (`ACX_ETH_MODE_100G),
    .TX_MAC_ID              (`ACX_ETH_QUAD0),
    .RX_MAC_ID              (`ACX_ETH_QUAD0),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD0_EXP0),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD0_EXP0),
    ...
) i_nap_eth_lb (...);
```

Ensure the user defines the appropriate mode, ID, and EIU channels to route data between the NAP and the Ethernet MACs correctly:

```
// ----------------------------------------------------------------------------
// NAP_ETHERNET parameter defines
// ----------------------------------------------------------------------------
...

`define ACX_ETH_MODE_100G       4'b0100
`define ACX_ETH_QUAD0           2'b10
`define ACX_ETH_CH_QUAD0_EXP0   5'd20
```

Notice how the ACX_ETH_CH_QUAD0_EXP0 specifically represents the base address of the EIU channel for the Quad MAC 0, Express Lanes configuration, which supports 10G to 100G pass-through. It's important to note that the defines shown here are only a subset for this particular ACX_ETHERNET_NODE instance and the complete set can be found in the eth_utils.svh file.

Example: Configuring Two Channels of 200G (4 × 50G) Ethernet Rate

The two-channel 200G QSI mode configuration is a unique scenario due to its specific requirements for EIU buffer channels. This mode requires careful configuration, ensuring that data traffic is directed to the designated buffer channels within the EIU.

8 https://support.achronix.com/hc/en-us/articles/38065028969236-How-do-I-Download-Demonstration-and-Reference-Designs

IP Configuration

For a dual 200G (4 × 50G) Ethernet configuration, users should begin by selecting the desired Ethernet rate in the ACE IP configuration editor.

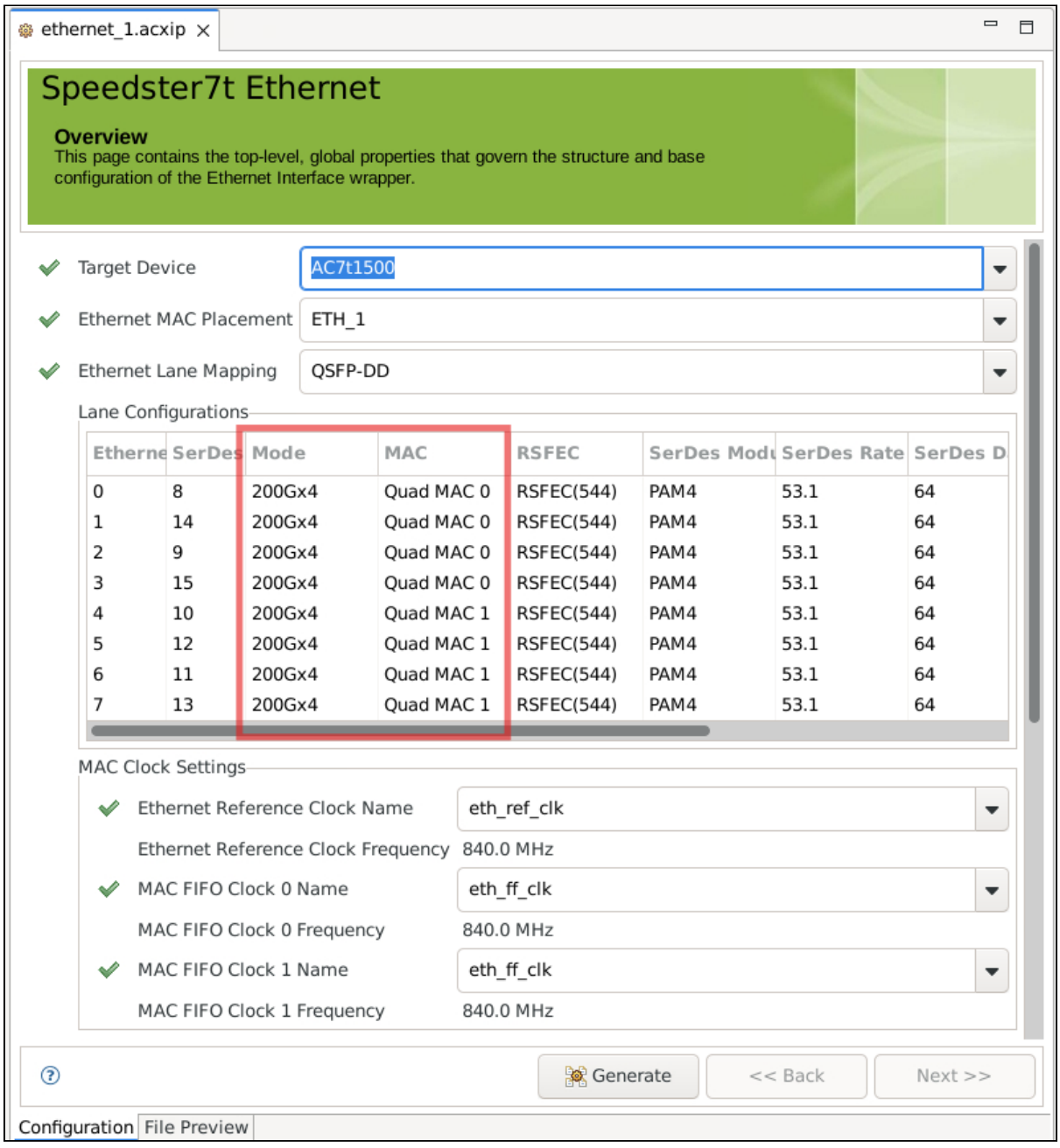


Figure 9 • Ethernet IP Mode Selection

After specifying the 200G rate twice, for each channel, the MAC column in the image above will indicate which of the four internal MACs are mapped to each Ethernet lane. The first selection will typically map to MAC0 and the second selection to MAC1. The user needs to understand that this MAC column is read-only and serves as a reference for the MAC-to-lane mapping. With this clear mapping, the user can proceed with the customization and configuration of the ACX_ETHERNET_NODE using the nap_ethernet_wrapper.

Instantiating ACX_ETHERNET_NODE Configured for Two Channels of 200G (4 × 50G) using the Wrapper:

For the first channel, using ACX_ETH_QUAD0 :

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_200G_QSI),
    .RX_MODE                (`ACX_ETH_MODE_200G_QSI),
    .TX_MAC_ID              (`ACX_ETH_QUAD0),
    .RX_MAC_ID              (`ACX_ETH_QUAD0),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
    ...
) i_nap_eth_lb0 (...);
```

For the second channel, using ACX_ETH_QUAD1 :

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_200G_QSI),
    .RX_MODE                (`ACX_ETH_MODE_200G_QSI),
    .TX_MAC_ID              (`ACX_ETH_QUAD1),
    .RX_MAC_ID              (`ACX_ETH_QUAD1),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
    ...
) i_nap_eth_lb1 (...);
```

Ensure the user defines the appropriate mode, ID, and EIU channels to route data between the NAP and the Ethernet MACs correctly:

```
// ----------------------------------------------------------------------------
// NAP_ETHERNET parameter defines
// ----------------------------------------------------------------------------
...
`define ACX_ETH_MODE_200G_QSI   4'b0110
`define ACX_ETH_QUAD0           2'b10
`define ACX_ETH_QUAD1           2'b11
`define ACX_ETH_CH_SUB0         5'd0
```

ACX_ETH_CH_SUB0 is defined as the base address, and its value is 5'd0 (which translates to decimal 0). In the wrapper instantiation, the user will notice:

```
.TX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
.RX_EIU_CHANNEL         (`ACX_ETH_CH_SUB0+2),
```

The +2 offset added to the ACX_ETH_CH_SUB0 is significant as it shifts from the base address of the EIU buffer channels to channels [3:2]. This is crucial because, for the 200G packet mode, the EIU buffer channels [3:2] are required. By adding this offset, it ensures that the data traffic for the 200G packet mode is directed to the appropriate buffer channels within the EIU.

Note

The defines shown here are only a subset for this particular ACX_ETHERNET_NODE instance and the complete set can be found in the eth_utils.svh file.

Example: Configuring Four Channels of Mixed Ethernet Rates

This example will instantiate 4 channels of mixed rates: 100G × 4, 50G × 2, and two 10G × 1 channels.

IP Configuration

For a dual 200G (4 × 50G) Ethernet configuration, users should begin by selecting the desired Ethernet rate in the ACE IP configuration editor.

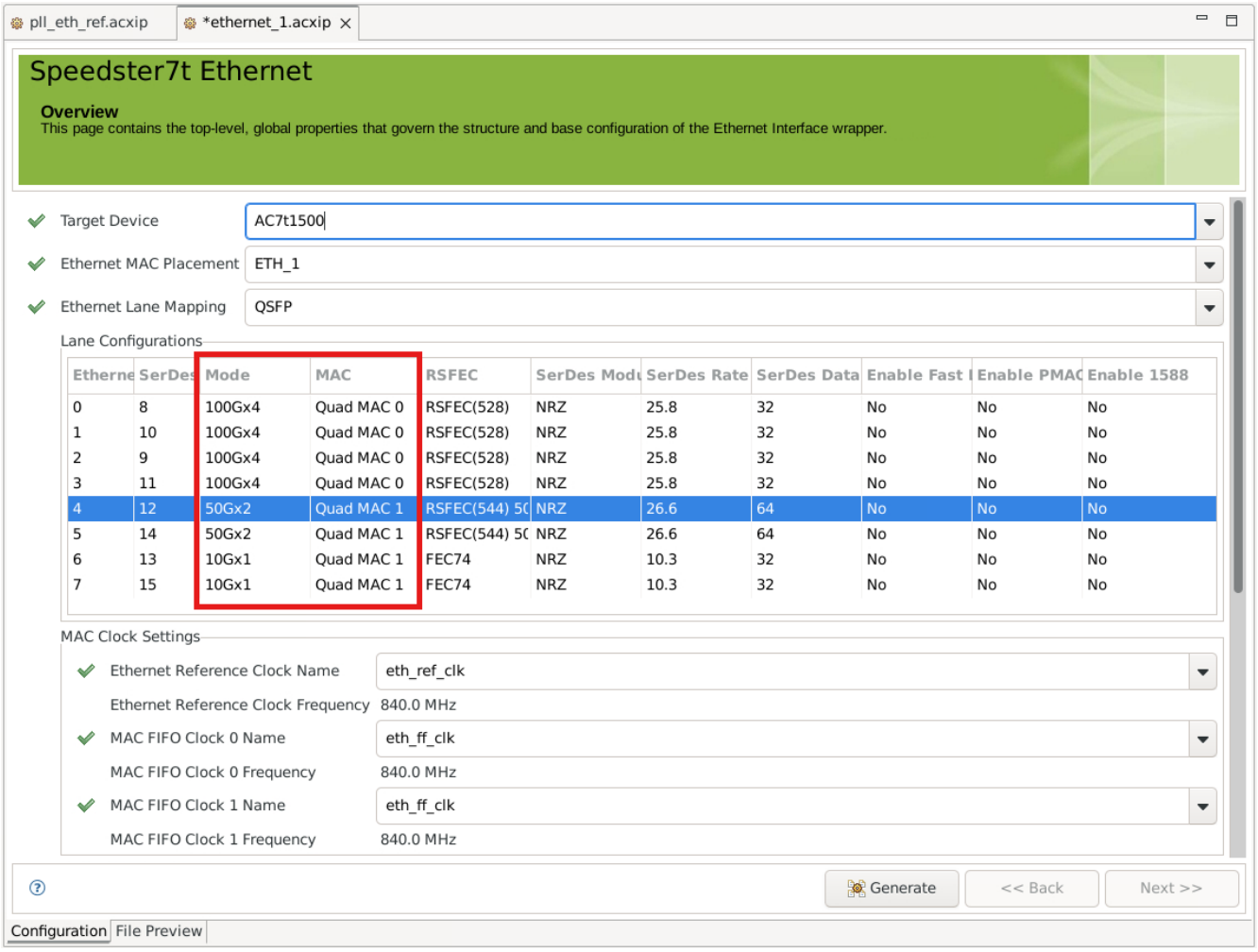


Figure 10 • Ethernet IP Mode Selection

After specifying the 4 channels, the MAC column in the image above will indicate which of the four internal MACs are mapped to each Ethernet lane. The user needs to understand that this MAC column is read-only and serves as a reference for the MAC-to-lane mapping. With this clear mapping, the user can proceed with the customization and configuration of the `ACX_ETHERNET_NODE` using the `nap_ethernet_wrapper`.

Instantiating `ACX_ETHERNET_NODE` Configured for the four channels using the Wrapper:

For the first channel, use `ACX_ETH_QUAD0` :

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_100G),
```

```
    .RX_MODE                (`ACX_ETH_MODE_100G),
    .TX_MAC_ID              (`ACX_ETH_QUAD0),
    .RX_MAC_ID              (`ACX_ETH_QUAD0),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD0_EXP0),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD0_EXP0),
    ...
) i_nap_eth_lb0 (...);
```

For the next three channels, use `ACX_ETH_QUAD1` :

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_50G),
    .RX_MODE                (`ACX_ETH_MODE_50G),
    .TX_MAC_ID              (`ACX_ETH_QUAD1),
    .RX_MAC_ID              (`ACX_ETH_QUAD1),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    ...
) i_nap_eth_lb1 (...);
```

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_10G),
    .RX_MODE                (`ACX_ETH_MODE_10G),
    .TX_MAC_ID              (`ACX_ETH_QUAD1),
    .RX_MAC_ID              (`ACX_ETH_QUAD1),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    ...
) i_nap_eth_lb2 (...);
```

```
nap_ethernet_wrapper #(
    ...
    .TX_MODE                (`ACX_ETH_MODE_10G),
    .RX_MODE                (`ACX_ETH_MODE_10G),
    .TX_MAC_ID              (`ACX_ETH_QUAD1),
    .RX_MAC_ID              (`ACX_ETH_QUAD1),
    .TX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    .RX_EIU_CHANNEL         (`ACX_ETH_CH_QUAD1_EXP0),
    ...
) i_nap_eth_lb3 (...);
```

Ensure the user defines the appropriate mode, ID, and EIU channels to route data between the NAP and the Ethernet MACs correctly:

```
// ----------------------------------------------------------------------------
// NAP_ETHERNET parameter defines
// ----------------------------------------------------------------------------
...
`define ACX_ETH_MODE_10G        4'b0000
`define ACX_ETH_MODE_50G        4'b0011
`define ACX_ETH_MODE_100G       4'b0100
`define ACX_ETH_QUAD0           2'b10
`define ACX_ETH_QUAD1           2'b11
`define ACX_ETH_CH_QUAD0_EXP0   5'd20
`define ACX_ETH_CH_QUAD1_EXP0   5'd28
```

Notice how the ACX_ETH_CH_QUAD0_EXP0 specifically represents the base address of the EIU channel for the Quad MAC 0, Express Lanes configuration, which supports 10G to 100G pass-through.

Note

The defines shown here are only a subset for this particular ACX_ETHERNET_NODE instance and the complete set can be found in the `eth_utils.svh` file.

Chapter 6 : Register Maps

In order to support configuration and status monitoring of the Ethernet subsystem, the configuration and status registers (CSR) are located within the overall device memory map. The memory area can be accessed through a number of mechanisms, including:

- PCIe subsystem
- JTAG port
- `NAP_AXI_MASTER` instantiated in the programmable fabric

Global Address Space

CSR Addressing

The Speedster7t FPGA supports a 42-bit address region. Within that address region, bits [41:34] define the top level spaces. The CSR register space is defined as shown in the table below.

Table 20 • Speedster7t Global Address Map

Address Bit	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	...	0
Destination	0	0	1	0	0	0	0	0	Target ID						IP ID				Register Address			

Target ID Addressing

The Ethernet subsystem `Target ID` field (address bits [33:28]) is defined as follows:

Table 21 • Ethernet Target ID Address Map

Target ID							Description
Address Bit	**33**	**32**	**31**	**30**	**29**	**28**	
Ethernet 0	0	1	1	0	1	1	Ethernet 0 control and status registers.
Ethernet 1	0	1	1	1	0	0	Ethernet 1 control and status registers.

IP ID Addressing

Each Ethernet subsystem is comprised of multiple MAC, PCS and SerDes elements. Each of these elements has an individual register map for programming. The individual register maps are addressed using the `IP_ID` address field (address bits [27:24]).

Table 22 • Ethernet IP ID Address Map

IP ID					Description
Address Bit	**27**	**26**	**25**	**24**	
System Control	0	0	0	0	Select clock and reset sources.
400G PCS	0	0	0	1	Common PCS block for 400/200G MACs.
Quad 0 PCS	0	0	1	0	PCS for Quad 0 MAC.
Quad 1 PCS	0	0	1	1	PCS for Quad 1 MAC.
400G MAC 0	0	1	0	0	400/200G MAC 0.
400G MAC 1	0	1	0	1	400/200G MAC 1.
Quad 0 MAC	0	1	1	0	Quad 0 MAC.
Quad 1 MAC	0	1	1	1	Quad 1 MAC.
SerDes Quad 0	1	0	0	0	SerDes Quad 0 (lanes 0-3).
SerDes Quad 1	1	0	0	1	SerDes Quad 1 (lanes 4-7).

Address Dictionary

To simplify access to the device memory space, there is an address dictionary which divides the memory areas into tokens. This dictionary is used to access memory areas and individual registers using names, rather than direct addresses. This token approach results in more readable code while reducing the need to remember individual addresses.

The address dictionary is included as part of ACE, accessed using the Tcl console, and used to access the memory space via the JTAG port. The tokens for the Ethernet subsystem address areas listed in the table (page 52), above, are as follows:

Table 23 • Address Dictionary Tokens

Top Level	Subsystem	IP ID
CSR_SPACE	ETHERNET_x (1)	EIU_CFG
	ETHERNET_x (1)	400G_PCS
	ETHERNET_x (1)	QUAD_PCS_0
	ETHERNET_x (1)	QUAD_PCS_1
	ETHERNET_x (1)	400G_MAC_0
	ETHERNET_x (1)	400G_MAC_1
	ETHERNET_x (1)	QUAD_MAC_0
	ETHERNET_x (1)	QUAD_MAC_1
	ETHERNET_x (1)	SERDES_0
	ETHERNET_x (1)	SERDES_1

Table Notes

1. In systems with more than one Ethernet subsystem, this token represents the subsystem number (e.g., ETHERNET_0, ETHERNET_1).

The equivalent address dictionary tokens are presented alongside the absolute address in the register definitions below.

System Control

The system control registers manage selection of clock and reset sources for the various modules of the Ethernet subsystem.

Base Address

The Ethernet subsystem system control registers are located at:

- Ethernet Subsystem 0 : 0x081_b000_0000 : "CSR_SPACE ETHERNET_0 CFG_EIU"

- Ethernet Subsystem 1 : `0x081_c000_0000` : `"CSR_SPACE ETHERNET_1 CFG_EIU"`

400G MAC Register Map

The two Quad MACs have identical register maps as detailed below.

Base Address

The Ethernet subsystem 400G MACs are located at:

- Ethernet Subsystem 0, 400G MAC 0 – `0x081_b400_0000` : `"CSR_SPACE ETHERNET_0 400G_MAC_0"`
- Ethernet Subsystem 0, 400G MAC 1 – `0x081_b500_0000` : `"CSR_SPACE ETHERNET_0 400G_MAC_1"`
- Ethernet Subsystem 1, 400G MAC 0 – `0x081_c400_0000` : `"CSR_SPACE ETHERNET_1 400G_MAC_0"`
- Ethernet Subsystem 1, 400G MAC 1 – `0x081_c500_0000` : `"CSR_SPACE ETHERNET_1 400G_MAC_1"`

Register Definitions

Table 24 • REVISION (Offset : 0x00_0000)

Name	Bits	Access	Default	Description
Revision	[31:0]	RO	`0xXX`	Indicates 400G MAC Revision.

Table 25 • SCRATCH (Offset : 0x00_0004)

Name	Bits	Access	Default	Description
Scratch	[31:0]	RW	`0x0000_0000`	Scratch Register to test read and write access.

Table 26 • COMMAND_CONFIG (Offset : 0x00_0008)

Name	Bits	Access	Default	Description
INV_LOOP	31	RW	0	Enable inverse loopback. When set to '1', data from the Rx FIFO interface is looped back to the Tx FIFO for test/debug purposes. When enabled, the input signals ff_rx_rdy and ff_tx_crc have no meaning. The input signal ff_tx_crc_ovr can be used to instruct the Tx MAC to re-calculate the CRC of the looped frames. **Note** This is a test feature only and has limitations depending on available FIFO depth and frame sizes. In general only short frames can be transferred through the inverse loopback. Frames exceeding 96 bytes (2 segments) may get corrupted occasionally by the loopback, hence should be avoided during such testing.
Reserved	[30:28]	RO	0	Unused.
FLT_HDL_DIS	27	RW	0	• **1 –** Disables RS fault handling. Incoming fault sequences are ignored and have no effect on the transmitter. • **0** – Default, the MAC RS automatically inserts remote faults and idles in egress direction on detection of local faults and remote faults, respectively, on ingress direction. In addition, the MAC transmitter is stopped and does not read frames from the transmit FIFO as long as the fault condition persists.
TX_FIFO_RESET (1)	26	RW, SC	0	Self-Clearing TX FIFO reset command. When set, the TX FIFO is reset, discarding any frames that might have been stored in the FIFO. The bit is cleared when the reset sequence has completed. To avoid outgoing frame corruption, the FIFO reset should be performed after the TX has been disabled (TX_ENA = 0) and no transmission is ongoing.
Reserved	[25:23]	RO	3'b000	Unused.
TX_FLUSH (2)	22	RW	0	Egress Flush Enable: • **1** – The Core reads out the TX FIFO and drops the data (data is not sent out on the line). The associated pause signals (link pause or priority flow control) are masked. In addition, statistics and timestamp return indications might not occur for the flushed frames. • **0** – Normal MAC operation.
RX_SFD_ANY (3)	21	RW	0	Disable check of SFD (0xd5) character at frame start: • **1** – Frame is accepted even if the SFD byte following the preamble is not 0xd5. • **0** – The frame is accepted only if the SFD byte is found with value 0xd5.

Name	Bits	Access	Default	Description
PAUSE_PFC_COMP	20	RW	0	Link Pause Compatibility with PFC Mode: • **1** – Pause frames in legacy pause mode are processed similar to PFC frames in PFC mode, i.e., the transmit path is not paused by incoming pause frames, only the pause status is asserted as long as the internal pause timer has not expired. This bit is relevant only when PFC_MODE = 0. • **0** – Pause frames are processed.
PFC_MODE	19	RW	0	Enable Priority Flow Control (PFC) mode of operation: • **1** – The MAC transmits and accepts PFC frames. • **0** – The MAC uses standard Link Pause frames.
Reserved	18	RW		Unused.
NO_LGTH_CHECK	17	RW		Disable Payload Length Check. • **1** – The payload length check is disabled. • **0** – The Core checks the frame's payload length with the frame's Length/Type field.
SEND_IDLE (4)	16	RW	0	• **1** – Suppresses any frame transmissions and forces IDLE on the transmit interface instead of frames. This control affects the MAC reconciliation layer (RS) which acts after the MAC datapath has processed the frame. • **0** – Normal Operation.
Reserved	[15:14]	RW	2'b00	Unused.
CMD_FRAME_ENA	13	RW	0	MAC Control Frame Enable: • **1** – MAC Control frames (type 0x8808) with any opcode other than 0x0001 when PFC_MODE = 0 or other than 0x0101 when PFC_MODE = 1 are accepted and forwarded to the Client interface. • **0** – MAC Control frames with any opcode other than 0x0001 (when PFC_MODE = 0) or other than 0x0101 (when PFC_MODE = 1) are silently discarded.
SW_RESET (5)	12	RW, SC	0	MAC Software Reset. Self clearing bit: • **1** – Resets all statistic counters as well as datapath functions. This should be issued after all traffic has been stopped as a result of clearing RX_ENA and TX_ENA. • **0** – Normal operation.
TX_PAD_EN	11	RW	1	Enable padding of frames in transmit direction: • **1** – Normal operation. All frames from the EIU which are less than 64-bytes must also have crc_insert set in the TX flags, otherwise the frame is sent as is which potentially could result in an invalid frame length. • **0** – The MAC does not extend frames received from the EIU to a minimum of 64 bytes, allowing for transmission of too short frames (violating the Ethernet minimum size requirement).

Name	Bits	Access	Default	Description
LOOPBACK_EN	10	RW	0	Enable PHY Interface Loopback. If set to '1', the signal loop_ena is set to '1'. If set to '0' (Reset value), the signal loop_ena is set to '0'. Has no function within the MAC besides controlling the top-level signal.
TX_ADDR_INS	9	RW	0	• **1** – Replace the frame source MAC address with the address programmed in MAC_ADDR_0 and MAC_ADDR_1. • **0** – Frame is unmodified.
PAUSE_IGNORE	8	RW	0	Ignore Pause Frame Quanta: • **1** – Received pause frames are ignored by the MAC. • **0** – The transmit process is stopped for the amount of time specified in the pause quanta received within a pause frame.
PAUSE_FWD	7	RW	0	Terminate/Forward Pause Frames: • **1** – Pause frames are forwarded to the user application. • **0** – Normal mode. Pause frames are terminated and discarded within the MAC.
CRC_FWD	6	RW	0	Terminate/Forward Received CRC: • **1** – Enabled, the CRC field of received frames is forwarded with the frame to the user application. • **0** – Disabled (set to reset value 0), the CRC field is stripped from the frame.
PAD_EN	5	RO	0	Always fixed to 0 (MAC cannot remove Rx padding)
PROMIS_EN	4	RW	0	• **1** – Enables MAC promiscuous operation. All frames are received without any MAC address filtering. • **0** – Disables MAC promiscuous operation.
Reserved	3:2	RO	00	Unused.
RX_ENA	1	RW	0	MAC Receive Path Enable: • **1** – Enable the MAC receive path. • **0** – Disable the MAC receive path.
TX_ENA	0	RW	0	MAC Transmit Path Enable: • **1** – Enable the MAC transmit path. • **0** – Disable the MAC transmit path.

Table Notes

1. When the credit-based FIFO is used, the credit return logic is also reset and the application must reinitialize its credit counter to maximum.
2. Egress flush operates independent of the RX link state.
3. For 802.3br operation, this bit must be set to 0. Can be set to 1 for normal MAC operation when 802.3br traffic is not expected.
4. This bit does not have an effect on fault handling (i.e., reception of local fault still causes transmit of remote fault).
5. Performing the SW reset can lead to an RX exception to the EIU if the reset is issued in the middle of a receive frame transfer. The end of packet is lost and the EIU might need to be reset in order to continue to receive frames.

Table 27 • MAC_ADDR_0 (Offset : 0x00_000C)

Name	Bits	Access	Default	Description
MAC_ADDR_0	[31:0]	RW	0x0000_0000	The lower 32-bits of the 48-Bit MAC Address. Bit 0 is LSB.

Table 28 • MAC_ADDR_1 (Offset : 0x00_0010)

Name	Bits	Access	Default	Description
MAC_ADDR_1	[31:16]	RW	0x0000	Unused and always set to 0.
	[15:0]	RW	0x0000	The upper 16-bits of the 48-Bit MAC Address. Bit 0 is Bit 32 of MAC address.

Table 29 • FRM_LENGTH / TX_MTU (Offset : 0x00_0014)

Name	Bits	Access	Default	Description
TX_FRM_LENGTH (1)	[31:16]	RW	0x0000	Defines the maximum length of a transmitted frame. When set to 0x0000, the value of RX_FRM_LENGTH is used, so transmit and receive frames have the same maximum length.
RX_FRM_LENGTH	[15:0]	RW	0x0600	Maximum supported RX frame length. When a frame exceeds the given length specified, the MAC RX truncates the frame as described in Frame Truncation (page 58). The MAC supports any frame size up to 32736 bytes (0x7FE0). Set to 1536 (0x0600) after reset.

Table Notes

1. This value is used for statistics purposes only and has no effect on the MAC transmit operation.

Frame Truncation

In the receive direction, the MAC is always checking the received frame length (total octets following the preamble and SFD) against the configured FRM_LENGTH value. If the frame exceeds the programmed value, the frame is truncated and provided to the NAP with the length error flag set. See Receive Flags (page 17). When frame truncation happens, the receive FIFO write might stop prior to the actual truncation point. Hence, the frame delivered to the EIU can be shorter than the given FRM_LENGTH but the truncation is always executed only when the exact limit is reached. In addition, the last octets of a truncated frame can contain arbitrary data.

Another cause of frame truncation can be a receive FIFO almost full condition during reception which is reported accordingly in the frame receive status.

Table 30 • RX_FIFO_SECTIONS (Offset : 0x00_001C)

Name	Bits	Access	Default	Description
RX_FIFO_SECTION	[31:0]	RW	0x0000_0000	• **[15:0]** – RX section available threshold. When the receive FIFO level reaches or exceeds this value, the FIFO signals to the EIU that data is available, either a complete frame or at least the threshold amount. When asserted, the data is transferred to the EIU until end of frame is reached. To enable store and forward on the receive path, set this value to 0. In this case, data is only transferred when a complete frame is available in the receive FIFO. Default: 0x08. • **[31:16]** – RX section empty threshold. When the receive FIFO level reaches or exceeds this value, an indication is provided to the MAC transmitter which generates an XOFF Pause frame to indicate congestion to the remote Ethernet transmitter. When the receive FIFO level drops below this value, the MAC transmits an XON Pause frame to indicate that congestion has been cleared and that the remote Ethernet transmitter may continue. A value of 0 disables the function, and no Pause frames are transmitted. Default: 0x00. All threshold values are in steps of FIFO words.

Table 31 • TX_FIFO_SECTIONS (Offset : 0x00_0020)

Name	Bits	Access	Default	Description
TX_FIFO_SECTION	[31:0]	RW	0x0000_0007	• **[15:0]** – TX section available threshold, default 0x8. When the transmit FIFO level reaches this value, the MAC commences transmission of the data in the FIFO, continuing until the end of frame is reached. Independent of this value, if a complete frame is stored in the FIFO, transmission always starts. To operate in a cut through mode, the value must be set greater than or equal to 4. A value of 0 configures store-and-forward operation with the MAC only transmitting when a complete frame is stored in the FIFO. In this scenario, the user design must ensure that all frames transmitted are smaller than the FIFO size. If too large a frame is sent, then a deadlock occurs which is only cleared by setting this threshold to a non-zero value. • **[31:16]** – TX section empty threshold, default 0x0. When the transmit FIFO level reaches this value, the EIU is signaled that the FIFO is almost full. This causes the EIU to de-assert ready to the respective NAP column. This could affect other NAPs transmitting on the column, possibly leading to data corruption due to data starvation on the other channel transmit FIFOs. A value of 0 disables the signalling of almost empty. All threshold values are in steps of FIFO words.

Table 32 • MAC_STATUS (Offset : 0x00_0040)

Name	Bits	Access	Default	Description
Reserved	31:9	RO	0	Unused.

Name	Bits	Access	Default	Description
TX_ISIDLE	8	RO	0	TX MAC idle indicator: • **1** – TX MAC is in idle. • **0** – TX MAC is not in idle. Toggles during normal operation when the MAC is transmitting (might assert during IPG times).
Reserved	7	ROR	0	Unused.
RX_EMPTY	6	RO	0	MAC RX FIFO empty indicator: • **1** – RX FIFO is empty. • **0** – RX FIFO has data.
TX_EMPTY	5	RO	0	MAC TX FIFO empty indicator: • **1** – TX FIFO is empty. • **0** – TX FIFO has data.
Reserved	4	RO	0	Unused.
TS_AVAIL	3	RW	0	Transmit Timestamp Available. Indicates that the timestamp of the last transmitted event frame (which had the transmit flag Frame set, (see Transmit Flags (page 9)) is available in the register TS_TIMESTAMP. To clear TS_AVAIL, the bit must be written with a 1.
PHY_LOS	2	RO	0	PHY indicates loss of signal indicator. Represents value of pin "phy_los". • **1** – PHY indicates loss-of-signal. • **0** – PHY is operating normally.
RX_REM_FAULT	1	ROR	0	Latch-High Remote Fault status: • **1** – the MAC detected Remote Fault Sequences on the CDfMII receive interface. • **0** – reset value after read or reset.
RX_LOC_FAULT	0	ROR	0	Latch-High Local Fault status: • **1** – the MAC detected Local Fault Sequences on the CDMII receive interface. • **0** – Reset value after read or reset.

Table 33 • TX_IPG_LENGTH (Offset : 0x00_0044)

Name	Bits	Access	Default	Description
TX_IPG_LENGTH (1)	[31:0]	RW	0x0000_000C	Transmit Inter-Packet-Gap (IPG) options: • [15:0] – reserved, write 12 always. • [18:16] – reserved, write 0 always. • [31:19] – Set IPG compensation count, increasing bandwidth. Supported values are: **0x0000** – IPG bias mode disabled, no idle blocks removed. **0x0800** – One 8-byte block of idle is removed after every 16384 blocks. **0x0A00** – One 8-byte block of idle is removed after every 20480 blocks.

Table Notes

1. The minimal required clock frequencies must be increased accordingly to support the additional bandwidth if biasing is used.

Table 34 • TS_TIMESTAMP (Offset : 0x00_007C)

Name	Bits	Access	Default	Description
TS_TIMESTAMP	[31:0]	RO	0x0000_0000	Timestamp of the last frame transmitted by the MAC that had the ff_tx_ts_frm signal asserted from the user application. Valid when the status bit STATUS(TS_AVAIL) is set to 1'b1.

Table 35 • XIF_MODE (Offset : 0x00_0080)

Name	Bits	Access	Default	Description
Reserved	[31:6]	RO	26'b0	Unused.
ONE_STEP_ENA	5	RW	1'b0	Enable 1-step IEEE-1588 functions. • **1** – The user application can instruct the MAC to perform on-the-fly correction field updates with the one-step vector signals. See IEEE Timestamping (page 96). • **0** – When disabled (0), the one-step functions are all disabled and a lower latency data path is enabled. Any user application 1-step requests are ignored. Changing of the one-step enable is allowed only when the TX FIFO is empty.
Reserved	[4:0]	RO	5'b0000	Unused.

400G PCS Register Map

Base Address

The Ethernet subsystem 400G PCSs are located at:

- Ethernet Subsystem 0, 400G PCS – `0x081_b100_0000` : "`CSR_SPACE ETHERNET_0 400G_PCS`"
- Ethernet Subsystem 1, 400G PCS – `0x081_c200_0000` : "`CSR_SPACE ETHERNET_1 400G_PCS`"

Register Definitions

Table 36 • CONTROL1 (Offset : 0x00_0000)

Name	Bits	Access	Default	Description
`CONTROL1`	15	RW, SC	`0`	PCS Reset: • **1 –** assert. • **0** – de-assert.
	14	RW	`0`	Loopback: • **1** – enable. • **0** – disable.
	13	RO	0	Speed selection. If [13,6] =0x11, then bits [5:2] select speed.
	12	RO	0	Reserved.
	11	RW	0	Low power: • **1 –** Normal Operation • **0 –** Low power mode
	10:7	RO	`0`	Reserved.
	6	RO	0	Speed selection. If [13,6] =0x11, then bits [5:2] select speed.
	5:2	[5:3] RO [2] RW	`4'b0000`	Speed selection: • **1 0 1 0** – 400 Gbps. • **1 0 0 1** – 200 Gbps.

Name	Bits	Access	Default	Description
	1:0	RO	2'b00	Reserved.

Table 37 • STATUS1 (Offset : 0x00_0004)

Name	Bits	Access	Default	Description
STATUS1	15:8	RO	20'h0000_0	Reserved
	7	RO	1'b0	Fault: • **1** – indicates that a fault condition is detected. • **0** – indicates that no fault condition is detected.
	6:3	RO	4'b0000	Reserved.
	2	RO, LL	1'b0	PCS Receive Link Status: • **1** – indicates that the PCS receive link is up. • **0** – indicates that the PCS receive link is or was down (latching low).
	[1:0]	RO	1'b0	Reserved.

Table 38 • DEVICE_ID0 (Offset : 0x00_0008)

Name	Bits	Access	Default	Description
Reserved	15:0	RO	0x0000	Unused.

Table 39 • DEVICE_ID1 (Offset : 0x00_000C)

Name	Bits	Access	Default	Description
Reserved	15:0	RO	`0x0000`	Unused.

Table 40 • SPEED_ABILITY (Offset : 0x00_0010)

Name	Bits	Access	Default	Description
`SPEED_ABILITY`	15:0	RO	0x0200	• **[15:10]** – Reserved. • **[9]** – 400G capable. • **[8]** – 200G capable. • **[7:0]** – always 0

Table 41 • DEVICES_IN_PKG1 (Offset : 0x00_0014)

Name	Bits	Access	Default	Description
`DEVICES_IN_PKG1`	15:0	RO	`16'h0008`	• **[3]** – PCS present • **[All other bits]** – Reserved.

Table 42 • DEVICES_IN_PKG2 (Offset : 0x00_0018)

Name	Bits	Access	Default	Description
`DEVICES_IN_PKG2`	15:0	RO	`16'h0000`	Unused.

Table 43 • CONTROL2 (Offset : 0x00_001C)

Name	Bits	Access	Default	Description
CONTROL2	15:4	RO	12'h000	Reserved.
	3:0	RO	4'b1101	PCS type: • **1 1 0 1** – 400GBASE-R PCS type. • **1 1 0 0** – 200GBASE-R PCS type.

Table 44 • STATUS2 (Offset : 0x00_0020)

Name	Bits	Access	Default	Description
STATUS2	15:14	RO	0x8000	Device present: **10** – device responding at this address.
	13:12	RO		Reserved.
	11	RO, LH		Transmit fault: **1** – fault condition (Latch high)
	10	RO		Receive fault: **1** – fault condition (Latch high)
	9:0	RO		Reserved. (capabilities <200G)

Table 45 • STATUS3 (Offset : 0x00_0024)

Name	Bits	Access	Default	Description
STATUS3	15:2	RO	0x0002	Reserved.
	1	RO		400GBase-R capable
	0	RO		200GBase-R capable

Table 46 • BASER_STATUS1 (Offset : 0x00_0080)

Name	Bits	Access	Default	Description
BASER_STATUS1	15:13	RO	0	Reserved.
	12	RO	0	Receive link status • **1** – link up. • **0** – link down.
	11:0	RO	0	Reserved.

Table 47 • BASER_STATUS2 (Offset : 0x00_0084)

Name	Bits	Access	Default	Description
BASER_STATUS2	15	RO	0	Latched block lock (Not available for 400G)
	14	RO	0	Latched high BER (Not available for 400G)
	13:8	RO	0	Reserved.
	7:0	RO	0	Errored blocks counter (non roll-over, saturating).

Table 48 • MULTILANE_ALIGN_STAT1 (Offset : 0x00_00C8)

Name	Bit	Access	Default	Description
Reserved	15:13	RO	0	Reserved.
Lane alignment status	12	RO	0	**1** – all Receive lanes locked and aligned.
Reserved	11:0	RO	0	Reserved. (Block lock not available for 400G)

Table 49 • MULTILANE_ALIGN_STAT2 (Offset : 0x00_00CC)

Name	Bits	Access	Default	Description
Reserved	15:0	RO	0	Reserved. (Block lock not available for 400G)

Table 50 • MULTILANE_ALIGN_STAT3 (Offset : 0x00_00D0)

Name	Bits [1]	Access	Default	Description
Reserved	15:8	RO	8'b0000_0000	Reserved.
Lane 7 alignment marker lock	7	RO	1'b0	**1** – alignment marker lock achieved.
Lane 6 alignment marker lock	6	RO		
Lane 5 alignment marker lock	5	RO		
Lane 4 alignment marker lock	4	RO		
Lane 3 alignment marker lock	3	RO		
Lane 2 alignment marker lock	2	RO		
Lane 1 alignment marker lock	1	RO		
Lane 0 alignment marker lock	0	RO		

Table 51 • MULTILANE_ALIGN_STAT4 (Offset – 0x00_00D4)

Name [1]	Bits	Access	Default	Description
Reserved	15:8	RO	4'b0000	Reserved.

Name [1]	Bits	Access	Default	Description
Lane 15 alignment marker lock	7	RO	1'b0	**1** – alignment marker lock achieved.
Lane 14 alignment marker lock	6	RO		
Lane 13 alignment marker lock	5	RO		
Lane 12 alignment marker lock	4	RO		
Lane 11 alignment marker lock	3	RO		
Lane 10 alignment marker lock	2	RO		
Lane 9 alignment marker lock	1	RO		
Lane 8 alignment marker lock	0	RO		

Quad MAC Register Map

The two Quad MACs have identical register maps as detailed below.

Base Address

The Ethernet subsystem Quad MACs are located at:

- Ethernet Subsystem 0, Quad MAC 0 – `0x081_b600_0000` : "`CSR_SPACE ETHERNET_0 QUAD_MAC_0`"
- Ethernet Subsystem 0, Quad MAC 1 – `0x081_b700_0000` : "`CSR_SPACE ETHERNET_0 QUAD_MAC_1`"
- Ethernet Subsystem 1, Quad MAC 0 – `0x081_c600_0000` : "`CSR_SPACE ETHERNET_1 QUAD_MAC_0`"
- Ethernet Subsystem 1, Quad MAC 1 – `0x081_c700_0000` : "`CSR_SPACE ETHERNET_1 QUAD_MAC_1`"

Register Names

In the tables below, as each Quad MAC has four independent MACs, each capable of 100G throughput, there are four instances of each register. Each register has a suffix of "`_<MAC number>`" appended to the register name. For example:

- For the Revision register, there are `REVISION_0`, `REVISION_1`, `REVISION_2` and `REVISION_3`
- For the scratch register, there are `SCRATCH_0`, `SCRATCH_1`, `SCRATCH_2` and `SCRATCH_3`

A sample listing of these registers is shown below.

```
cmd> ac7t1500::get_dict_spaces CSR_SPACE ETHERNET_0 QUAD_MAC_0 REVISION*
The AC7t1500ES0 register names for CSR_SPACE ETHERNET_0 QUAD_MAC_0 REVISION* are REVISION_0 REVISION_1 REVISION_2 REVISION_3
REVISION_0 REVISION_1 REVISION_2 REVISION_3
cmd> ac7t1500::get_dict_spaces CSR_SPACE ETHERNET_0 QUAD_MAC_0 SCRATCH*
The AC7t1500ES0 register names for CSR_SPACE ETHERNET_0 QUAD_MAC_0 SCRATCH* are SCRATCH_0 SCRATCH_1 SCRATCH_2 SCRATCH_3
SCRATCH_0 SCRATCH_1 SCRATCH_2 SCRATCH_3
cmd>
```

Figure 11 • Quad MAC Register Names

Register Definitions

Table 52 • REVISION (Offset : 0x00_0000)

Name	Bits	Access	Default	Description
Revision	[31:0]	RO	0xXX	Indicates Quad MAC Revision.

Table 53 • SCRATCH (Offset : 0x00_0004)

Name	Bits	Access	Default	Description
Scratch	[31:0]	RW	0x0000_0000	Scratch Register to test read and write access.

Table 54 • PMAC COMMAND_CONFIG (Offset : 0x00_0008)

Name	Bits	Access	Default	Description
NO_PREAM (1)	31	RW	0	• **1** – Enables no-preamble mode. The MAC removes the preamble and SFD of frames being transmitted and the first packet byte follows the start character. On receive, the MAC expects the first packet byte to follow the start character. • **0** – Disables no-preamble mode.
Reserved	30	RO	0	Unused.

Name	Bits	Access	Default	Description
PMAC_RX_PAUSE_DIS	29	RW	0	• **1** – Disables the PMAC RX PAUSE function (both FC and PFC). The RX PMAC does not cause the TX to be paused (in the case of FC) and does not assert any of the <prefix>_quad<n>_pmac_pause_on[n:0] priorities (in the case of PFC) when PAUSE frames are received. Setting PMAC_RX_PAUSE_DIS does not change the functionality of PMAC_PAUSE_FWD and PMAC_PAUSE_IGNORE. The difference with PMAC_PAUSE_IGNORE is that the underlying pause timers are still operational when PMAC_PAUSE_IGNORE is set whereas setting PMAC_RX_PAUSE_DIS is immediate and causes the relevant <prefix>_quad<n>_pmac_pause_on[n:0] signals to de-assert and reset the internal timers. • **0** – Enables the PMAC RX PAUSE function.
PMAC_TX_PAUSE_DIS	28	RW	0	• **1** – Disables the PMAC TX PAUSE function (both FC and PFC). The TX PMAC does not react to <prefix>_quad<n>_pmac_xoff_gen[n:0] in order to generate PAUSE frames. If a priority was paused (i.e. the pause timer had not expired) a XON frame is transmitted when this bit is set. • **0** – Enables the PMAC TX PAUSE function.
FLT_HDL_DIS	27	RW	0	• **1** – Disables RS fault handling. Incoming fault sequences are ignored and have no effect on the transmitter. • **0** – Default, the MAC RS automatically inserts remote faults and idles in egress direction on detection of local faults and remote faults, respectively, on ingress direction. In addition, the MAC transmitter is stopped and does not read frames from the transmit FIFO as long as the fault condition persists.
PMAC_TX_FIFO_RESET (2)	26	RW, SC	0	Self-Clearing TX FIFO reset command. When set, the TX FIFO is reset, discarding any frames that might have been stored in the FIFO. The bit is cleared when the reset sequence has completed. To avoid outgoing frame corruption, the FIFO reset should be performed after the TX has been disabled (TX_ENA = 0) and no transmission is ongoing.
Reserved	[25:23]	RO	3'b000	Unused.
PMAC_TX_FLUSH (3)	22	RW	0	Egress Flush Enable: • **1** – The Core reads out the TX FIFO and drops the data (data is not sent out on the line). The associated pause signals (link pause or priority flow control) are masked. In addition, statistics and timestamp return indications might not occur for the flushed frames. • **0** – Normal MAC operation.
RX_SFD_ANY (4)	21	RW	0	Disable check of SFD (0xd5) character at frame start: • **1** – Frame is accepted even if the SFD byte following the preamble is not 0xd5. • **0** – The frame is accepted only if the SFD byte is found with value 0xd5.

Name	Bits	Access	Default	Description
PMAC_PAUSE_PFC_COMP	20	RW	0	Link Pause Compatibility with PFC Mode: • **1** – Pause frames in legacy pause mode are processed similar to PFC frames in PFC mode, i.e., the transmit path is not paused by incoming pause frames, only the pause status is asserted as long as the internal pause timer has not expired. This bit is relevant only when PFC_MODE = 0. • **0** – Pause frames are processed.
PMAC_PFC_MODE	19	RW	0	Enable Priority Flow Control (PFC) mode of operation: • **1** – The MAC transmits and accepts PFC frames. • **0** – The MAC uses standard Link Pause frames.
PMAC_PAUSE_MASK_E	18	RW		• **1** – A PMAC PAUSE frame does not cause the EMAC to pause transmission. • **0** – A PMAC PAUSE frame causes the EMAC to pause transmission. Has no effect when PFC mode is used.
PMAC_PAUSE_MASK_P	17	RW		• **1** – A PMAC PAUSE frame does not cause the PMAC to pause transmission. • **0** – A PMAC PAUSE frame causes the PMAC to pause transmission. Has no effect when PFC mode is used.
FORCE_SEND_IDLE (5)	16	RW	0	• **1** – Suppresses any frame transmissions and forces IDLE on the transmit interface instead of frames. This control affects the MAC reconciliation layer (RS) which acts after the MAC datapath has processed the frame. • **0** – Normal Operation.
Reserved	[15:14]	RW	2'b00	Unused.
PMAC_CNTL_FRM_ENA	13	RW	0	MAC Control Frame Enable: • **1** – MAC Control frames (type 0x8808) with any opcode other than 0x0001 when PFC_MODE = 0 or other than 0x0101 when PFC_MODE = 1 are accepted and forwarded to the Client interface. • **0** – MAC Control frames with any opcode other than 0x0001 (when PFC_MODE = 0) or other than 0x0101 (when PFC_MODE = 1) are silently discarded.
PMAC_SW_RESET (6)	12	RW, SC	0	PMAC Software Reset. Self clearing bit: • **1** – Resets all statistic counters as well as datapath functions. This should be issued after all traffic has been stopped as a result of clearing RX_ENA and TX_ENA. • **0** – Normal operation.

Name	Bits	Access	Default	Description
TX_PAD_EN	11	RW	1	Enable padding of frames in transmit direction: • **1** – Normal operation. All frames from the EIU which are less than 64-bytes must also have crc_insert set in the TX flags, otherwise the frame is sent as is which potentially could result in an invalid frame length. • **0** – The MAC does not extend frames received from the EIU to a minimum of 64 bytes, allowing for transmission of too short frames (violating the Ethernet minimum size requirement).
Reserved	10	RW	0	Unused.
PMAC_TX_ADDR_INS	9	RW	0	• **1** – Replace the frame source MAC address with the address programmed in MAC_ADDR_0 and MAC_ADDR_1. • **0** – Frame is unmodified.
PMAC_PAUSE_IGNORE	8	RW	0	Ignore Pause Frame Quanta: • **1** – Received pause frames are ignored by the MAC. • **0** – The transmit process is stopped for the amount of time specified in the pause quanta received within a pause frame.
PMAC_PAUSE_FWD	7	RW	0	Terminate/Forward Pause Frames: • **1** – Pause frames are forwarded to the user application. • **0** – Normal mode. Pause frames are terminated and discarded within the MAC.
PMAC_CRC_FWD	6	RW	0	Terminate/Forward Received CRC: • **1** – Enabled, the CRC field of received frames is forwarded with the frame to the user application. • **0** – Disabled (set to reset value 0), the CRC field is stripped from the frame.
Reserved	5	RO	0	Unused.
PMAC_PROMIS_EN	4	RW	0	• 1 – Enables MAC promiscuous operation. All frames are received without any MAC address filtering. • 0 – Disables MAC promiscuous operation.
Reserved	3:2	RO	00	Unused.
PMAC_RX_ENA	1	RW	0	Preemptible MAC Receive Path Enable: • **1** – Enable the MAC receive path. • **0** – Disable the MAC receive path.
PMAC_TX_ENA	0	RW	0	Preemptible MAC Transmit Path Enable: • 1 – Enable the MAC transmit path. • 0 – Disable the MAC transmit path.

Name	Bits	Access	Default	Description

Table Notes

1. Not supported when traffic interspersing is enabled. Traffic interspersing occurs when both the preemptive MAC (PMAC) and express MAC (EMAC) are used simultaneously.
2. When the credit-based FIFO is used, the credit return logic is also reset and the application must reinitialize its credit counter to maximum.
3. Egress flush operates independent of the RX link state.
4. For 802.3br operation, this bit must be set to 0. Can be set to 1 for normal MAC operation when 802.3br traffic is not expected.
5. This bit does not have an effect on fault handling (i.e., reception of local fault still causes transmit of remote fault).
6. Performing the SW reset can lead to an RX exception to the EIU if the reset is issued in the middle of a receive frame transfer. The end of packet is lost and the EIU might need to be reset in order to continue to receive frames.

Table 55 • PMAC_MAC_ADDR_0 (Offset : 0x00_000C)

Name	Bits	Access	Default	Description
MAC_ADDR_0	[31:0]	RW	0x0000_0000	The lower 32-bits of the 48-Bit MAC Address. Bit 0 is LSB.

Table 56 • PMAC_MAC_ADDR_1 (Offset : 0x00_0010)

Name	Bits	Access	Default	Description
MAC_ADDR_1	[31:16]	RW	0x0000	Unused and always set to 0.
	[15:0]	RW	0x0000	The upper 16-bits of the 48-Bit MAC Address. Bit 0 is Bit 32 of MAC address.

Table 57 • PMAC_FRM_LENGTH (Offset : 0x00_0014)

Name	Bits	Access	Default	Description
TX_FRM_LENGTH (1)	[31:16]	RW	0x0000	Defines the maximum length of a transmitted frame. When set to 0x0000, the value of RX_FRM_LENGTH is used, so transmit and receive frames have the same maximum length.
RX_FRM_LENGTH	[15:0]	RW	0x0600	Maximum supported RX frame length. When a frame exceeds the given length specified, the MAC RX truncates the frame as described in Frame Truncation (page 74). The MAC supports any frame size up to 32736 bytes (0x7FE0). Set to 1536 (0x0600) after reset.

Table Notes

1. This value is used for statistics purposes only and has no effect on the MAC transmit operation.

Frame Truncation

In the receive direction, the MAC is always checking the received frame length (total octets following the preamble and SFD) against the configured FRM_LENGTH value. If the frame exceeds the programmed value, the frame is truncated and provided to the NAP with the length error flag set. See Receive Flags (page 17). When frame truncation happens, the receive FIFO write might stop prior to the actual truncation point. Hence, the frame delivered to the EIU can be shorter than the given FRM_LENGTH but the truncation is always executed only when the exact limit is reached. In addition, the last octets of a truncated frame can contain arbitrary data.

Another cause of frame truncation can be a receive FIFO almost full condition during reception which is reported accordingly in the frame receive status.

Table 58 • PMAC_RX_FIFO_SECTIONS (Offset : 0x00_001C)

Name	Bits	Access	Default	Description
RX_FIFO_SECTIONS(1)	[31:0]	RW	0x0000_0000	• **[15:0]** – RX section available threshold. When the receive FIFO level reaches or exceeds this value, the FIFO signals to the EIU that data is available, either a complete frame or at least the threshold amount. When asserted, the data is transferred to the EIU until end of frame is reached. To enable store and forward on the receive path, set this value to 0. In this case, data is only transferred when a complete frame is available in the receive FIFO. • **[31:16]** – RX section empty threshold. When the receive FIFO level reaches or exceeds this value, an indication is provided to the MAC transmitter which generates an XOFF Pause frame to indicate congestion to the remote Ethernet transmitter. When the receive FIFO level drops below this value, the MAC transmits an XON Pause frame to indicate that congestion has been cleared and that the remote Ethernet transmitter may continue. A value of 0 disables the function, and no Pause frames are transmitted. All threshold values are in steps of FIFO words.

Table Notes

1. For the Quad MACs, each RX FIFO is 256 bits (32-bytes) wide and 32 entries deep. Therefore, only bits [4:0] of each threshold value is applicable.

Table 59 • PMAC_TX_FIFO_SECTIONS (Offset : 0x00_0020)

Name	Bits	Access	Default	Description
TX_FIFO_SECTIONS[1]	[31:0]	RW	0x0000_0007	• **[15:0]** – TX section available threshold, default 0x7. When the transmit FIFO level reaches this value, the MAC commences transmission of the data in the FIFO, continuing until the end of frame is reached. Independent of this value, if a complete frame is stored in the FIFO, transmission always starts. To operate in a cut through mode, the value must be set greater than or equal to 4. A value of 0 configures store-and-forward operation with the MAC only transmitting when a complete frame is stored in the FIFO. In this scenario, the user design must ensure that all frames transmitted are smaller than the FIFO size. If too large a frame is sent, then a deadlock occurs which is only cleared by setting this threshold to a non-zero value. • **[31:16]** – TX section empty threshold, default 0x0. When the transmit FIFO level reaches this value, the EIU is signaled that the FIFO is almost full. This causes the EIU to de-assert ready to the respective NAP column. This could affect other NAPs transmitting on the column, possibly leading to data corruption due to data starvation on the other channel transmit FIFOs. A value of 0 disables the signalling of almost empty. All threshold values are in steps of FIFO words.

Table Notes

1. For the Quad MACs, each TX FIFO is 256-bit (32-bytes) wide and 32 entries deep. Therefore, only bits [4:0] of each threshold value is applicable.

Table 60 • MAC_STATUS (Offset : 0x00_0040)

Name	Bits	Access	Default	Description
Reserved	31:24	RO	0	Unused.
AVB_FIFO_OVR	23	RO	0	Latched-High indication that an AVB FIFO overrun has occurred. If this event occurs, it is necessary to increase the scale of the AVB time_Xbyte strobe in the AVB_CONTROL register, or to increase the frequency of the ff_tx_clk clock (see Ethernet Clocks (page 31)). Reset to 0 after register read.
EMAC_RX_EMPTY	22	RO	0	EMAC RX FIFO empty indicator: • **1** – EMAC RX FIFO is empty. • **0** – EMAC RX FIFO has data.
EMAC_TX_EMPTY	21	RO	0	EMAC TX FIFO empty indicator: • **1** – EMAC TX FIFO is empty. • **0** – EMAC TX FIFO has data.
Reserved	20:10	RO	0	Unused.

Name	Bits	Access	Default	Description
TX_MERGE_ISIDLE	9	RO	0	TX MERGE layer idle indicator: • **1** – TX MERGE layer is in idle. • **0** – TX MERGE layer is not in idle.
TX_ISIDLE	8	RO	0	TX MAC idle indicator: • **1** – TX MAC is in idle. • **0** – TX MAC is not in idle. Toggles during normal operation when the MAC is transmitting (might assert during IPG times).
RX_LINT_FAULT (1)	7	ROR	0	Latch-High Link Interruption (fault) status: **1** – RS detected Link Interruption Sequences on the XGMII receive interface. **0** – reset value after read or reset.
PMAC_RX_EMPTY	6	RO	0	PMAC RX FIFO empty indicator: • **1** – PMAC RX FIFO is empty. • **0** – PMAC RX FIFO has data.
PMAC_TX_EMPTY	5	RO	0	PMAC TX FIFO empty indicator: • **1** – PMAC TX FIFO is empty. • **0** – PMAC TX FIFO has data.
RX_LOWP	4	RO	0	Receive Low Power Idle indicator: • **1** – Low Power Idle is currently detected on the RX XLGMII interface. • **0** – the MAC is operating in normal mode.
TS_AVAIL	3	RW	0	Transmit Timestamp Available. Indicates that the timestamp of the last transmitted event frame (which had the transmit flag Frame set, (see Transmit Flags (page 9)) is available in the register TS_TIMESTAMP. To clear TS_AVAIL, the bit must be written with a 1.
PHY_LOS	2	RO	0	PHY loss of signal indicator: • **1** – PHY indicates loss-of-signal. • **0** – PHY is operating normally.
RX_REM_FAULT	1	ROR	0	Latch-High Remote Fault status: • **1** – the RS detected Remote Fault Sequences on the XLGMII receive interface. • **0** – reset value after read or reset.
RX_LOC_FAULT	0	ROR	0	Latch-High Local Fault status: • **1** – the RS detected Local Fault Sequences on the XLGMII receive interface. • **0** – Reset value after read or reset.

Name	Bits	Access	Default	Description

Table Notes

1. Only applicable in 10Gbps/25Gpbs mode of operation.

Table 61 • TX_IPG_LENGTH (Offset : 0x00_0044)

Name	Bits	Access	Default	Description
TX_IPG_LENGTH (1)	[31:0]	RW	0x0000_000C	Transmit Inter-Packet-Gap (IPG) options: • [5:0] – Configure the average IPG length in increments of 8, starting from 12 up to 68. Due to the Deficit Idle Count (DIC) the per-packet IPG can be -4/+7 bytes of the programmed value in CGMII mode and -3/+3 bytes in XGMII mode. Special modes: • [0] – When set 1, disables DIC and enables shortened IPG mode. Should be 0 for normal operation. • [1] – Not used, write 0 always. • [15:6] – reserved, read only bits. • [31:16] – Set IPG compensation count, increasing bandwidth. Supported values are: **0x0000** – IPG bias mode disabled, no idle blocks removed. Standard compliant mode of 10G/25Gbe when not using PCS with RS-FEC. **0x3FFF** – One 8-byte block of idle is removed after every 16383 blocks. This is the standard compliant mode for 40G/100Gbe to account for 40G/100Gbe alignment marker compensation. **0x4FFF** – One 8-byte block of idle is removed after every 20479 blocks. This is the standard compliant mode for 25G/50Gbe when using PCS with RS-FEC to account for 25G/50Gbe alignment marker compensation.

Table Notes

1. If IPG is reduced below the default of 12, then the minimal required clock frequencies must be increased accordingly to support the additional bandwidth.

Table 62 • CRC_MODE (Offset : 0x00_0048)

Name	Bits	Access	Default	Description
CRC_MODE	[31:0]	RW	0x0000_0000	Configures the CRC operating mode. • [15:0] – Reserved. Read-only. Always 0. • [16] – Disable RX CRC checking. • [31:17] – Reserved. Read-only. Always 0.

Table 63 • CRC_INV_MASK (Offset : 0x00_004C)

Name	Bits	Access	Default	Description
CRC_INV_MASK	[31:0]	RW	0xFFFF_FFFF	Defines the invert mask to use for the CRC-inverse function. See Transmit FCS Flags (page 9). Enabling the inverted CRC is achieved by setting the crc_invert flag from TX NAP frame.

Table 64 • TS_TIMESTAMP (Offset : 0x00_007C)

Name	Bits	Access	Default	Description
TS_TIMESTAMP	[31:0]	RO	0x0000_0000	Timestamp of the last frame transmitted by the MAC that had the frame = 1'b1 signal asserted in TX flags. Valid when the status bit STATUS(TS_AVAIL) is set to 1'b1.

Table 65 • XIF_MODE (Offset : 0x00_0080)

Name	Bits	Access	Default	Description
Reserved	[31:13]	RO	19'b0	Unused.
TS_UPD64_MODE	12	RW	1'b0	Only applies when bit[15] is set in the one-step control vector (64-bit seconds/nanoseconds time field update). When bit[15] is not set, this control has no effect. • **1** – Enable update 64 mode in 1-step operations. When bit[15] is set, timestamp the frame using the received and current timestamp as {TX timestamp[31:0], free_running_counter[31:0]}. • **0** – When bit[15] is set, timestamp the frame with free_running_counter[63:0] See IEEE Timestamping (page 96) for further details.
Reserved	[11:4]	RO	8'h00	Unused.
ONE_STEP_ENA	5	RW	1'b0	Enable 1-step IEEE-1588 functions. • **1** – The user application can instruct the MAC to perform on-the-fly correction field updates with the one-step vector signals. See IEEE Timestamping (page 96). • **0** – When disabled (0), the one-step functions are all disabled and a lower latency data path is enabled. Any user application 1-step requests are ignored. Changing of the one-step enable is allowed only when the TX FIFO is empty.
Reserved	[4:0]	RO	5'b0000	Unused.

Table 66 • EMAC_COMMAND_CONFIG (Offset : 0x00_0108)

Name	Bits	Access	Default	Description
Reserved	[31:30]	RO	2'b00	Unused.
EMAC_RX_PAUSE_DIS	29	RW	0	Disables the EMAC RX PAUSE function (both FC and PFC). • **1** – The RX EMAC does not cause the TX to be paused (in the case of FC) and does not assert any of the <prefix>_quad<n>_emac_pause_on[n:0] priorities (in the case of PFC) when PAUSE frames are received. Setting EMAC_RX_PAUSE_DIS does not change the functionality of EMAC_PAUSE_FWD and EMAC_PAUSE_IGNORE. The difference with EMAC_PAUSE_IGNORE is that the underlying pause timers are still operational when EMAC_PAUSE_IGNORE is set whereas setting EMAC_RX_PAUSE_DIS is immediate and causes the relevant <prefix>_quad<n>_emac_pause_on[n:0] signals to de-assert and resets the internal timers. • **0** – Enables the EMAC RX PAUSE function.
EMAC_TX_PAUSE_DIS	28	RW	0	Disables the EMAC TX PAUSE function (both FC and PFC). • **1** – The TX EMAC does not react to <prefix>_quad<n>_emac_xoff_gen[n:0] in order to generate PAUSE frames. If a priority was paused (i.e., the pause timer had not expired) a XON frame is transmitted when this bit is set. • **0** – Enables the EMAC TX PAUSE function.
Reserved	27	RO	0	Unused.
EMAC_TX_FIFO_RESET (1)	26	RW, SC	0	Self-Clearing TX FIFO reset command. • **1** – The TX FIFO is reset, discarding any frames that might have been stored in the FIFO. The bit is cleared when the reset sequence has completed. To avoid outgoing frame corruption, the FIFO reset should be performed after the TX has been disabled (TX_ENA = 0) and no transmission is ongoing. • **0** – Normal TX FIFO operation.
Reserved	[25:23]	RO	3'b000	Unused.
EMAC_TX_FLUSH (2)	22	RW	0	Egress Flush Enable: • **1** – The Core reads out the TX FIFO and drops the data (data is not sent out on the line). The associated pause signals (link pause or priority flow control) are masked. In addition, statistics and timestamp return indications might not occur for the flushed frames. • **0** – Normal MAC operation.
Reserved	21	RW	0	Unused.

Name	Bits	Access	Default	Description
EMAC_PAUSE_PFC_COMP	20	RW	0	Link Pause Compatibility with PFC Mode: • **1** – Pause frames in legacy pause mode are processed similar to PFC frames in PFC mode, i.e., the transmit path is not paused by incoming pause frames. Only the pause status is asserted as long as the internal pause timer has not expired. • **0** – Pause frames are processed. This bit is relevant only when PFC_MODE = 0.
EMAC_PFC_MODE	19	RW	0	Enables Priority Flow Control (PFC) mode of operation: • **1** – The MAC transmits and accepts PFC frames. • **0** – The MAC uses standard Link Pause frames.
EMAC_PAUSE_MASK_E	18	RW		• **1** – An EMAC PAUSE frame does not cause the EMAC to pause transmission. • **0** – An EMAC PAUSE frame causes the EMAC to pause transmission. Has no effect when PFC mode is used.
EMAC_PAUSE_MASK_P	17	RW		• **1** – An EMAC PAUSE frame does not cause the PMAC to pause transmission. • **0** – An EMAC PAUSE frame causes the PMAC to pause transmission. Has no effect when PFC mode is used.
Reserved	[16:14]	RW	3'b000	Unused.
EMAC_CNTL_FRM_ENA	13	RW	0	EMAC Control Frame Enable: • **1** – MAC Control frames (type 0x8808) with any opcode other than 0x0001 when PFC_MODE = 0 or other than 0x0101 when PFC_MODE = 1 are accepted and forwarded to the Client interface. • **0** – MAC Control frames with any opcode other than 0x0001 (when PFC_MODE = 0) or other than 0x0101 (when PFC_MODE = 1) are silently discarded.
EMAC_SW_RESET (3)	12	RW, SC	0	EMAC Software Reset. Self clearing bit: • **1** – Resets all statistic counters as well as datapath functions. • **0** – Statistic counters and datapath functions are not affected. Self set. This should be issued after all traffic has been stopped as a result of clearing EMAC_RX_ENA and EMAC_TX_ENA.
Reserved	[11:10]	RW	2'b00	Unused.
EMAC_TX_ADDR_INS	9	RW	0	EMAC TX MAC address insertion: • **1** – Replace the frame source MAC address with the address programmed in MAC_ADDR_0 and MAC_ADDR_1. • **0** – Frame is unmodified.

Name	Bits	Access	Default	Description
EMAC_PAUSE_IGNORE	8	RW	0	Ignore Pause Frame Quanta: • **1** – Received pause frames are ignored by the MAC. • **0** – The transmit process is stopped for the amount of time specified in the pause quanta received within a pause frame.
EMAC_PAUSE_FWD	7	RW	0	Terminate/Forward Pause Frames: • **1** – Pause frames are forwarded to the user application. • **0** – Normal mode. Pause frames are terminated and discarded within the MAC.
Reserved	[6:5]	RO	2'b00	Unused.
EMAC_PROMIS_EN	4	RW	0	Enables MAC promiscuous operation: • **1** – All frames are received without any MAC address filtering. • **0** – All frames are filtered normally by MAC address.
Reserved	[3:2]	RO	2'b00	Unused.
EMAC_RX_ENA	1	RW	0	Express MAC Receive Path Enable: • **1** – Enable the MAC receive path. • **0** – Disable the MAC receive path.
EMAC_TX_ENA	0	RW	0	Express MAC Transmit Path Enable: • **1** – Enable the MAC transmit path. • **0** – Disable the MAC transmit path.

Table Notes

1. When the credit-based FIFO is used, the credit return logic is also reset and the application must reinitialize its credit counter to the maximum.
2. Operates independent from the RX link state.
3. Performing the SW reset can lead to an RX exception to the EIU if the reset is issued in the middle of a receive frame transfer. The end of packet is lost and the EIU might need to be reset in order to continue to receive frames.

Quad PCS Register Map

The two Quad PCSs have identical register maps.

Base Address

The Ethernet subsystem Quad PCSs are located at:

- Ethernet Subsystem 0, Quad PCS 0 – `0x081_b200_0000` : "CSR_SPACE ETHERNET_0 QUAD_PCS_0"
- Ethernet Subsystem 0, Quad PCS 1 – `0x081_b300_0000` : "CSR_SPACE ETHERNET_0 QUAD_PCS_1"
- Ethernet Subsystem 1, Quad PCS 0 – `0x081_c200_0000` : "CSR_SPACE ETHERNET_1 QUAD_PCS_0"
- Ethernet Subsystem 1, Quad PCS 1 – `0x081_c300_0000` : "CSR_SPACE ETHERNET_1 QUAD_PCS_1"

Register Names

In the tables below, each Quad PCS has the following:

- Four independent PCS blocks each capable of 50G throughput, numbered 0 to 3
- Two additional 100G PCS blocks, numbered 01 and 23
- One dedicated 200G PCS block, numbered 200

In normal operation, PCS [3:0] are used, however, if four channels of 100G are required, then PCS 0 and 2 are used alongside PCS 01 and 23. If multiple lanes of 200G are required, one lane is used from the 200G/400G PCS, and the other 200G from this Quad MAC PCS.

Due to this arrangement, there are seven instances of each register. Each register has a suffix of "_<PCS number>" appended to the register name. For example:

- For the CONTROL1 register, there are CONTROL1_0, CONTROL1_1, CONTROL1_2, CONTROL1_3, CONTROL1_01, CONTROL1_23, and CONTROL1_200
- For the STATUS1 register, there are STATUS1_0, STATUS1_1, STATUS1_2, STATUS1_3, STATUS1_01, STATUS1_23 and STATUS1_200

An example of listing these registers is shown below

```
cmd> ac7t1500::get_dict_spaces CSR_SPACE ETHERNET_0 QUAD_PCS_0 CONTROL1*
The AC7t1500ES0 register names for CSR_SPACE ETHERNET_0 QUAD_PCS_0 CONTROL1* are CONTROL1_0 CONTROL1_1 CONTROL1_2 CONTROL1_3 CONTROL1_200 CONTROL1_01 CONTROL1_23
CONTROL1_0 CONTROL1_1 CONTROL1_2 CONTROL1_3 CONTROL1_200 CONTROL1_01 CONTROL1_23
cmd> ac7t1500::get_dict_spaces CSR_SPACE ETHERNET_0 QUAD_PCS_0 STATUS1*
The AC7t1500ES0 register names for CSR_SPACE ETHERNET_0 QUAD_PCS_0 STATUS1* are STATUS1_0 STATUS1_1 STATUS1_2 STATUS1_3 STATUS1_200 STATUS1_01 STATUS1_23
STATUS1_0 STATUS1_1 STATUS1_2 STATUS1_3 STATUS1_200 STATUS1_01 STATUS1_23
cmd>
```

Figure 12 • Quad PCS Register Names

Register Definitions

Table 67 • CONTROL1 (Offset : 0x00_0000)

Name	Bits	Access	Default	Description
CONTROL1	31:16	RO	0x0000	Unused.
	15	RW, SC	0	PCS Reset: • **1 –** assert. • **0** – de-assert.
	14	RW	0	Loopback: • **1** – enable. • **0** – disable.

Name	Bits	Access	Default	Description
	13:6	RO	8'b1000_0001	Reserved.
	5:2	RO	4'b0000	Speed selection: • **0 1 1 0** – 50 Gbps. • **0 1 0 1** – 25 Gbps. • **0 1 0 0** – 100 Gbps. • **0 0 1 1** – 40 Gbps. • **0 0 0 0** – 10 Gbps.
	1:0	RO	2'b00	Reserved.

Table 68 • STATUS1 (Offset : 0x00_0004)

Name	Bits	Access	Default	Description
STATUS1	31:12	RO	20'h0000_0	Reserved
	11	RO, LH	1'b0	TX_LPI: • **1** – transmit is or was in LPI (EEE). • **0** – normal operation.
	10	RO, LH	1'b0	RX_LPI: • **1** – receive is or was in LPI state (EEE). • **0** – normal operation.
	9	RO	1'b0	TX_LPI_ACTIVE: • **1** – transmit is currently in LPI state (EEE). • **0** – normal operation.

Name	Bits	Access	Default	Description
	8	RO	1'b0	RX_LPI_ACTIVE: • **1** – receive is currently in LPI state (EEE). • **0** – normal operation.
	7	RO	1'b0	Fault: • **1** – indicates that a fault condition is detected. • **0** – indicates that no fault condition is detected.
	6:3	RO	4'b0000	Reserved.
	2	RO, LL	1'b0	PCS Receive Link Status: • **1** – indicates that the PCS receive link is up. • **0** – indicates that the PCS receive link is or was down (latching low).
	1[(1)]	RO	1'b1	Low-power Ability: • **1** – indicates that the PCS implements a low power mode. • **0** – indicates that the PCS does not implement a low power mode.
	0	RO	1'b0	Reserved.

Table Notes

1. This bit does not refer to Energy Efficient Ethernet (EEE). It only indicates that the control bit 11 in the PCS Control register 1 is supported and allows placing the PCS in a reset state.

Table 69 • DEVICE_ID0 (Offset : 0x00_0008)

Name	Bits	Access	Default	Description
Reserved	31:0	RO	`0x0000_0000`	Unused.

Table 70 • DEVICE_ID1 (Offset : 0x00_000C)

Name	Bits	Access	Default	Description
Reserved	31:0	RO	`0x0000_0000`	Unused.

Table 71 • SPEED_ABILITY (Offset : 0x00_0010)

Name	Bits	Access	Default	Description
SPEED_ABILITY	31:16	RO	`16'h0000`	Unused.
	15:0	RO	**CH0**: `0x03d` **CH1**: `0x0031` or `0x0011`	• **[15:6]** – Reserved. • **[5]** – 50G capable. • **[4]** – 25G capable. • **[3]** – 100G capable. • **[2]** – 40G capable. • **[1]** – always 0. • **[0]** – 10G capable.

Table 72 • DEVICES_IN_PKG1 (Offset : 0x00_0014)

Name	Bits	Access	Default	Description
DEVICES_IN_PKG1	31:16	RO	`16'h0000`	Unused.

Name	Bits	Access	Default	Description
	15:0	RO	16'h0008	• **[15:7]** – Reserved. • **[6]** – TC present. • **[5]** – DTE XS present. • **[4]** – PHY XS present. • **[3]** – PCS present. • **[2]** – WIS present. • **[1]** – PMD/PMA present. • **[0]** – Clause 22 registers present.

Table 73 • DEVICES_IN_PKG2 (Offset : 0x00_0018)

Name	Bits	Access	Default	Description
DEVICES_IN_PKG2	31:16	RO	16'h0000	Unused.
	15:0	RO	16'h0000	• **[15]** – Vendor specific device 2 present. • **[14]** – Vendor specific device 1 present. • **[13]** – Clause 22 extension present. • **[12:0]** – Reserved.

Table 74 • CONTROL2 (Offset : 0x00_001C)

Name	Bits	Access	Default	Description
CONTROL2	31:16	RO	16'h0000	Unused.
	15:4	RO	12'h000	Reserved.

Name	Bits	Access	Default	Description
	3:0	RO	4'b0000	PCS type: • **1 1 0 0** – 200GBASE-R PCS type. • **1 0 1 1** – Reserved. • **1 0 1 0** – Reserved. • **1 0 0 1** – Reserved. • **1 0 0 0** – 50GBASE-R PCS type. • **0 1 1 1** – 25GBASE-R PCS type. • **0 1 1 0** – Reserved. • **0 1 0 1** – 100GBASE-R PCS type. • **0 1 0 0** – 40GBASE-R PCS type. • **0 0 1 1** – Reserved. • **0 0 1 0** – Reserved. • **0 0 0 1** – Reserved. • **0 0 0 0** – 10GBASE-R PCS type.

Table 75 • STATUS2 (Offset : 0x00_0020)

Name	Bits	Access	Default	Description
STATUS2	31:16	RO	16'h0000	Unused.
	15:14	RO	**CH0** – 0x81b1 **CH1** – 0x8181 or 0x8081 **Quad-CH2, 6** – 0x8181 or 0x81a1	Device present: **10** – device responding at this address.
	13:12	RO		Reserved.
	11	RO, LH		Transmit fault: **1** – fault condition on RX path.

Name	Bits	Access	Default	Description
	10	RO		Reserved.
	9	RO		Reserved.
	8	RO		50GBase-R capable.
	7	RO		25GBase-R capable.
	6	RO		Reserved.
	5	RO		100GBase-R capable.
	4	RO		40GBase-R capable.
	3:1	RO		Reserved.
	0	RO		10GBase-R capable.

Table 76 • BASER_STATUS1 (Offset : 0x00_0080)

Name	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
Reserved	15:13	RO	`3'b000`	Reserved.
Receive link status	12	RO	`1'b0`	• **1** – link up. • **0** – link down.
Reserved	11:2	RO	`10'b00_0000_0000`	Reserved.
High BER	1	RO	`1'b0`	**1** – PCS reporting a high BER.
Block lock	0	RO	`1'b0`	**1** – PCS locked to received blocks.

Table 77 • MULTILANE_ALIGN_STAT1 (Offset : 0x00_00C8)

Name	Bits [1]	Access	Default	Description
Unused	31:16	RO	16'h0000	Unused.
Reserved	15:13	RO	3'b000	Reserved.
Lane alignment status	12	RO	1'b0	**1** – all Receive lanes locked and aligned.
Reserved	11:8	RO	4'b0000	Reserved.
Lane 7 block lock	7	RO	1'b0	**1** – block lock achieved.
Lane 6 block lock	6	RO		
Lane 5 block lock	5	RO		
Lane 4 block lock	4	RO		
Lane 3 block lock	3	RO		
Lane 2 block lock	2	RO		
Lane 1 block lock	1	RO		
Lane 0 block lock	0	RO		

Table Notes

1. Bit 0 is relevant when operating in 10/25G. Bits 3:0 are relevant when operating in 40/50G. Bits 7:0 are relevant when operating in 100G.

Table 78 • MULTILANE_ALIGN_STAT2 (Offset : 0x00_00CC)

Name [1]	Bits	Access	Default	Description
Unused	31:16	RO	16'h0000	Unused.
Reserved	15:12	RO	4'b0000	Reserved.
Lane 19 block lock	11	RO	1'b0	**1** – block lock achieved.

Name [1]	Bits	Access	Default	Description
Lane 18 block lock	10	RO		
Lane 17 block lock	9	RO		
Lane 16 block lock	8	RO		
Lane 15 block lock	7	RO		
Lane 14 block lock	6	RO		
Lane 13 block lock	5	RO		
Lane 12 block lock	4	RO		
Lane 11 block lock	3	RO		
Lane 10 block lock	2	RO		
Lane 9 block lock	1	RO		
Lane 8 block lock	0	RO		

Table Notes

1. This register is only relevant when operating in 100G mode.

Table 79 • MULTILANE_ALIGN_STAT3 (Offset : 0x00_00D0)

Name	Bits [1]	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
Reserved	15:8	RO	`8'b0000_0000`	Reserved.
Lane 7 alignment marker lock	7	RO	`1'b0`	**1** – alignment marker lock achieved.
Lane 6 alignment marker lock	6	RO		
Lane 5 alignment marker lock	5	RO		

Name	Bits [1]	Access	Default	Description
Lane 4 alignment marker lock	4	RO		
Lane 3 alignment marker lock	3	RO		
Lane 2 alignment marker lock	2	RO		
Lane 1 alignment marker lock	1	RO		
Lane 0 alignment marker lock	0	RO		

Table Notes

1. Bits 3:0 are relevant when operating in 40/50G modes. Bits 7:0 are relevant when operating in 100G mode with no FEC.

Table 80 • MULTILANE_ALIGN_STAT4 (Offset : 0x00_00D4)

Name [1]	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
Reserved	15:12	RO	`4'b0000`	Reserved.
Lane 19 alignment marker lock	11	RO	`1'b0`	**1** – alignment marker lock achieved.
Lane 18 alignment marker lock	10	RO		
Lane 17 alignment marker lock	9	RO		
Lane 16 alignment marker lock	8	RO		
Lane 15 alignment marker lock	7	RO		

Name [1]	Bits	Access	Default	Description
Lane 14 alignment marker lock	6	RO		
Lane 13 alignment marker lock	5	RO		
Lane 12 alignment marker lock	4	RO		
Lane 11 alignment marker lock	3	RO		
Lane 10 alignment marker lock	2	RO		
Lane 9 alignment marker lock	1	RO		
Lane 8 alignment marker lock	0	RO		

Table Notes

1. This register is only relevant when operating in 100G mode with no FEC.

SerDes Register Map

Each Ethernet subsystem consists of two quad SerDes PHY blocks (4 lanes each). Both quads have identical register maps. For detailed register information, refer to the *Speedster7t SerDes User Guide* (UG099)[9]

It is recommended to perform all SerDes management tasks through remote procedure calls (RPCs). For instructions on using RPCs, refer to the *Speedster7t Device Manager User Guide* (UG115)[10]

Note

On AC7t1400 and AC7t700 devices, direct access to SerDes registers is not supported. All tasks involving those registers must be performed through RPC functions.

Base Address

The Ethernet subsystem Quad SerDes are located at:

- Ethernet Subsystem 0, Quad SerDes 0 – `0x081_b800_0000` : `"CSR_SPACE ETHERNET_0 SERDES_0"`
- Ethernet Subsystem 0, Quad SerDes 1 – `0x081_b900_0000` : `"CSR_SPACE ETHERNET_0 SERDES_1"`

9 https://achronix.com/documentation/speedster7t-serdes-user-guide-ug099
10 https://achronix.com/documentation/speedster7t-device-manager-user-guide-ug115

- Ethernet Subsystem 1, Quad SerDes 0 – `0x081_c800_0000` : "`CSR_SPACE ETHERNET_1 SERDES_0`"
- Ethernet Subsystem 1, Quad SerDes 1 – `0x081_c900_0000` : "`CSR_SPACE ETHERNET_1 SERDES_1`"

Register Names

There are four instances of each register for each of the four lanes in a Quad SerDes. Each register has a suffix of "`_<SerDes lane number>`" appended to the register name. For example, for the `TX_PHASE_FIFO` register, there are `TX_PHASE_FIFO_0`, `TX_PHASE_FIFO_1`, `TX_PHASE_FIFO_2`, `TX_PHASE_FIFO_3`

An example of listing these registers is shown below

```
cmd> ac7t1500::get_dict_spaces CSR_SPACE ETHERNET_0 SERDES_0 TX_PHASE_FIFO_*
The AC7t1500ES0 register names for CSR_SPACE ETHERNET_0 SERDES_0 TX_PHASE_FIFO_* are TX_PHASE_FIFO_0 TX_PHASE_FIFO_1 TX_PHASE_FIFO_2 TX_PHASE_FIFO_3
TX_PHASE_FIFO_0 TX_PHASE_FIFO_1 TX_PHASE_FIFO_2 TX_PHASE_FIFO_3
cmd>
```

Figure 13 • Quad SerDes Register Names

Table 81 • TX_PHASE_FIFO_0 (Offset : 0x00_50EC)

Name (1)	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
elecidle_nt	4:1	RW	`4'b0000`	Override control of the transmit electrical idle function • **11** – transmitter placed into electrical idle • **10** – transmitter in normal operating mode • **01** – transmitter in normal operating mode • **00** – transmitter in normal operating mode This bit is kept timing aligned with the txdata through tx fifo and into afe
Lane 0 bypass_ena_a	0	RW	`1'b0`	**1**: Enable bypass of the electrical idle function

Table 82 • TX_PHASE_FIFO_1 (Offset : 0x00_90EC)

Name (1)	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.

Name [1]	Bits	Access	Default	Description
elecidle_nt	4:1	RW	`4'b0000`	Override control of the transmit electrical idle function • **11** – transmitter placed into electrical idle • **10** – transmitter in normal operating mode • **01** – transmitter in normal operating mode • **00** – transmitter in normal operating mode This bit is kept timing aligned with the txdata through tx fifo and into afe
Lane 0 bypass_ena_a	0	RW	`1'b0`	**1**: Enable bypass of the electrical idle function

Table 83 • TX_PHASE_FIFO_2 (Offset : 0x00_D0EC)

Name [1]	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
elecidle_nt	4:1	RW	`4'b0000`	Override control of the transmit electrical idle function • **11** – transmitter placed into electrical idle • **10** – transmitter in normal operating mode • **01** – transmitter in normal operating mode • **00** – transmitter in normal operating mode This bit is kept timing aligned with the txdata through tx fifo and into afe
Lane 0 bypass_ena_a	0	RW	`1'b0`	**1**: Enable bypass of the electrical idle function

Table 84 • TX_PHASE_FIFO_3 (Offset : 0x01_10EC)

Name [1]	Bits	Access	Default	Description
Unused	31:16	RO	`16'h0000`	Unused.
elecidle_nt	4:1	RW	`4'b0000`	Override control of the transmit electrical idle function • **11** – transmitter placed into electrical idle • **10** – transmitter in normal operating mode • **01** – transmitter in normal operating mode • **00** – transmitter in normal operating mode This bit is kept timing aligned with the txdata through tx fifo and into afe
Lane 0 bypass_ena_a	0	RW	`1'b0`	**1**: Enable bypass of the electrical idle function

Chapter 7 : 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.

Enabling Timestamping within ACE

Timestamp signals are made available to the fabric design by setting the **Enable 1588** option to **Yes** within the Ethernet IP wizard.

The figure below details a 100G (4 x 25G) Ethernet channel with 1588 TimeStamping (TSN) signals enabled:

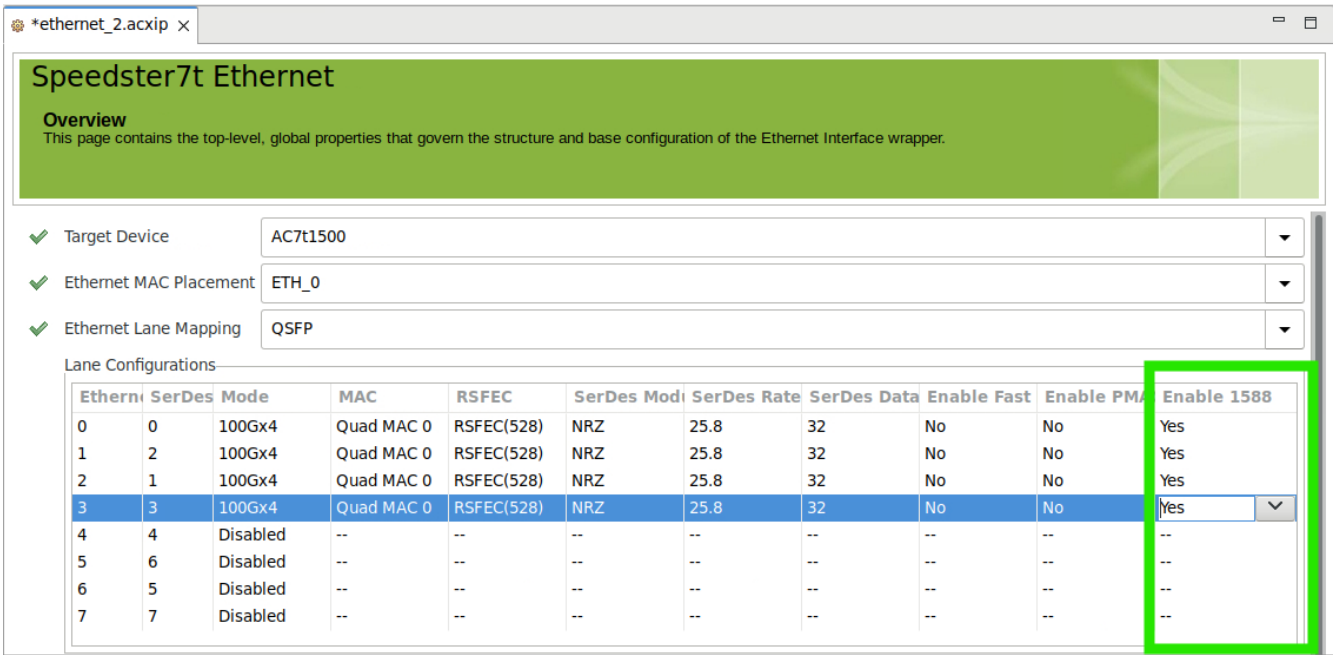


Figure 14 • ACE Enabling 1588 Control

Subsequently, the figure below shows the signals from this configuration output from the Ethernet interface subsystem to the programmable core, when **Enable 1588** is set:

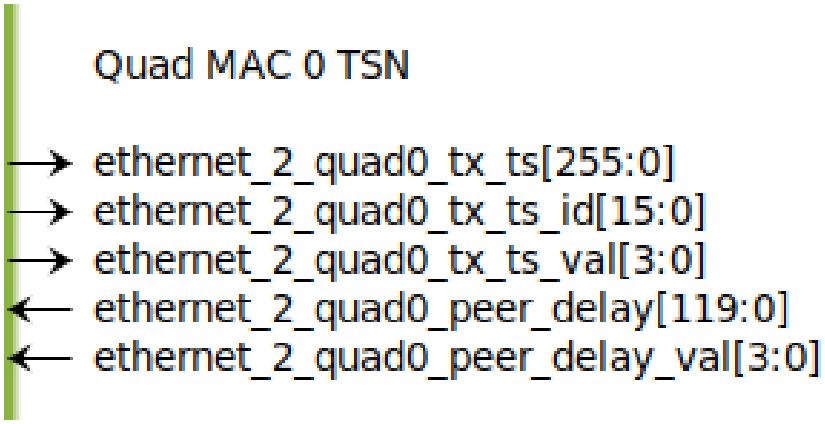


Figure 15 • ACE TSN Control Signals

Designing With TSN Signals

Table 85 • TSN Common Control Signals

Name	Direction (1)	Width	Description (2)
<prefix>_free_running_counter (3)	Input	64	Timestamp counter input. For IEEE-1588 operation: • Must count in nanoseconds. • The lower 32 bits represent nanoseconds and should wrap at 10^9. The upper 32-bits represent seconds. User implementation independent for non-IEEE-1588 operation.

Table Notes

1. Direction is with respect to the Ethernet interface subsystem. The direction is therefore reversed with respect to the user design from the fabric.
2. The counter input to the MAC must be synchronous to <prefix>_ref_clk_divby2.
3. It is recommended to run the counter at 500MHz, with a count of 2 in order to meet the 1ns step size.

Note

Achronix provides the `eth_timestamp_counter` module which generates the correct count, including the optional 10^9 nanosecond wrap if required, and perform the required clock domain crossing from the counter 500MHz to the `<prefix>_ref_clk_divby2` clock domain. In addition, a further output that is CDC crossed to the transmit NAP frequency can be used to add the transmit timestamp. Examples of use of this module are included within the appropriate Ethernet based reference and demo designs.

The `eth_timestamp_counter` module is freely available within many of the reference or demo designs. Reference designs can be downloaded via the Achronix Support website (an Achronix support account is required). Refer to the knowledge base article, **How do I Download Demonstration and Reference Designs for Speedster7t and Speedcore Devices?**[11], which contains demonstration and reference design downloads.

Table 86 • TSN Control Signals per MAC

Name	Direction (1)	Width (2)	Description (3)
`<prefix>_tx_ts`	Output	64	Value of timestamp as packet is transmitted (packet leaves FIFO and enters MAC). Lower 32-bits stored in TIMESTAMP register (see below).
`<prefix>_tx_ts_id`	Output	4	Value of frame identifier as set in **one-step control vector** (page 97).
`<prefix>_tx_ts_val`	Output	1	Asserted for a single clock cycle to validate `tx_ts` and `tx_ts_id`.
`<prefix>_peer_delay`	Input	30	Additional timestamp delay to be added to IEEE-1588 event frame correction field. Calculated by the software from IEEE-1588 PDELAY_REQ/RESP messages.
`<prefix>_peer_delay_val`	Input	1	Valid signal for peer_delay input.

Table Notes

1. Direction is with respect to the Ethernet interface subsystem. The directions are therefore reversed for the user design in the fabric.
2. When enabling TSN Control signals for a quad MAC, all the widths are four times larger as the TSN signals are for all four MACs within a quad.
3. All signals are synchronous to `<prefix>_ref_clk_divby2`

11 https://support.achronix.com/hc/en-us/articles/4405040109204

The TSN signal names, sizes and descriptions are identical regardless of the MAC selected, (Quad MAC, 200G, 400G). However different physical signals are used from the programmable core to the respective MACs. Therefore it is important to connect to the appropriate TSN signals, as enabled by the Ethernet IP within ACE.

> **Note**
>
> Achronix provides the `eth_tsn_control` module which interfaces between the TSN control signals and the register_control_block commonly used in many of the designs. The module supports all necessary signal clock domain crossings necessary between the `<prefix>_ref_clk_divby2` clock and the clock used for the register control block. The module allows software to capture the `tx_ts` and `tx_ts_id` values and to set the peer_delay value. Examples of use of this module are included within the appropriate Ethernet-based reference and demo designs.
>
> The `eth_tsn_control` module is freely available within many of the reference or demo designs. Reference designs can be downloaded via the Achronix Support website (an Achronix support account is required). Refer to the knowledge base article, **How do I Download Demonstration and Reference Designs for Speedster7t and Speedcore Devices?**[12], which contains demonstration and reference design downloads.

Frame Timestamping

All of the following assume that TSN has been enabled for the MAC and that the respective TSN control signals are being driven as described above.

Receive Timestamping

When a packet is received, the MAC latches the value of the receive timestamp field (input from `free_running_counter`). The timestamp, bits [29:0] are presented with the start-of-packet (SoP) pulse in the RX frame flags. See **System Architecture** (page 9), *NAP Data Connections*.

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`)
- The **one-step control vector** (page 100) must be set to indicate the control information
- The timestamp field must be assigned

To enable one-step frame updates, the timestamp field should be assigned the receive timestamp of the associated packet. Alternatively, bits[29:0] of the `free_running_counter` can be assigned to the timestamp.

Transmit Identifier

Within the **one-step control vector** (page 100), the frame identifier can be used by the application to mark specific frames and then use the returned timestamp for those marked frames as required (e.g., 1588 event frames). For example, the MAC could store timestamps and generate interrupts only for frames with a specific identifier and

12 https://support.achronix.com/hc/en-us/articles/4405040109204

ignore all timestamps for frames with other identifiers. The frame identifier is output to the programmable fabric by `tx_ts_id` and validated by the `tx_ts_val` signal.

Capturing Transmit Timestamp

When `frame = 1'b1`, the MAC additionally stores the lower 32-bits of the timestamp for the frame in the register `TS_TIMESTAMP`. The status bit `STATUS.TS_AVAIL` is set to indicate that a new timestamp is available. A user software application would implement a handshaking procedure by setting the `TX frame` flag when it transmits a frame for which it requires a timestamp. The 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 performed for all 1588 event frames. Other frames do not set the `TX frame` flag, and hence, would not interfere with the timestamp capture.

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

IEEE 1588 TX One-Step Frame Field Update

Overview

The IEEE 1588 field update functions are controlled by the one-step control vector, `TX ID vector`, applied as part of the Ethernet NAP transmit flags during each packet transmission, 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.

One-step Control Vector

Table 87 • One-Step Update Control Vector Values

Bit	Description
[3:0] (1)	Frame Identifier. An arbitrary value that can be used to mark specific frames. This value is returned to the fabric in the `tx_ts_id` pins when the frame has been transmitted. The usage of the identifier is transparent to the MAC.
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 is 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 is needed. See UDP Payload Checksum Correction Field Update (page 0) on usage and limitations of this function.
[14:7] (2)	Start offset from the beginning of the frame where the field to update is found (index to the most significant byte). An 8-bit value given in steps of bytes. 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).
15 (3)	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. The field is either `{TX timestamp[31:0], free_running_counter[31:0]}`, or `free_running_counter[63:0]`, depending on the value of `XIF_MODE.TS_UPD64_MODE` control. The first byte of the field is the most significant. The minimum offset for 64-bit field update is 16. • `1'b0` – No update occurs.
16	Unused. Must be set to 1'b0.

Table Notes

1. The ID vector is only valid when `frame=1'b1` in the associated transmit flags (bit[17]).
2. The minimum offset for field update is 16. VLAN tags require adding +4 accordingly.
3. This function sets the field, overwriting the previous frame contents. This does not perform a mathematical correction.

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 specifies that 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**[13] document.

13 https://standards.ieee.org/ieee/1588/6825/

Table 88 • IEEE 1588v2 Message Header (PTPv2)

Bits									
7	6	5	4	3	2	1	0	Octets	Offset
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 `messageId` field as follows (informal only):

Table 89 • PTPv2 Message Type Identification

messageID	Message Name	Message
0x0	SYNC	Event message
0x1	DELAY_REQ	
0x2	PATH_DELAY_REQ	Event message
0x3	PATH_DELAY_RESP	

messageID	Message Name	Message
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 whether or not it wants an update 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 in 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, (bits[31:0] must wrap at 10^9). The user design also requires an accurate clock (not 507MHz) that can give precise nanosecond values, the recommendation is to use 500MHz. When this requirement is met, the correction field update can consider delays up to one second. The Achronix eth_timestamp_counter module provides a suitable timestamp generator.

Since it is recommended to use a 500 MHz clock, this frequency ensures precise timing for data transmission and reception, contributing to a correction accuracy of 2 ns.

Transparent Clock Concepts

Overview

A transparent clock receives frames at a port and forwards them to other ports, updating the frames to correct for the time spent traversing that network equipment instance. The transient time from reception to transmission must 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 initiator at the port where the frame was received (link delay on ingress port).

With reference to the figure below, the timestamps are as follows;

- Rx timestamp – the receive timestamp (value of the Free Running Counter when the packet is received by the MAC) output in the RX flags. 30-bits.
- Tx timestamp (flags) – the timestamp applied to the TX flags. 30-bits. This should be the stored and forwarded Rx timestamp.
- Tx timestamp (actual) – the value of the Free Running Counter at the point the packet is transmitted from the MAC.
- `peer_delay[29:0]` – TSN control value driven from the user application. This value is determined using IEEE-1588 PDELAY_REQ/RESP messages.

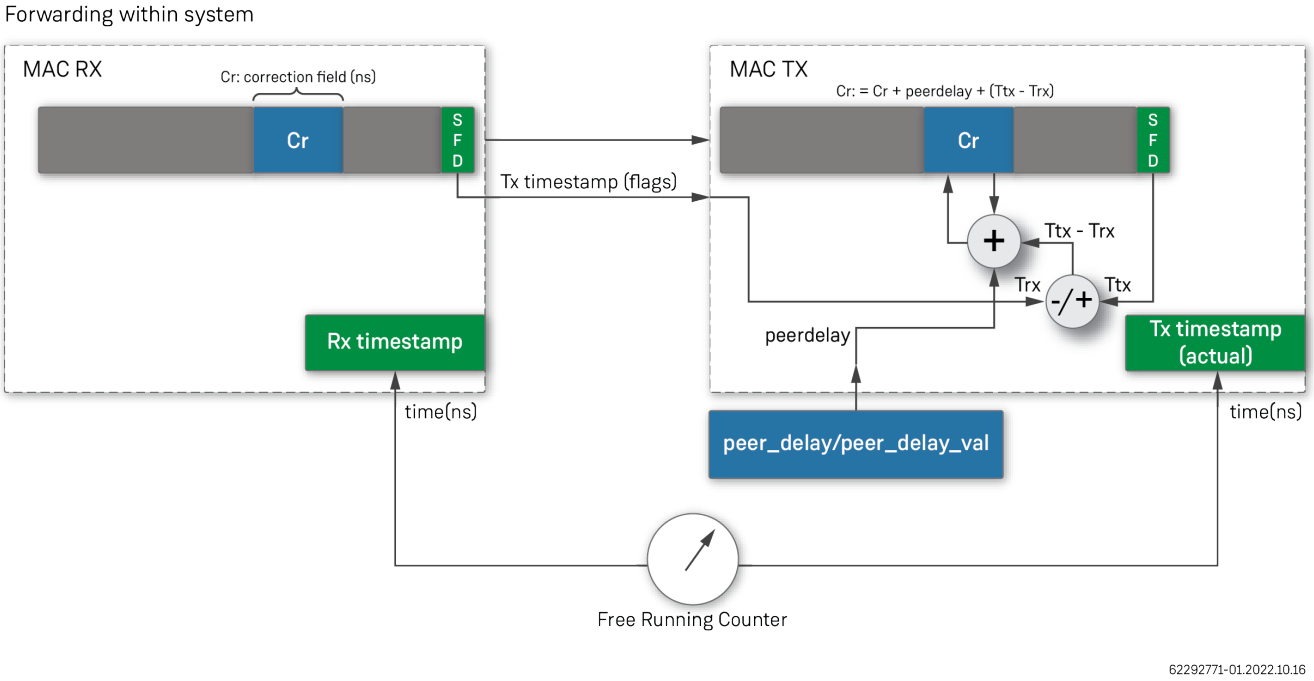


Figure 16 • Transparent Clock Concepts Overview

With the preceding in mind, the designer must consider the following system functions and concepts when implementing transparent clocks:

End-to-End Transparent Clock

The receive `timestamp` (available with every RX frame, using the flags field at SoP) must be forwarded throughout the system and provided to the TX frame in its `timestamp` flags field. In addition, the application must parse the

frame to determine if it is a 1588 event frame requiring an update as well as to identify the transport layer to provide this information to the TX ID vector.

> **Note**
>
> In an end-to-end transparent clock network, PEERDELAY is not used and should be set to 0.

Peer-to-Peer Transparent Clock

In peer-to-peer transparent clocks mode, the forwarding process operates identically to end-to-end transparent 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 user PTP software by periodically transmitting PDELAY_REQ messages at the port where the initiator is connected (they may also be transmitted on all other ports but only the port that connects to the initiator is relevant). The appropriate value of peer delay is then calculated by the user PTP software, and presented to the TSN Control signals peer_delay[29:0], validated with peer_delay_val. See TSN Control Signals per MAC (page 98). The user PTP software is responsible to write the value to the peer_delay/peer_delay_val interface.

As there can only be one initiator 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 initiator are received.

> **Note**
>
> - Achronix provides the eth_tsn_control module that connects directly to the register_control module to enable software to write the peer delay signals.
> - For peer-to-peer transparent clocks, the only forwarded message is a SYNC message from the initiator. All other messages (PDELAY_REQ/RESP) are locally generated and terminated.

Locally Generated Event Messages

When implementing a 1588-capable node, the local system might 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. Implementing an end-node might require creating DELAY_REQ/DELAY_RESP frames when connected to an end-to-end transparent clock environment.

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

With reference to Locally Generated Event Frames (see figure 17) below, the timestamps are as follows:

- Tx timestamp (flags) – the timestamp applied to the TX flags. 30-bits. This should be the nanosecond portion of the System Timer field.
- Tx timestamp (actual) – the value of the System Timer at the point the packet is transmitted from the MAC.

- System Timer – a version of the Free Running Counter, synchronized to the correct time-of-day (TOD) using IEEE-1588 SYNC messages. The clock for this timer may be controlled from either external PLLs or fabric-based digital PLLs. The details of how the System Timer is implemented is outside the scope of this document.

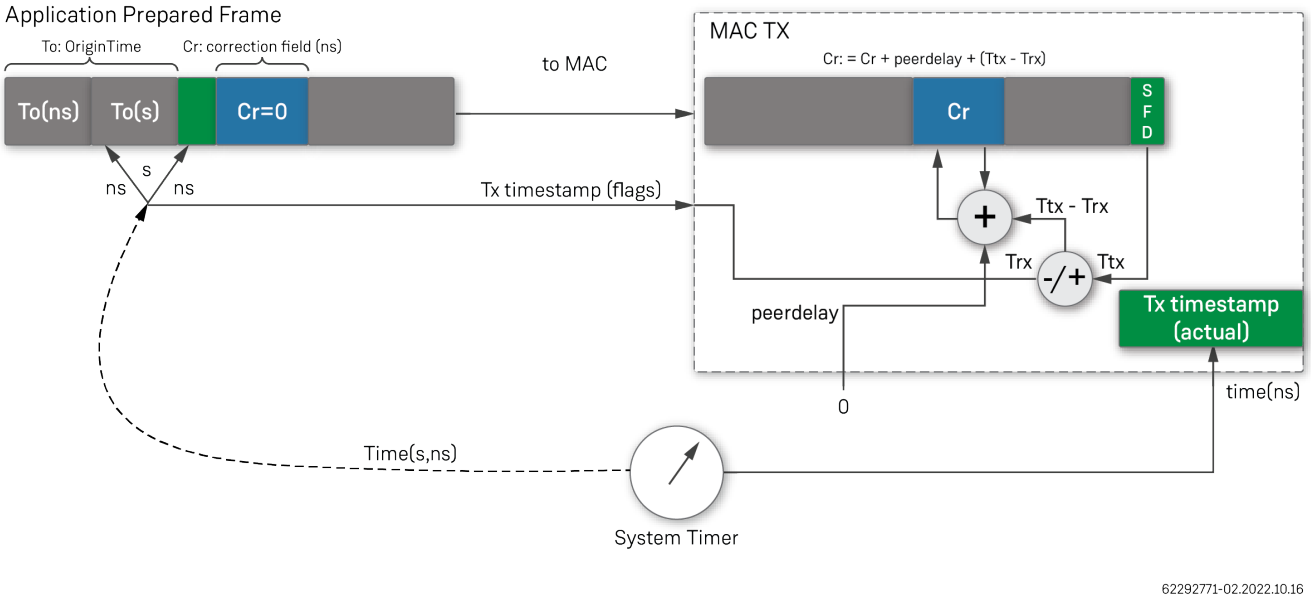


Figure 17 • One-Step Update Concepts for Locally Generated Event Frames

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 the field 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 used to set `originTime` (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, ensuring the precision of the final timestamp value.

> **Note**
>
> The `originTime` correction requires that the message generation (i.e., setting of the `originTime` field) and its actual transmission must occur within one second as the correction field update function can only respect latencies up to one second since 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., a driver) needs to classify the frame and control the MAC update function accordingly through the `TX ID vector` when writing 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 90 • 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 (1)
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 the system role or origination of the frame: • **0**: Local or end-to-end transparent • **1**: Peer-to-peer 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.

Table Notes

1. It is the responsibility of the application to determine these values. The values must be available at start-of-frame (SoP) so the values must be determined before the frame can be presented to the NAP.

UDP Payload Checksum Correction Field Update

The Ethernet interface subsystem supports correction of the UDP/IP payload checksum by modifying the last two bytes within the payload (following the 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 made 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.

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`. This function differs from the IEEE 1588 update function in that it ignores the frame contents and, instead, replaces/overwrites the frame contents with the timestamp value. No UDP checksum update is available in this mode.

Chapter 8 : Ethernet IP Support in ACE

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 selecting the placement and visualizing package routing. When 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

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

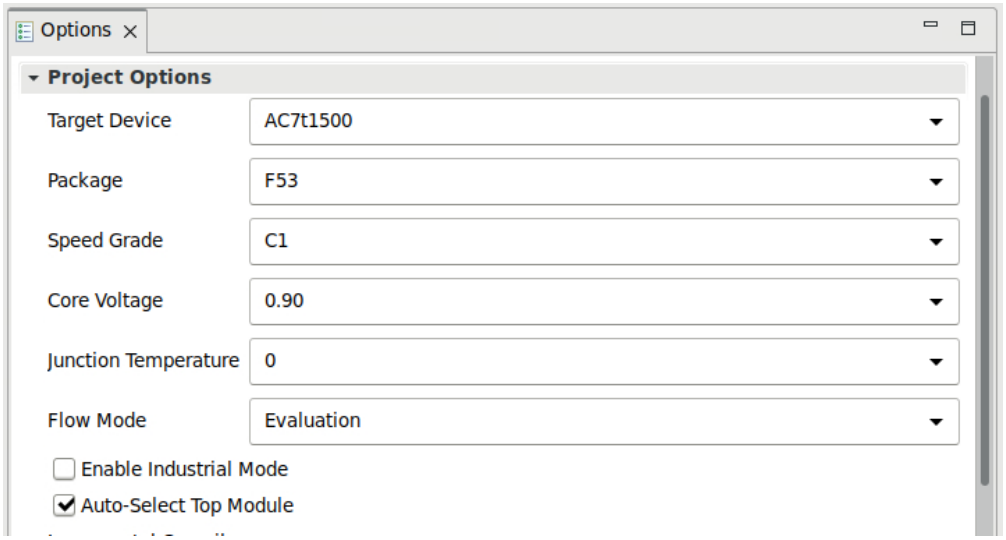


Figure 18 • Design Preparation Options in the ACE Project

Step 2 – Configure and Generate ACE IP

System Clock Input

1. Switch to the 'IP Configuration' perspective and, under **Speedster7t**, select **IO Ring** then select **Clock I/O Bank**.

2. 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).
3. In the **Clock I/O Bank** wizard, select the required clock source. This example uses a single ended reference clock, so select `clock_io_bank_1_msio_p`, and rename it `sys_clk`.
4. 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.

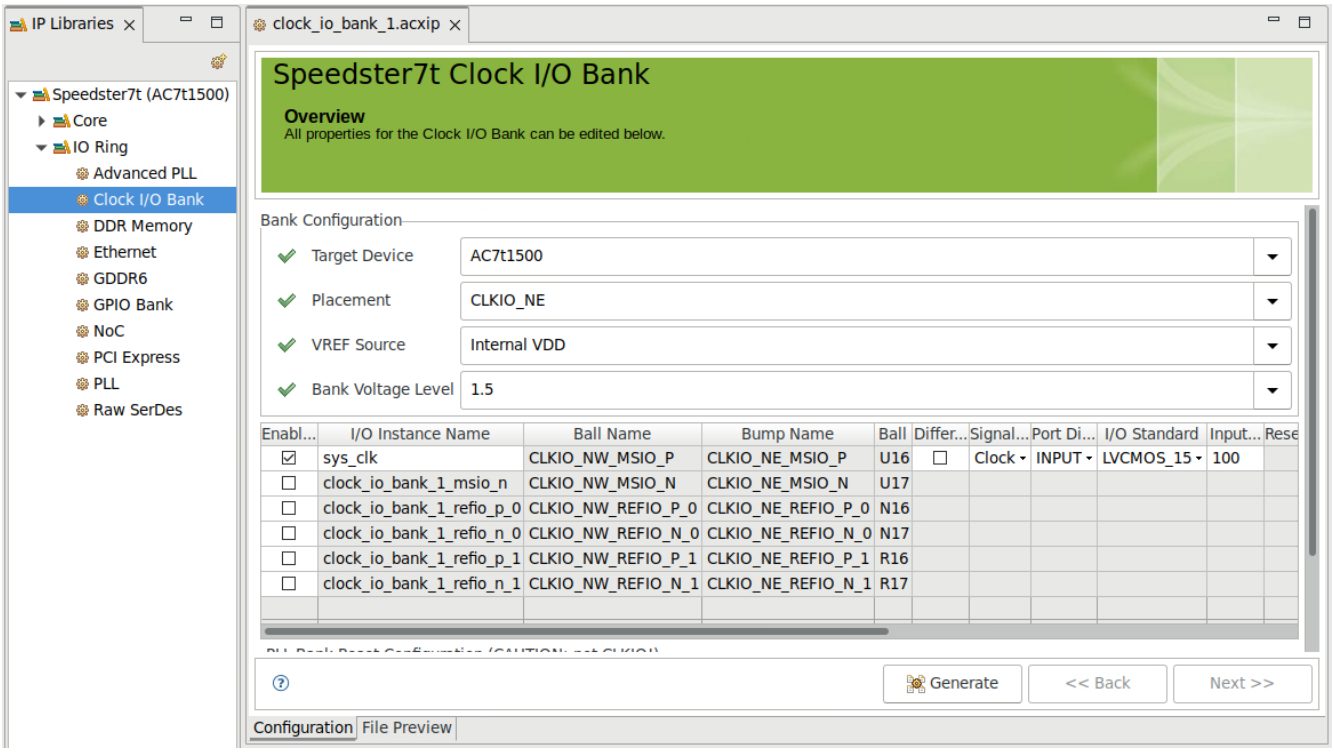


Figure 19 • Clock I/O Configuration in ACE I/O Designer

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 `pll_noc.acxip`.

Table 91 • Settings for noc_pll

Setting	Value
Placement	Compatible location with the system clock input
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.

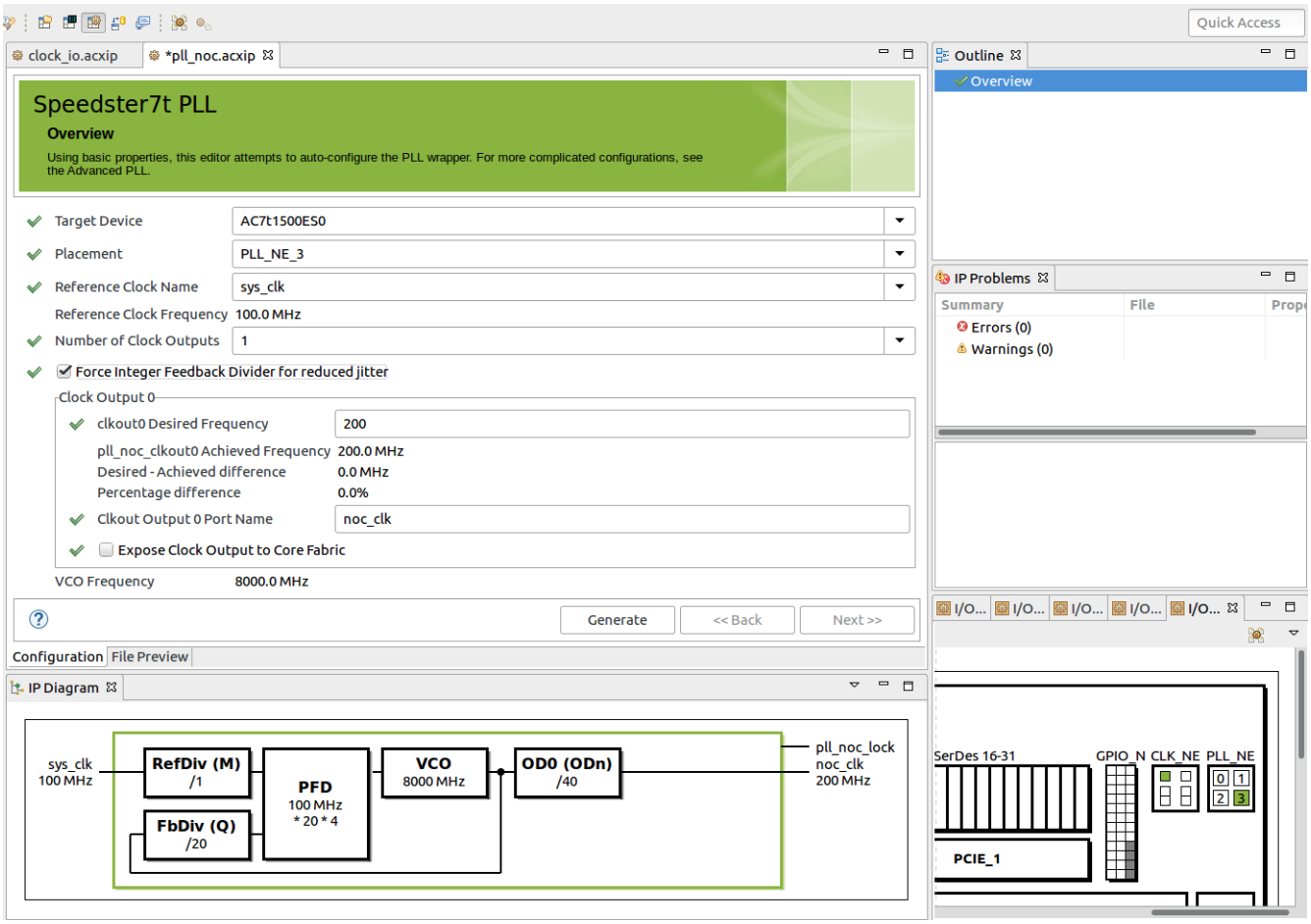


Figure 20 • Configuration of NoC PLL

NoC Clock

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

> **Note**
>
> There are only two settings required to configure the NoC:
>
> - The Target Device, which should already be set to the same device as the project.
> - The NoC Reference Clock Name For this field.

3. Select **noc_clk** from the available drop down list.
4. Confirm that the NoC Reference Clock Frequency is indicated to be 200.0 MHz:

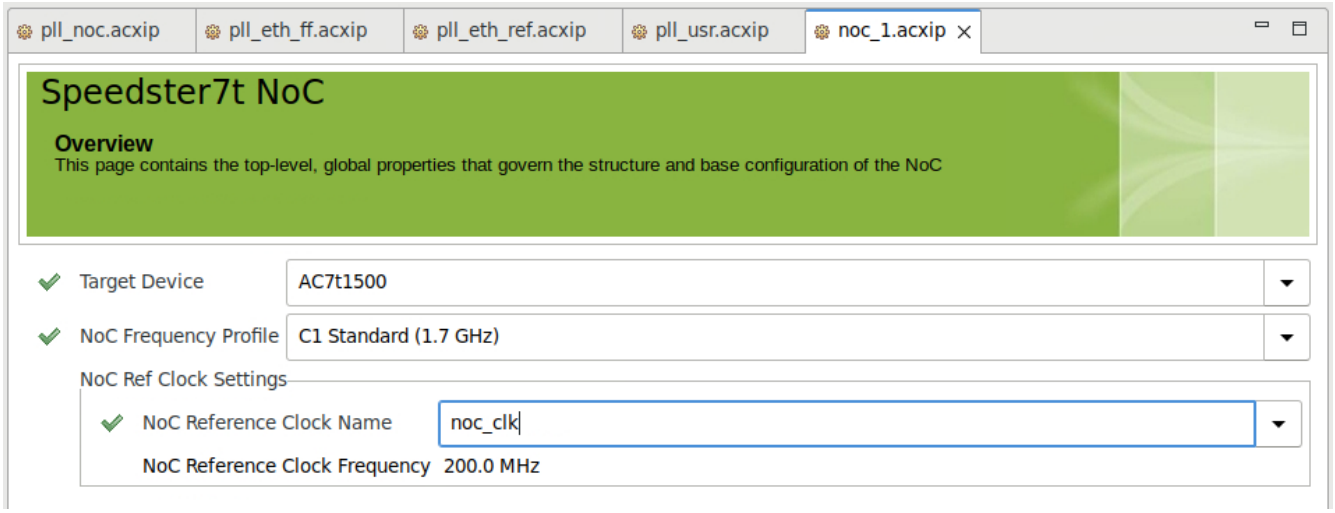


Figure 21 • NoC IP Configuration in ACE I/O Designer

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 92 • Settings for eth_ff_pll

Setting	Value
Placement	Compatible location
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	On
clkout0 Desired Frequency	840

Setting	Value
Clkout Output 0 Port Name	eth_ff_clk
Expose Clock Output to Core Fabric	Off

Table 93 • Settings for eth_ref_pll

Setting	Value
Placement	Compatible location
Reference Clock Name	sys_clk
Number of Clock Outputs	1
Force Integer Feedback Divider for Reduced Jitter	On
clkout0 Desired Frequency	840
Clkout Output 0 Port Name	eth_ref_clk
Expose Clock Output to Core Fabric	Off

Table 94 • Settings for usr_pll

Setting	Value
Placement	Compatible location
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, when 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 Ethernet interface must be configured.

1. From the **IP Libraries**, select **Ethernet**.
2. Select the desired **Ethernet MAC Placement** and **Ethernet Lane Mapping** for the interface.
3. In the **Lane Configurations** table, select the desired Ethernet channel operating modes. For this example, choose the following configuration:

Table 95 • Ethernet Interface IP Settings

Setting	Value
Placement	ETH_0 or ETH_1
Ethernet Lane Mapping	QSFP
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 clock selections should all appear as drop-down menu items if the preceding Ethernet PLLs have been correctly configured.

When the lanes are correctly specified, 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` and `eth_ff_clk`

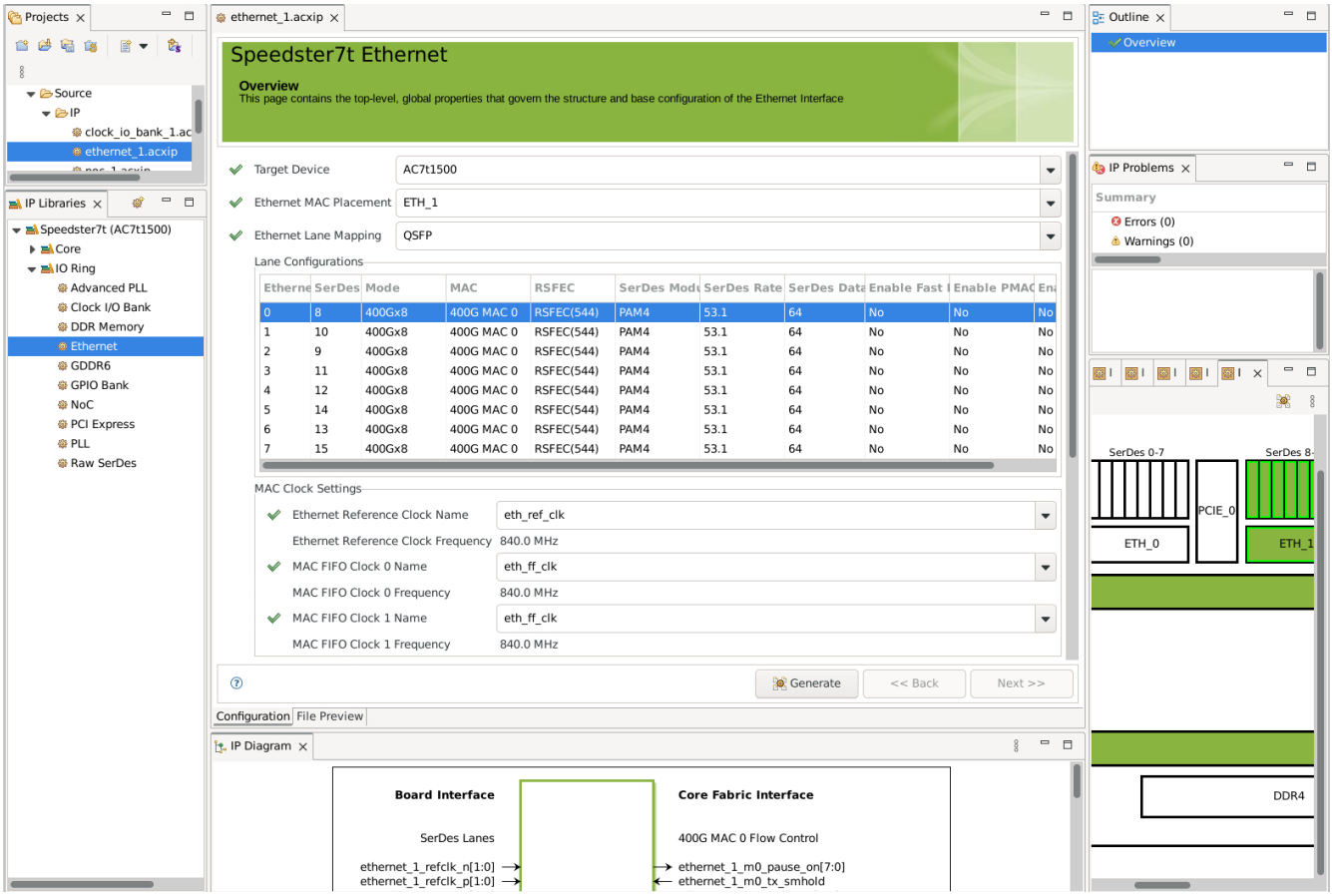


Figure 22 • Ethernet Interface Configuration

If it is intended to build a design with multiple Ethernet interfaces, 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, be assured that the entire I/O interface with all the required IP are integrated properly. When these checks are complete, 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 clocks 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 are specified by this `.pdc` file. This I/O must be instantiated in the top-level design

file (even if 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.

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

Chapter 9 : 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, 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, thus avoiding the 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 (the transmit and receive SerDes bandwidths must be 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 the 40/80-bit SerDes interface is performed

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 loop back 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).

Chapter 10 : 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** (page 96).
2.0	02 Nov 2021	• Add details on NAPs, NAP flags and flow control • Add Lane Mapping options • Add register descriptions
2.1	07 Nov 2022	• Corrected details on IEEE-1588 timestamping. Added details of enabling timestamping within ACE. Aligned diagram signal names with user signals. • Corrected DCI signals • Added additional details on preemptive MACs, and how to enable in ACE
2.2	16 Nov 2026	• Deprecated references to Speedster AC7t1550 device. • Added AC7t800 Information. • Updated AC7t1500 capabilities. • Added sections, **400G MAC Register Map** (page 54) and **400G PCS Register Map** (page 62). • Added the chapter, **Design Considerations** (page 39).