10G Ethernet IP

IP Version: v3.4.0

User Guide

FPGA-IPUG-02245-1.8

December 2025

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
APB	Advanced Peripheral Bus
AXI4-Lite	Advanced eXtensible Interface 4 Lite
AXI4-Stream/AXI4S	Advanced eXtensible Interface 4 Stream
CRC	Cyclic Redundancy Check
DA	Destination Address
DFE	Decision Feedback Equalizer
DIC	Deficit Idle Count
EBR	Embedded Block RAM
FCS	Frame Check Sequence
FIFO	First-In First-Out
GPLL	Generic Phase-Locked Loop
IFG	Inter-Frame Gap
LINTR	Lattice Interrupt Interface
LUT	Look-Up Table
L/T	Length/Type
MAC	Media Access Controller
MDIO	Management Data Input/Output
MMD	MDIO Manageable Device
MTU	Maximum Transmission Unit
PCS	Physical Coding Sublayer
PHY	Physical Layer Device
PMA	Physical Medium Attachment
PTP	Precision Time Protocol
SA	Source Address
SFD	Start Frame Delimiter
SOF	Start of Frame
ToD	Time of Day
TSU	Timestamp Unit
XGMII	10-Gigabit Media Independent Interface

1. Introduction

1.1. Overview of the IP

The Lattice Semiconductor 10G Ethernet (GbE) IP core supports the ability to transmit and receive data between a host processor and an Ethernet network. The 10GbE IP core consists of the 10-Gigabit Media Independent Interface (XGMII), which connects media access controllers (MACs) and Physical Layer devices (PHYs). The main function of the 10GbE MAC is to ensure that the media access rules specified in the 802.3 IEEE standards are met while transmitting a frame of data over an Ethernet. On the receiver side, the Ethernet MAC extracts the different components of a frame and transfers them to higher applications through an AXI4-Stream interface. The PHY implements the physical coding sublayer (PCS) and physical medium attachment (PMA) functionality based on the IEEE 802.3 10GBASE-R specification.

The following figure shows an example of a 10GBASE-R application. The 10GbE MAC is connected to the 10GbE PHY within the 10G Ethernet IP core through the XGMII interface. The clock source to the Ethernet MAC is from xg_tx_gclk_o[0] (clock output from the Ethernet PHY).

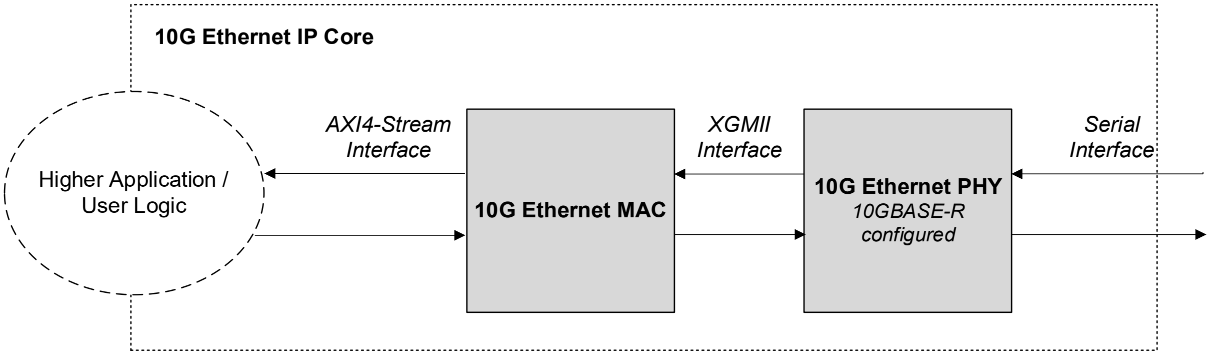


Figure 1.1. 10GBASE-R Application

1.2. Quick Facts

The following table provides quick facts about the 10G Ethernet IP core.

Table 1.1. Summary of the 10G Ethernet IP

IP Requirements	Supported Devices	CertusPro™-NX (Speed grade 9 only for PHY support)	Avant™-AT-G and Avant-AT-X
	IP Option	• MAC only • PHY only • MAC + PHY + 1588[1] • MAC + PHY	• MAC only • PHY only • MAC + PHY
	IP Changes[2]	For a list of changes to the IP, refer to the 10G Ethernet IP Release Notes (FPGA-RN-02031).	
Resource Utilization	Supported User Interface	AXI4-Stream/APB/AXI4-Lite interface.	
	Resources	See Appendix A. Resource Utilization.	
Design Tool Support	Lattice Implementation	IP core v3.4.0—Lattice Radiant™ software 2025.2 or later.	
	Synthesis	Synopsys® Synplify Pro® software, O-2018.09LR-SP1.	
	Simulation	For the list of supported simulators, see the Lattice Radiant Software User Guide.	
Driver Support	API Reference	Refer to the 2.5G, 10G, and 25G Ethernet Driver API Reference (FPGA-TN-02375).	

Notes:

1. PHY + MAC + 1588 option currently only supports 2-step mode. The 2-step hardware validation and 1-step mode is planned for future milestones.

2. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

1.3. IP Support Summary

The following table provides IP support information on the 10G Ethernet IP core.

Table 1.2. 10G Ethernet IP Support Readiness

Device Family	Mode	Radiant Timing Model	Hardware Validated
Avant-AT-G/X	MAC only, PHY only, MAC + PHY	Preliminary	Yes
CertusPro-NX	MAC only, PHY only	Final	Yes
	MAC + PHY + 1588	Final	No

1.4. Features

The 10G Ethernet IP core offers the following key features:

MAC

- Compliant to the IEEE 802.3-2012 standard
- Supports standard 10 Gbps Ethernet link layer data rate
- 64-bit wide internal datapath operating at 156.25 MHz
- AXI4-Stream interface on the client's transmit and receive interfaces
- Supports deficit idle count (DIC)
- Supports VLAN and Jumbo frames of 9,600 bytes
- Custom preamble mode*
- Independent TX and RX Maximum Transmission Unit (MTU) frame length
- Comprehensive statistics support
- Optional frame check sequence (FCS) generation on transmission
- Optional FCS stripping during reception
- Optional multicast address filtering
- Programmable Inter-Frame Gap
- Supports flow control using pause frames
- Automatic padding of short frames
- Inter-Frame Stretch Mode during transmission
- Supports full-duplex operation
- Advanced Peripheral Bus (APB) interface or AXI4-Lite interface for register access
- Supports 8-bit/16-bit GMII or 4-bit MII to and from PHY layer
- Programmable promiscuous (transparent) mode

*** Note:** Only supported by XGMII interface.

PHY

- Designed to the IEEE 802.3-2012 10GBASE-R specification
- 64b/66b encoding and decoding
- XGMII interface: 64-bit, 156.25 MHz
- Supports AXI4-Lite and management data input/output for PCS
- TX phase FIFO and RX clock compensation FIFO (CertusPro-NX devices only)
- Support APB interfaces (CertusPro-NX devices only)
- Support FCFEC

MAC + PHY + 1588 (CertusPro-NX)

- Timestamping support as specified in the IEEE 1588v2 (2-step mode)

1.5. Licensing and Ordering Information

The 10G Ethernet IP is available with Lattice Radiant Subscription Software. To purchase the Lattice Radiant Subscription license, contact Lattice Sales or go to the Lattice Online Store.

1.5.1. Hardware Evaluation

The 10G Ethernet IP core supports Lattice's IP hardware evaluation capability when used with CertusPro-NX, Avant-AT-G, and Avant-AT-X devices. The hardware evaluation capability enables you to create versions of the IP core that operate in hardware for a limited period (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs. The hardware evaluation capability may be enabled or disabled in the **Strategy** dialog box. It is enabled by default. To change this setting, go to **Project > Active Strategy > Synplify Pro Settings**.

1.6. Hardware Support

Refer to the Example Design section for more information on the boards used.

1.7. Minimum Device Requirements

The minimum device requirements for the 10G Ethernet IP core are as follows:

Table 1.3. Minimum Device Requirements for 10G Ethernet IP Core

Devices	Speed Grades
Avant-AT-G/X	All speed grades
CertusPro-NX	Speed grade 9

1.8. Naming Conventions

1.8.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.8.2. Signal Names

Signal names that end with:

- _*n* are active low signals (asserted when value is logic 0)
- _*i* are input signals
- _*o* are output signals
- _*io* are bi-directional input/output signals

2. Functional Description

2.1. MAC + PHY (Avant Devices)

2.1.1. IP Architecture Overview

The 10G Ethernet IP core transmits and receives data between a host processor and an Ethernet network. With the *MAC + PHY* option selected, the IP consists of a MAC and a PHY that is connected through the XGMII internally.

This IP option is supported on Avant-AT-G and Avant-AT-X devices.

For Lane Merging details, refer to the Lane Merging (Avant Devices) section.

2.1.2. Block Diagram

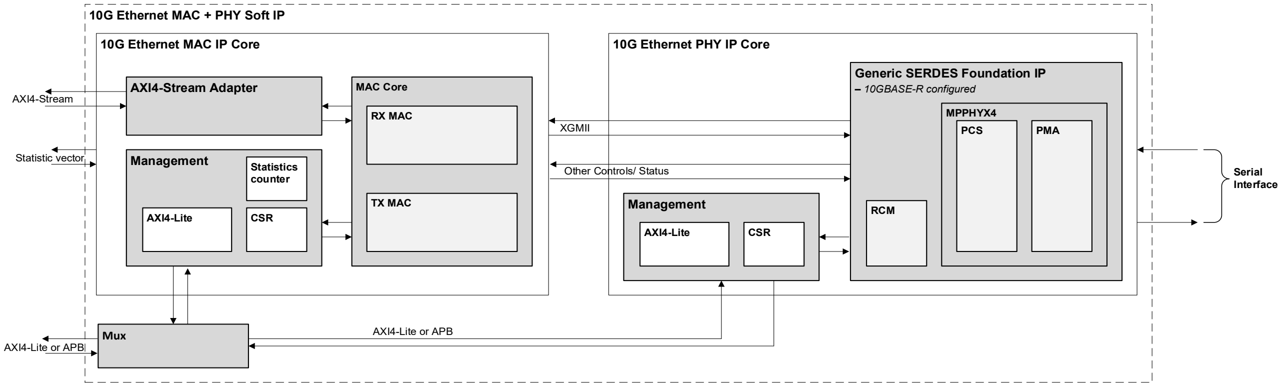


Figure 2.1. 10GbE MAC + PHY IP Core Block Diagram

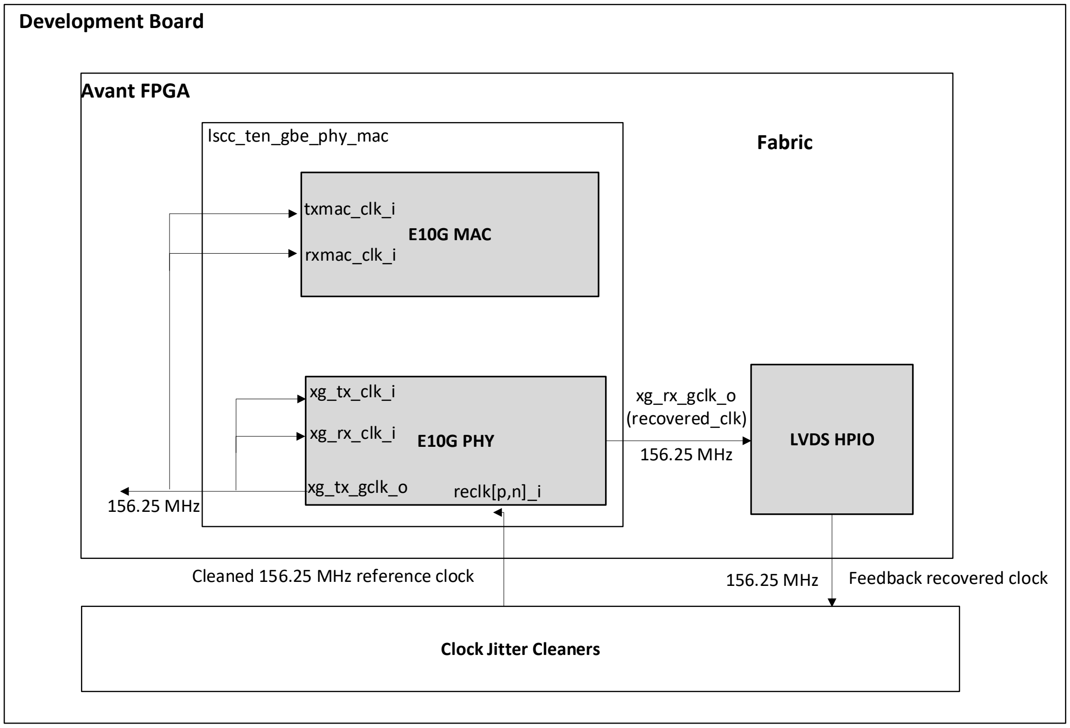


Figure 2.2. 10GbE MAC + PHY IP Core SyncE Block Diagram

2.1.3. Management Block

The Management block can be accessed through the AXI4-Lite or APB interface when the *MAC + PHY* option is selected.

For more details about the management blocks in MAC and PHY respectively, refer to the MAC Management Block and PHY Management Block sections. For protocol timing details, refer to AMBA 4 AXI and ACE Protocol Specification issue H or AMBA 3 APB Protocol version 1.0 Specification.

2.2. MAC

2.2.1. IP Architecture Overview

The main function of the 10G Ethernet MAC IP is to ensure that the media access rules specified in the 802.3 IEEE standards are met while transmitting a frame of data over an Ethernet. On the receiver side, the Ethernet MAC extracts the different components of a frame and transfers them to higher applications through an AXI4-Stream interface.

2.2.2. Block Diagram

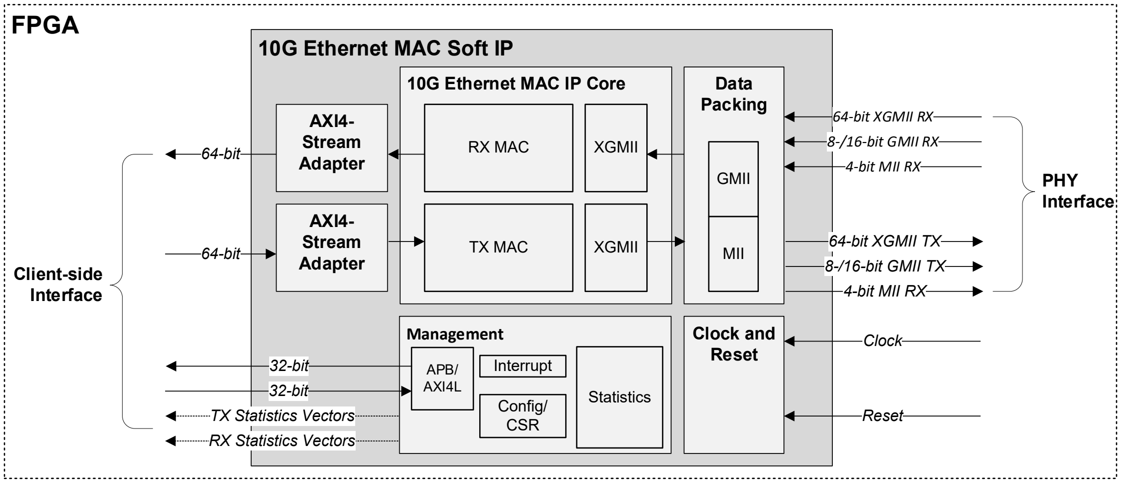
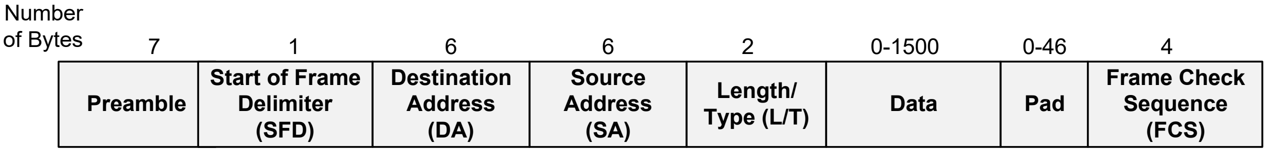


Figure 2.3. 10G Ethernet MAC IP Core Block Diagram

2.2.3. Ethernet Data Format

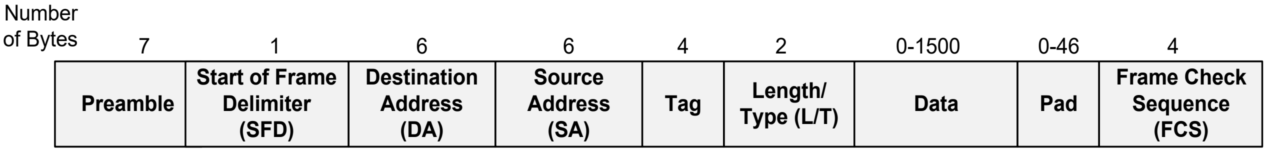
The following figure shows an untagged Ethernet frame format.



Number of Bytes	7	1	6	6	2	0-1500	0-46	4
	Preamble	Start of Frame Delimiter (SFD)	Destination Address (DA)	Source Address (SA)	Length/ Type (L/T)	Data	Pad	Frame Check Sequence (FCS)

Figure 2.4. Untagged Ethernet Frame Format

The following figure shows a tagged Ethernet frame format.



Number of Bytes	7	1	6	6	4	2	0-1500	0-46	4
	Preamble	Start of Frame Delimiter (SFD)	Destination Address (DA)	Source Address (SA)	Tag	Length/ Type (L/T)	Data	Pad	Frame Check Sequence (FCS)

Figure 2.5. Tagged Ethernet Frame Format

The MAC is responsible for constructing a valid frame from the data received from the client before transmitting it. On the receiver path, it receives frames from the network through XGMII interface and passes the parameters of the frame to the client through the AXI4-Stream interface.

The fields that are expected from the client-side interface can be one of the following:

- The frame contains the FCS along with the destination address (DA), source address (SA), length/type (L/T), and data. The TX MAC adds the preamble and start frame delimiter (SFD) before transmitting the frame. This mode can be set by enabling tx_pass_fcs bit in the TX_CTL register. For timing details, see the In-Band FCS Passing section.
- The TX MAC calculates the number of bytes to be padded as well (if required) in addition to the FCS for the entire frame and adds the preamble and SFD before transmitting the frame. For timing details, see the Default Normal Frame section.
- When tx_pass_pream bit is set in the TX_CTL register, the client supplies a custom data in the preamble and SFD field. The TX MAC preserves the preamble field and passed it directly to physical layer. This applies to both cases mentioned above. For timing details, see the Custom Preamble Passing section.

On the receiver path, the MAC can be programmed to transfer frame in one of the following cases:

- Transfer the frame after stripping off the FCS and any pad fields. This is the default settings of the RX MAC.
- Transfer the frame with the FCS field and any pad fields by setting the rx_pass_fc bit of RX_CTL register.
- Transfer the frame in promiscuous mode by setting the prms bit of RX_CTL register. This mode transfers all frames it receives to the client rather than passing only the frames that is specifically programmed to receive.

In all cases, the preamble and SFD bytes are always stripped off the frame before it is transferred on the AXI4-Stream interface unless the rx_pass_pream bit is set in the RX_CTL register. When rx_pass_pream bit is set, the RX MAC does not check for the SFD.

2.2.4. Receive MAC

The Receive MAC receives the incoming frames and transfers them to the client via the AXI4-Stream interface. In the process, it performs the following operations:

- Checks the frame for a valid start of frame (SOF) and SFD symbol. This checking is disabled when RX is configured to custom preamble mode.
- Determines whether the frame should be received by analyzing the DA.
- Determines the type of the frame by analyzing the length/type field.
- Checks for any errors in the frame by recalculating the CRC and comparing it with the expected value.

The RX MAC operation is determined by programming the MODE and RX_CTL registers.

You can specify whether the FCS field should be transferred on to the AXI4-Stream interface by programming the rx_pass_fcs bit of RX_CTL register. If the FCS field is to be stripped off the frame, any padding bytes within the frame are stripped off as well.

Once a valid SOF is detected, the DA field of the incoming frame is analyzed. If the DA field is a unicast address, it is compared with the programmed MAC address. Unless the prms bit of RX_CTL register was set, the incoming frame is discarded if the DA field and the programmed MAC address (MAC_ADDR_{0,1} registers) do not match. If the frame had a multicast address and if receive_all_mc signal is not asserted, all such frames are dropped (except PAUSE frames). If the frame had a multicast address and if receive_all_mc signal is asserted, the multicast frames are subject to the following address filtering rules:

- For all frames with multicast address, the CRC of the destination address is computed and the mid-six bits of the least significant byte of the CRC is chosen as the address to a hash table. The 64-bit hash table is programmed in the MC_TABLE_{0,1} registers. The MAC implements an eight-row table with eight bits in each row. The lower three bits of the selected CRC are used to select one of these eight rows and the next three bits are used to select one of the bits in the selected table. The incoming multicast frame is accepted if the bit selected from the hash table is set to one. It is discarded if the bit selected is zero.

If the incoming frame had a broadcast address, it is accepted if either the prms or the receive_bc bit of RX_CTL register is set. A broadcast frame is discarded if none of these bits are set.

2.2.5. Transmit MAC

The Transmit MAC controls access to the physical medium. Its main functions are as follow:

- Data padding for short frames when FCS generation is enabled.
- Generation of a pause frame when the tx_pausreq bit of MAC_CTL register is asserted. The bit tx_fc_en of TX_CTL register should be set to 1 to enable this feature.
- To stop frame transmission when a pause frame is received by the Receive MAC.
- Implement link fault signaling logic and transmit appropriate sequences based on the remote link status of the TX_RX_STS register.

The TX MAC operation is determined by programming the MODE and TX_CTL registers.

By default, the Transmit MAC is configured to generate the FCS pattern for the frame to be transmitted. However, this can be prevented by setting tx_pass_fcs bit of the TX_CTL register. This feature is useful if the frames being presented for transmission already contain the FCS field. When FCS field generation by the MAC is disabled, ensure that short frames are properly padded before the FCS is generated. If the MAC receives a frame to transmit that is shorter than 64 bytes when FCS generation is disabled, the frame is sent as is and a statistic vector for the condition is generated.

The DA, SA, L/T, and data fields are derived from higher applications through the AXI4-Stream interface and then encapsulated into an un-tagged Ethernet frame. The frame encapsulation consists of adding the preamble bits, the SFD bits, and the CRC check sum to the end of the frame. If transmit_short bit of TX_CTL register is not set, all short frames are padded.

The frame is not sent over the network until the network has been idle for a minimum of Inter-Packet Gap (IPG) of 12 bytes. The IPG is adjustable through the tx_ipg field of the IPG_VAL register. The Transmit MAC uses DIC to reach an average IPG of 8 bytes + (tx_ipg x 4 bytes). With the default tx_ipg value of 1, the Transmit MAC aims for an average IPG of 12 bytes. For more information on DIC, refer to the IEEE 802.3-2012 Section 46.

The TX MAC requires a continuous stream of data for the entire frame. There cannot be any bubbles of no data transfer within a frame. After the TX MAC is done transmitting a frame, it waits for more frames from the AXI4-Stream interface. During this time, it goes to an idle state that can be detected by reading the TX_RX_STS register. Because the MODE register can be written at any time, the TX MAC can be disabled while it is actively transmitting a frame. In such cases, the MAC completely transmits the current frame and then return to the idle state. The control registers should be programmed only after the MAC has returned to the IDLE state.

There are two different methods for transmitting a pause frame. In the first method, the application layer forms a pause frame and submits it for transmission via the AXI4-Stream interface. In the other method, the application layer signals the TX MAC directly to transmit a pause frame. This is accomplished by asserting tx_pausreq bit of MAC_CTL register. In this case, the TX MAC completes the transmission of the current packet and then transmit a pause frame with the pause time value supplied through the tx_paustim bits of the PAUSE_TMR register.

2.2.6. Receive AXI4-Stream Interface

The receive client-side interface supports the AXI4-Stream interface.

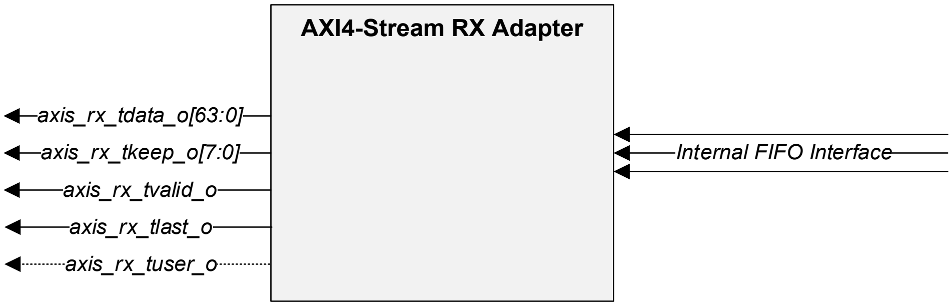


Figure 2.6. AXI4-Stream RX Adapter Interface Diagram

2.2.6.1. Default Normal Frame

The following figure shows the timing diagram of a default normal frame at the Receive AXI4-Stream interface:

- T0: MAC asserts axis_rx_tvalid_o to indicate start of frame. The valid packet includes from the DA field up to the end of the data field.
- T1: MAC asserts axis_rx_tlast_o to indicate last packet transfer.
- T2: MAC deasserts axis_rx_tvalid_o when there is no next packet transfer.

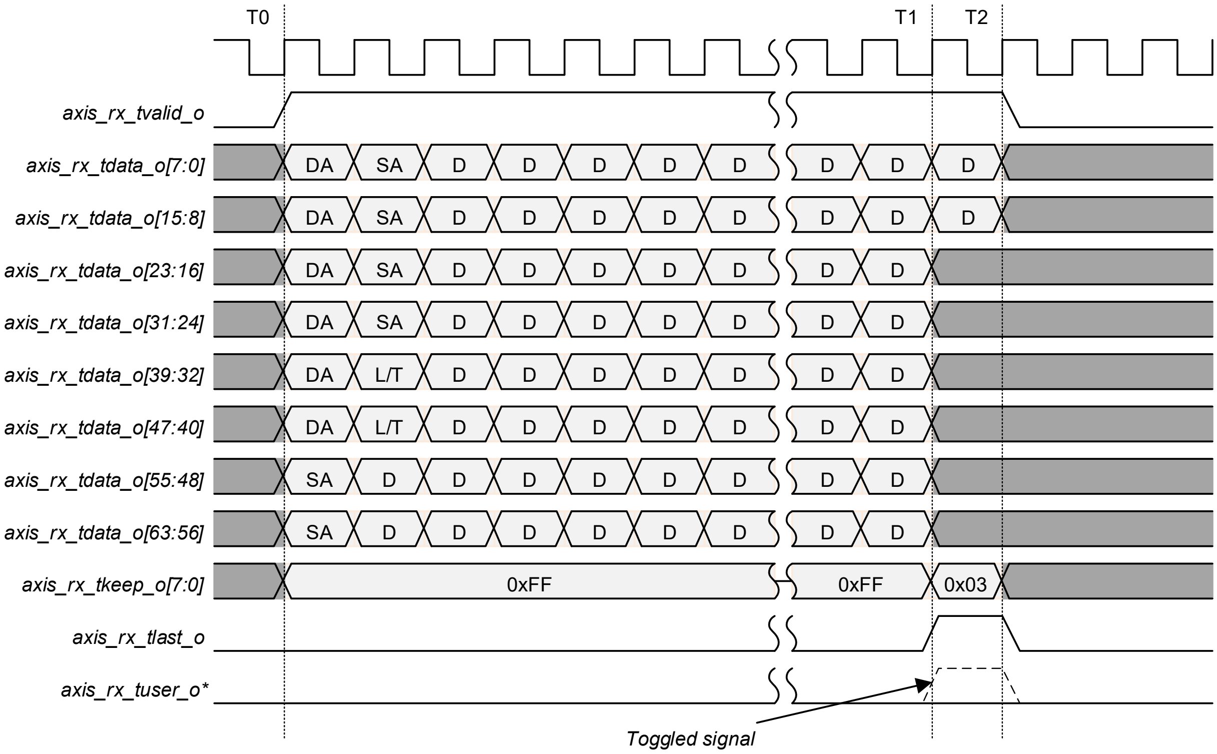


Figure 2.7. Normal Frame Reception

*** Note:** Under normal circumstances, the axis_rx_tuser_o signal must not be toggled when there is no error. However, if an error occurs and the axis_rx_tuser_o signal is toggled, you must check on the status signal—rx_statvec_o signal.

The following figure shows the timing diagram of a back-to-back frame at the Receive AXI4-Stream interface:

- T0: MAC asserts axis_rx_tvalid_o to indicate start of frame. The valid packet starts from the DA field up to the end of the data field.
- T1: MAC asserts axis_rx_tlast_o to indicate last packet transfer.
- T2: MAC still asserts the axis_rx_tvalid_o to indicate another packet transfer.

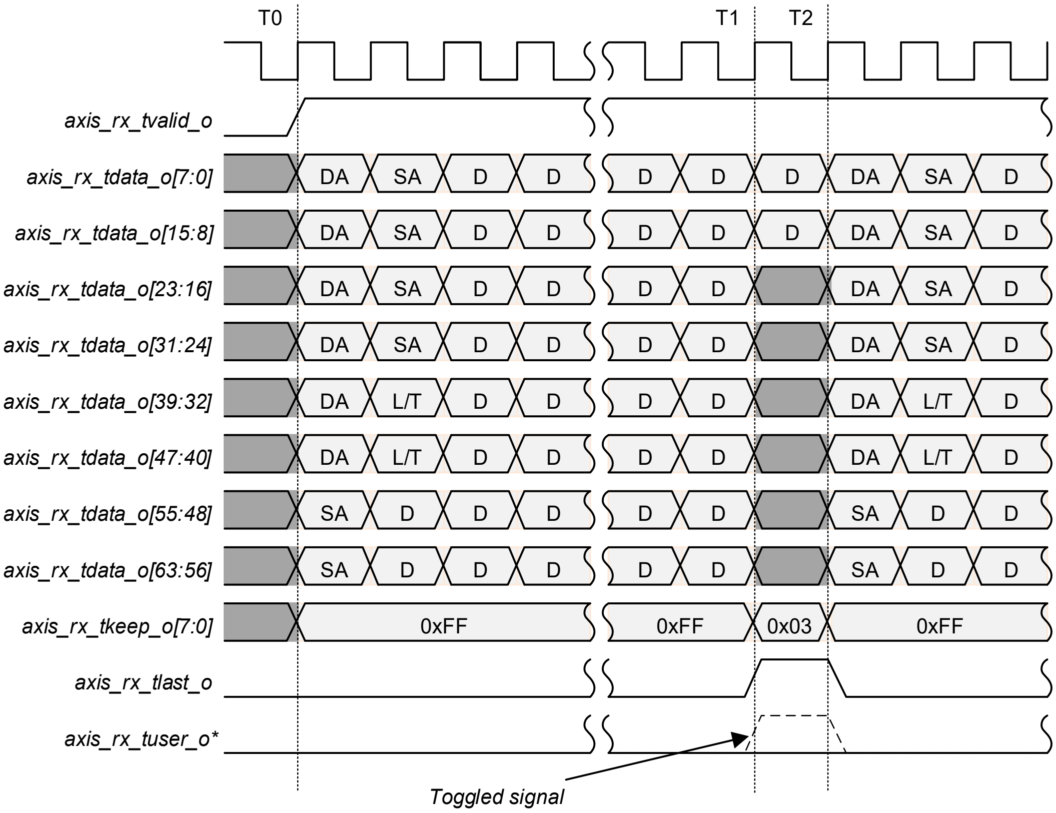


Figure 2.8. Back-to-back Frames Reception

*** Note:** Under normal circumstances, the axis_rx_tuser_o signal must not be toggled when there is no error. However, if an error occurs and the axis_rx_tuser_o signal is toggled, you must check on the status signal—rx_statvec_o signal.

2.2.6.2. In-Band FCS Passing

The following figure shows the timing diagram of a frame with in-band FCS. The FCS field is included inside the valid frame.

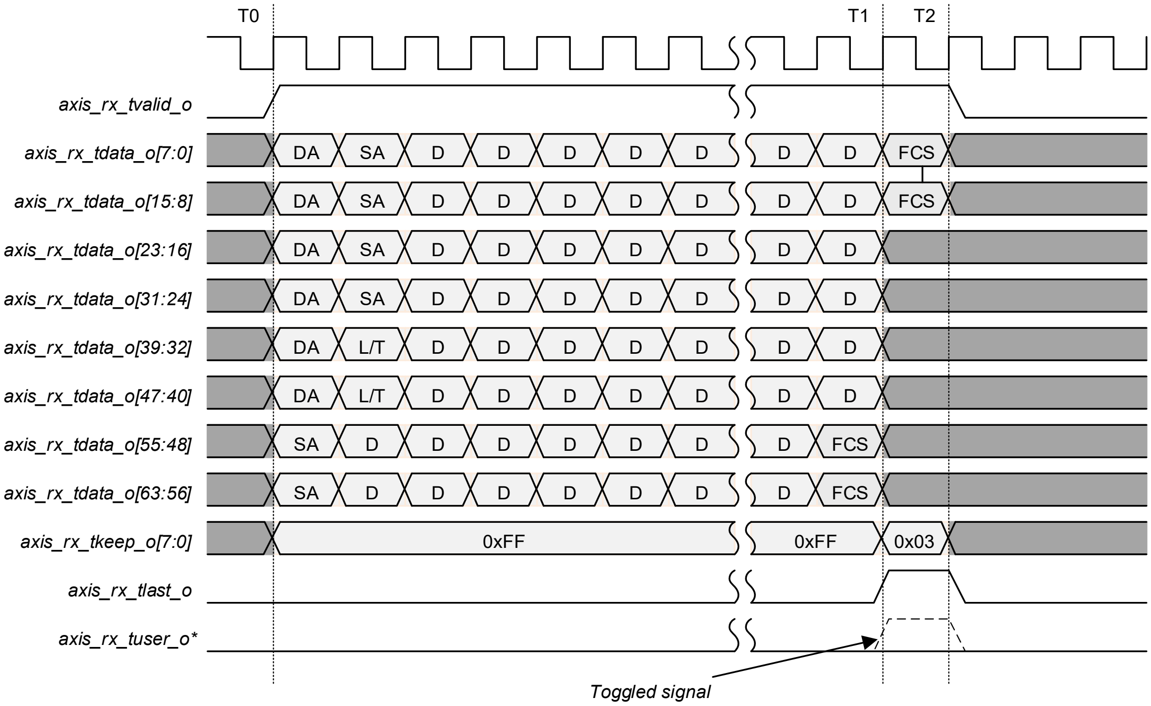


Figure 2.9. Frame Reception with In-Band FCS Passing

*** Note:** Under normal circumstances, the axis_rx_tuser_o signal must not be toggled when there is no error. However, if an error occurs and the axis_rx_tuser_o signal is toggled, you must check on the status signal—rx_statvec_o signal.

2.2.6.3. Custom Preamble Passing

The following figure shows the timing diagram of a frame with custom preamble mode enabled. A custom preamble (CP) field is included inside the valid frame.
Note: Custom preamble mode is only supported by XGMII interface.

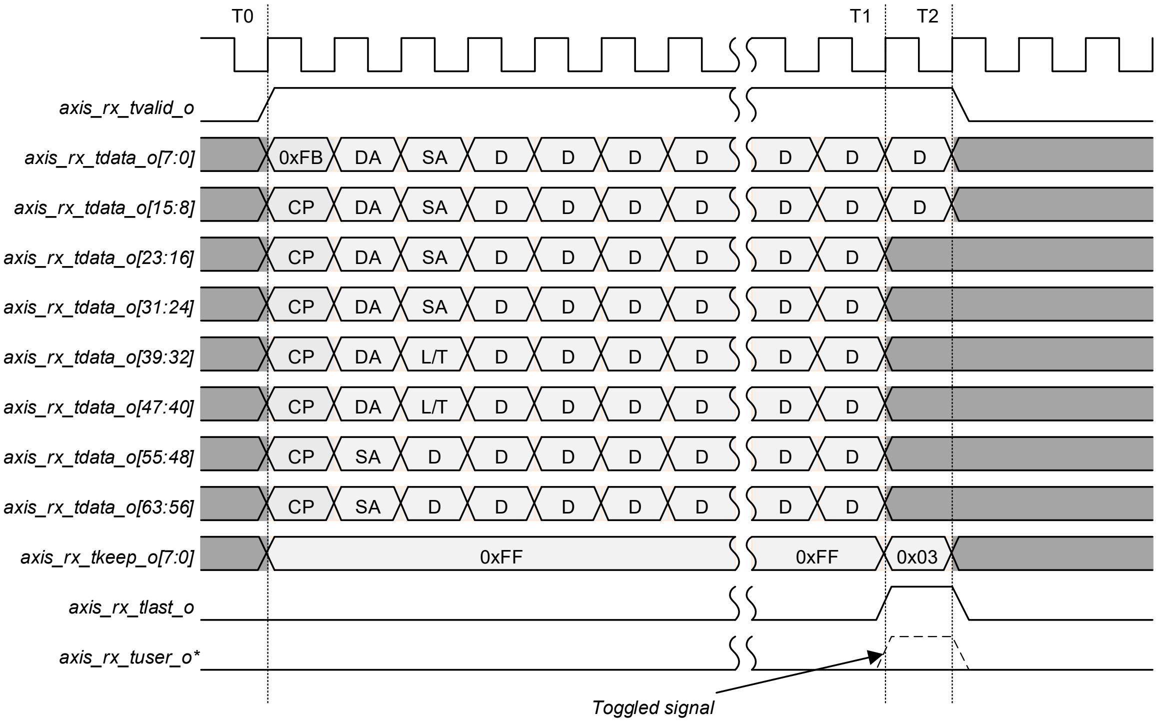


Figure 2.10. Reception with Custom Preamble

*** Note:** Under normal circumstances, the axis_rx_tuser_o signal must not be toggled when there is no error. However, if an error occurs and the axis_rx_tuser_o signal is toggled, you must check on the status signal—rx_statvec_o signal.

2.2.7. Transmit AXI4-Stream Interface

The transmit client-side interface supports the AXI4-Stream interface.

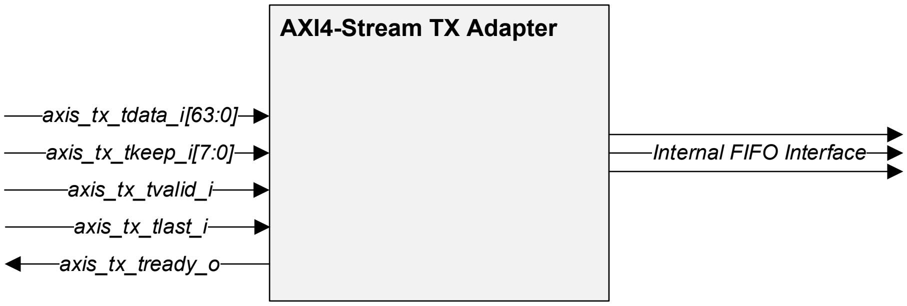


Figure 2.11. AXI4-Stream TX Adapter Interface Diagram

2.2.7.1. Default Normal Frame

The following figure shows the timing diagram of a default normal frame at the Transmit AXI4-Stream interface:

- T0: Client asserts axis_tx_tvalid_i to indicate the start of packet transfer. It must hold the data and axis_tx_tvalid_i until axis_tx_tready_o asserts.
- T1: Client continues to send data once it sees axis_tx_tready_o asserted.
- T2: Client asserts axis_tx_tlast_i to indicate last packet transfer.
- T3: Client deasserts axis_tx_tvalid_i when there is no packet transfer. Transmit MAC also deasserts axis_tx_tready_o once it sees the last packet transfer (axis_tx_tlast_i = 1).

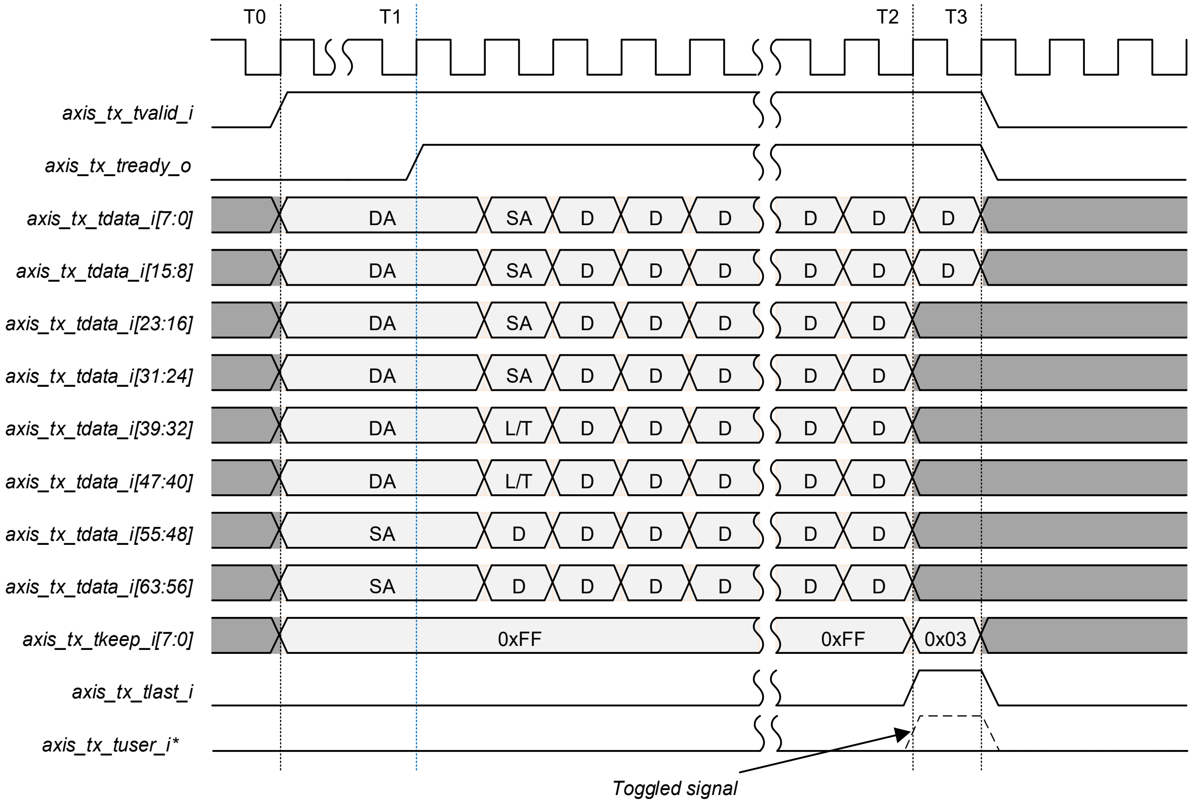


Figure 2.12. Default Normal Frame Transmission

* **Note:** You can inject the error by asserting the axis_tx_tuser_i signal during EOP. Otherwise, the axis_tx_tuser_i signal must drive to low.

2.2.7.2. In-Band FCS Passing

The following figure shows the timing diagram of a frame when In-Band FCS passing is enabled.

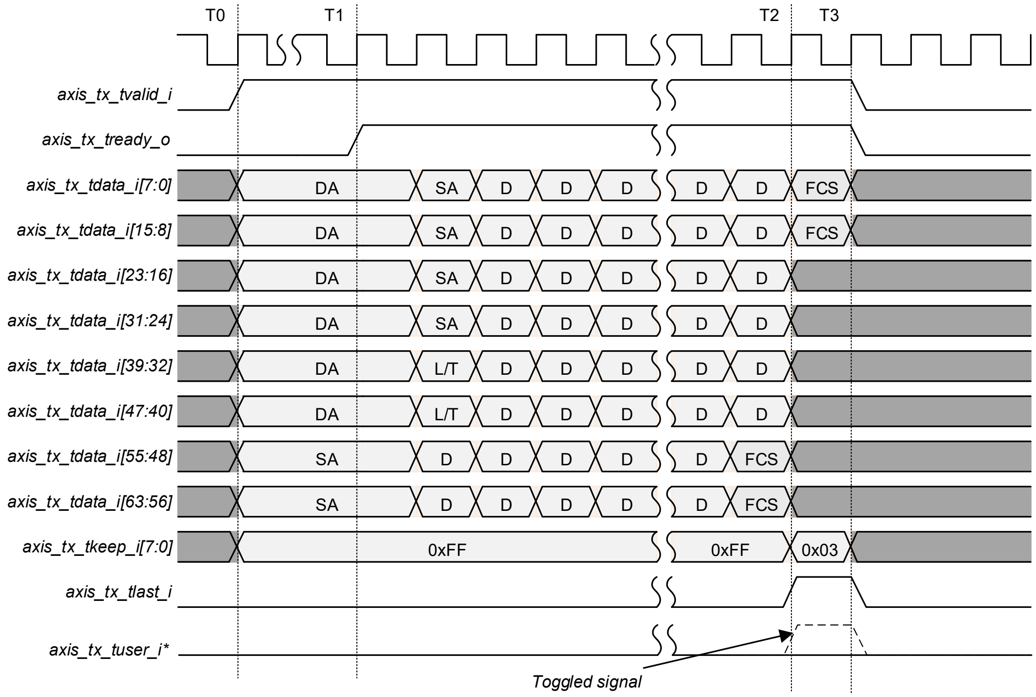


Figure 2.13. Transmission with In-Band FCS Passing

*** Note:** You can inject the error by asserting the axis_tx_tuser_i signal during EOP. Otherwise, the axis_tx_tuser_i signal must drive to low.

2.2.7.3. Custom Preamble Passing

The following figure shows the timing diagram of a frame when custom preamble passing is enabled.
Note: Custom preamble mode is only supported by XGMII interface.

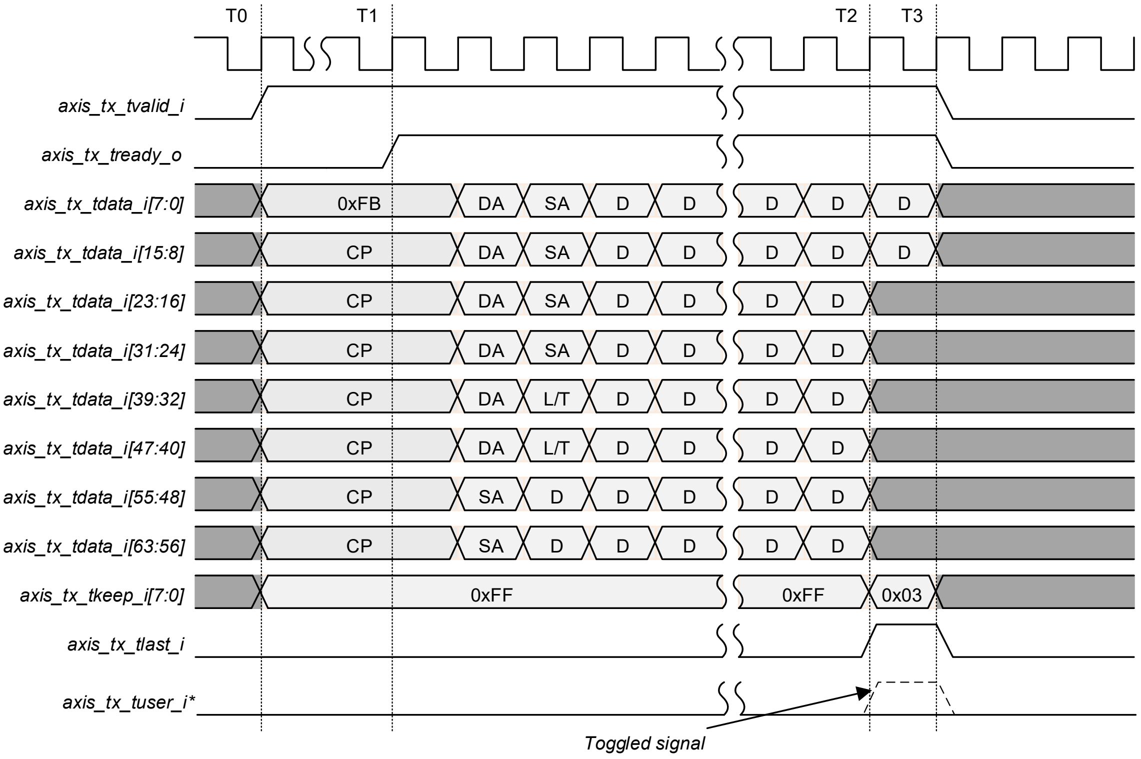


Figure 2.14. Transmission with Custom Preamble Passing

*** Note:** You can inject the error by asserting the axis_tx_tuser_i signal during EOP. Otherwise, the axis_tx_tuser_i signal must drive to low.

2.2.8. Data Packing

This module converts between XGMII, GMII, and MII interfaces. This is enabled when the *Dynamic Speed Selection* attribute is enabled or if *PHY interface* is set to any of the following: 8-bit GMII, 16-bit GMII, or MII.

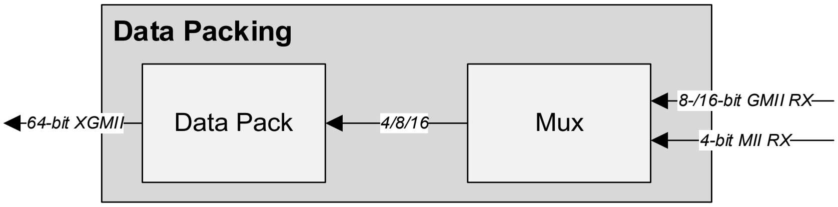


Figure 2.15. Receive Physical Interface for 8/6-bit GMII or MII

The receiver side takes the 8-bit/16-bit GMII or 4-bit MII and transforms it into XGMII packets. The transformation process involves detection of the assertion of the RX data valid signal (gmii_rx_dv_i or gmii_16_rx_dv_i or mii_rx_dv_i). It then replaces the first preamble symbol (0x55) to a start character (0xFB), pack the 4/8/16 bit data to a 64-bit data and store the reconstructed 64-bit data to a first-in first-out (FIFO) data flow before it is sent out to the MAC layer. The module also inserts an end-of-frame character (0xFD) when the data valid signal has deasserted.

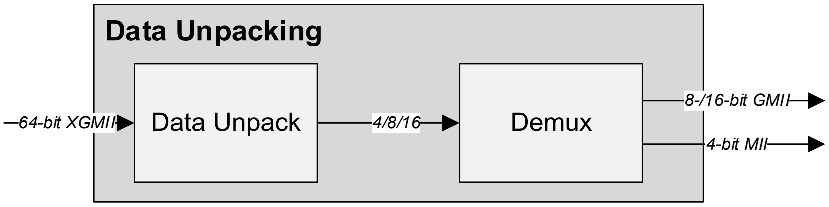


Figure 2.16. Transmit Physical Interface for 8/16-bit GMII or MII

The transmitter side transfers the 64-bit XGMII data to 8-bit/16-bit GMII or 4-bit MII data. When the module detects the start character (0xFB), it asserts the TX data valid signal (gmii_tx_en_o or gmii_16_tx_en_o or mii_tx_en_o) and replaces the start character (0xFB) to a preamble symbol (0x55). It then unpacks and stores the reconstructed 4/8/16-bit data to a FIFO before it is sent out the PHY layer. When it detects the end-of-frame character (0xFD), the module deasserts gmii_tx_en_o or gmii_16_tx_en_o or mii_tx_en_o signal and remove 0xFD character before transmitting the data.

The IFG is longer than expected for 8-bit/16-bit GMII or 4-bit MII mode because we are packing or unpacking from 64-bit data to 4/8/16-bit data and vice versa.

2.2.9. MAC Management Block

The MAC Management block is accessed through the APB interface. This block is responsible for the following:

- Configuration of the core
- Access to interrupt block
- Access to statistics counter

The various events that occur during the reception of a frame are also logged into the statistics vector signals (rx_statvec_o) and the TX_RX_STS register. At the end of reception, the rx_staten_o signal is asserted to qualify the rx_statvec_o signal. A vector is not generated for all those frames that are discarded (no address match or frame length is less than 64 bytes) or ignored (you assert the ignore_pkt of MAC_CTL register). For every frame transmitted, a statistics vector signals (tx_statvec_o) is generated, including all the statistical information collected in the process of transmitting the frame. Data on tx_statvec_o is qualified by assertion of the tx_staten_o signal. These statistics can also be accessed in the statistics counter when the *Statistics Counter Register* attribute is enabled.

For the timing details of the AMBA 3 APB protocol, refer to the AMBA 3 APB Protocol version 1.0 Specification.

2.3. PHY (Avant Devices)

2.3.1. IP Architecture Overview

The 10G Ethernet PHY IP core provides the XGMII interface to the MAC and follows the IEEE802.3 10GBASE-R standard. It supports 64 bits of data and 8 bits of control signals for both the transmit and receive paths.

This IP core instantiates the MPPHY foundation IP configured as 1-Lane 64b/66b PCS and a Management block that supports AXI4-Lite, or APB access to PCS registers. For more information on MPPHY foundation IP, refer to the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).

The functional description in this section is only applicable for 10G Ethernet PHY IP core on Avant-AT-G and Avant-AT-X devices (LAV-AT-G70/X70).

2.3.2. Block Diagram

The following figure shows the top-level block diagram of the 10G Ethernet PHY IP core.

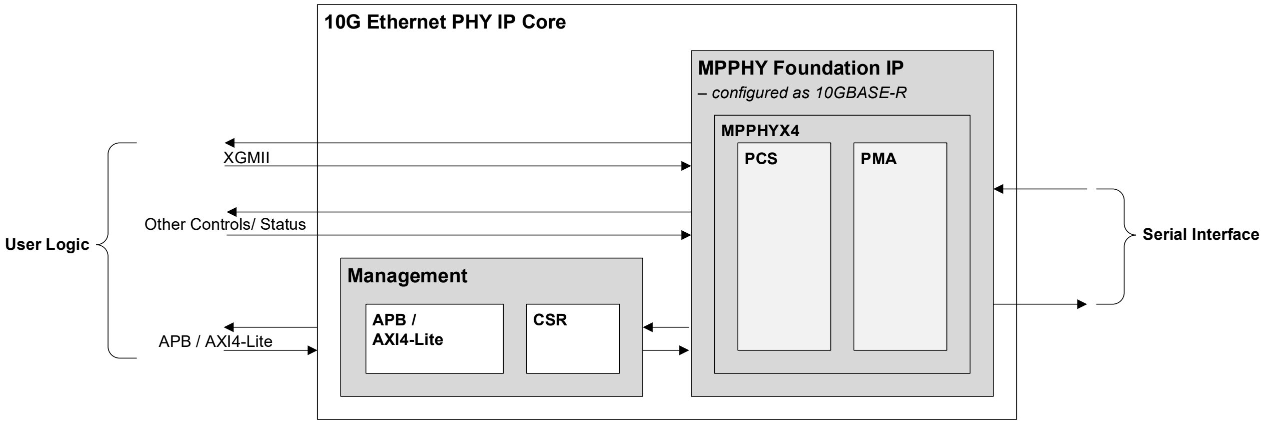


Figure 2.17. 10G Ethernet PHY IP Core Block Diagram

2.3.3. Lane Merging (Avant Devices)

2.3.3.1. Overview

The LAV-AT-G/X70 silicon can have up to seven usable quads while the LAV-AT-G/X30 silicon have up to three usable quads depending on the package. Each MPPHY block is a quad made up of four lanes (X4). Therefore, a single LAV-AT-G/X70 device could theoretically contain up to 28 lanes across seven quads.

You can configure the quad to use each lane individually, enabling you to merge multiple MPPHY instances into a single physical quad. This maximizes the usage of silicon resources in your design.

Note: Lane merging is only supported by LAV-AT-G70 and LAV-AT-X70 devices. For more information, refer to the Lattice Avant-G/X MPPHY Module User Guide (FPGA-IPUG-02233).

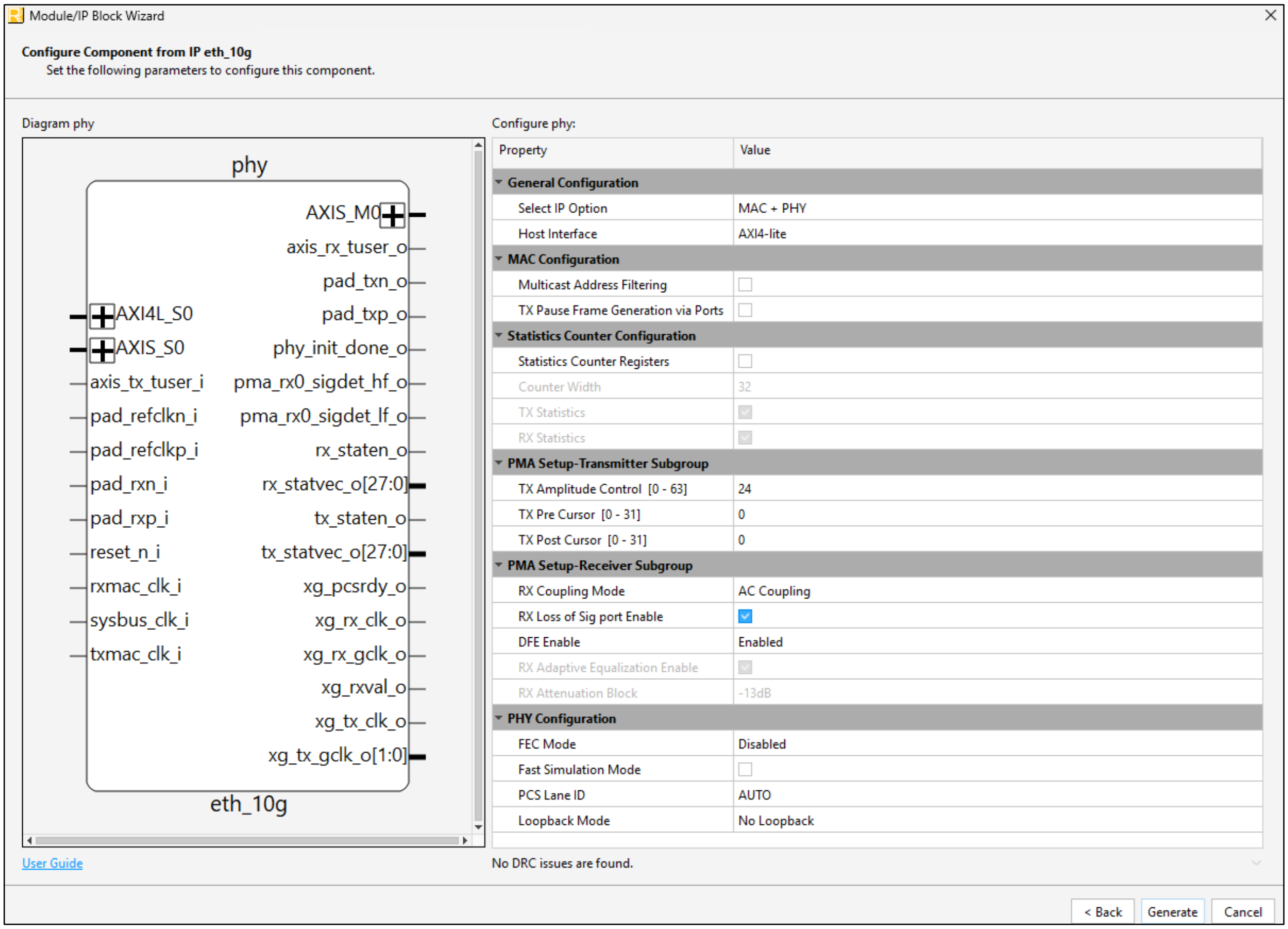
2.3.3.2. Usage

The 10G Ethernet PHY IP core instantiates the MPPHY foundation IP configured with a 1X1 link width, which occupies a single lane of a quad. By default, the Radiant software attempts to merge MPPHY instances to minimize the device power consumption.

If you want to override this behavior and select specific locations for the MPPHY instances, you can set the LANE ID configuration of the IP according to the number of quads you want to use.

To set the Lane ID, follow these steps:

1. In the Select IP Option field, select **PHY only** or **MAC + PHY**.
2. In the PCS Lane ID field, set the lane ID configuration of the IP as shown in the figure below. By default the ID is set to **AUTO**. If AUTO lane ID is selected, the pin location assignment must be included in the sdc or ldc constraint file. If the ID is not set to **AUTO**, in the case of a conflict between the Lane ID configuration and a top-level design port constraint, the top-level design port constraint takes precedence, which means the Lane ID setting is ignored and a warning message is shown.



Module/IP Block Wizard

Configure Component from IP eth_10g

Set the following parameters to configure this component.

Diagram phy

Configure phy:

Property	Value
General Configuration	
Select IP Option	MAC + PHY
Host Interface	AXI4-lite
MAC Configuration	
Multicast Address Filtering	☐
TX Pause Frame Generation via Ports	☐
Statistics Counter Configuration	
Statistics Counter Registers	☐
Counter Width	32
TX Statistics	☑
RX Statistics	☑
PMA Setup-Transmitter Subgroup	
TX Amplitude Control [0 - 63]	24
TX Pre Cursor [0 - 31]	0
TX Post Cursor [0 - 31]	0
PMA Setup-Receiver Subgroup	
RX Coupling Mode	AC Coupling
RX Loss of Sig port Enable	☑
DFE Enable	Enabled
RX Adaptive Equalization Enable	☑
RX Attenuation Block	-13dB
PHY Configuration	
FEC Mode	Disabled
Fast Simulation Mode	☐
PCS Lane ID	AUTO
Loopback Mode	No Loopback

User Guide

No DRC issues are found.

Figure 2.18. Set the Lane ID Configuration of the 10G Ethernet PHY IP Core

The Lane ID value is global for the entire device, starting at 0 from Lane 0 of the left-most quad, then incrementing up to 27 for Lane 3 of the right-most quad for the largest package of LAV-AT-G/X70. An example of Lane ID numbering is shown in the figure below.

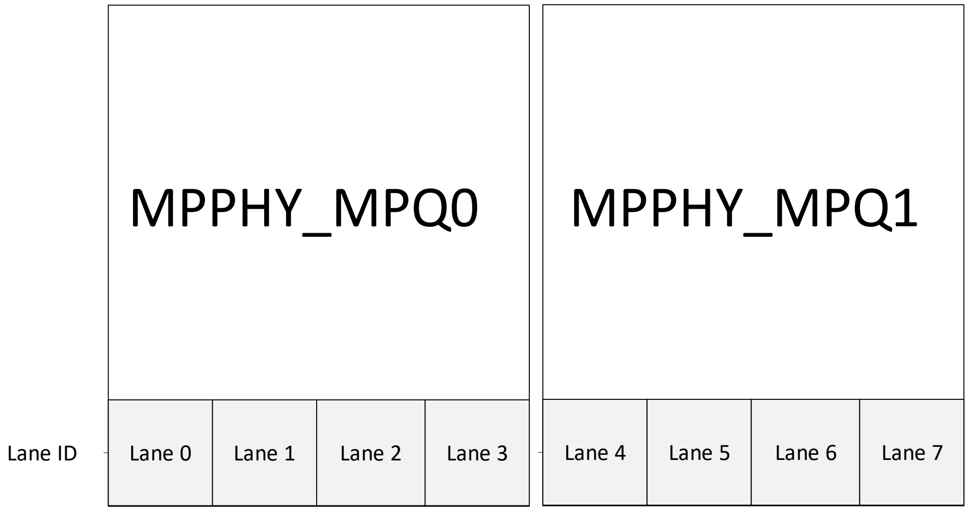
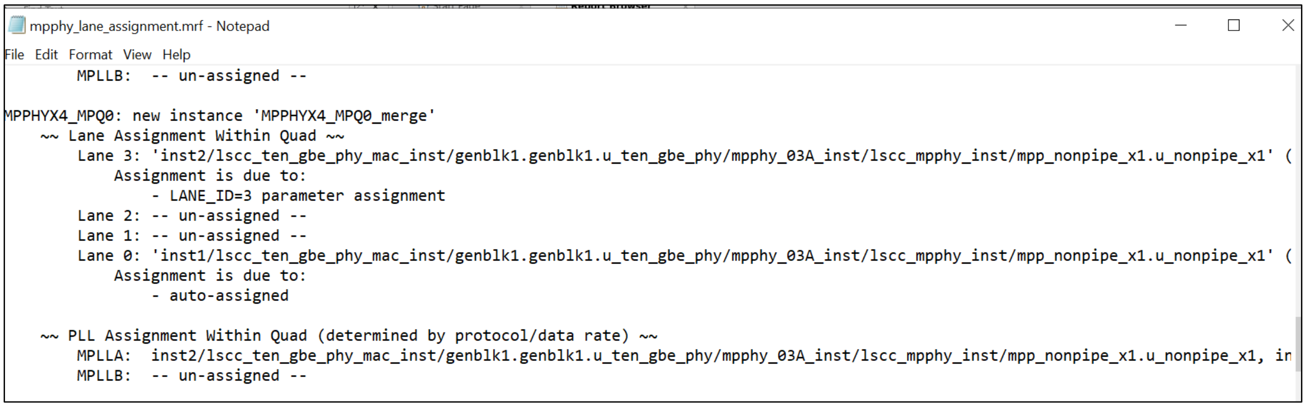


Figure 2.19. Lane ID Numbering

2.3.3.3. Lane Merging Report

During lane merging, the Post-Synthesis process produces a report file named *mpphy_lane_assignment.mrf* in the project's active implementation folder that shows how the MPPHY design instances were merged into the device MPPHY quads. You can open the report file with any text editor.


mpphy_lane_assignment.mrf - Notepad
File Edit Format View Help

```
       MPLLB:  -- un-assigned --

MPPHYX4_MPQ0: new instance 'MPPHYX4_MPQ0_merge'
    ~~ Lane Assignment Within Quad ~~
        Lane 3: 'inst2/lscc_ten_gbe_phy_mac_inst/genblk1.genblk1.u_ten_gbe_phy/mpphy_03A_inst/lscc_mpphy_inst/mpp_nonpipe_x1.u_nonpipe_x1' (
            Assignment is due to:
                - LANE_ID=3 parameter assignment
        Lane 2: -- un-assigned --
        Lane 1: -- un-assigned --
        Lane 0: 'inst1/lscc_ten_gbe_phy_mac_inst/genblk1.genblk1.u_ten_gbe_phy/mpphy_03A_inst/lscc_mpphy_inst/mpp_nonpipe_x1.u_nonpipe_x1' (
            Assignment is due to:
                - auto-assigned

    ~~ PLL Assignment Within Quad (determined by protocol/data rate) ~~
        MPLLA:  inst2/lscc_ten_gbe_phy_mac_inst/genblk1.genblk1.u_ten_gbe_phy/mpphy_03A_inst/lscc_mpphy_inst/mpp_nonpipe_x1.u_nonpipe_x1, in
        MPLLB:  -- un-assigned --
```

Figure 2.20. Lane Merging Report File

2.3.3.4. Restrictions and Limitations

To merge instances into the same quad, instances must abide by the following restrictions:

- Shared reference clock connection.
- Shared LMMI clock and reset connections.
- Compatible PLL settings (this is protocol/data rate dependent, and will be configured by the IP Catalog tool)
- Compatible "per-quad" connections – a limited number of ports exposed on the MPPHY IP physically have only a single instance on the silicon. These must be connected to the same net if it is an input, or have a maximum of one connected per-quad if it is an output.
- All input must be driven by the same source. For output clock, use only the output clock from one instance of the quad to drive the logic. For output clocks from other IP instance that is intended to be merged into the same quad, do not use the clock to drive any logic. Refer to the table below.

Table 2.1. Shared Signal Mapping of 10G Ethernet IP to MPPHY for Lane Merging

MPPHY	10G Ethernet IP	Direction
lmmiclk_q0_i	sysbus_clk_i	Input
refclk_p_q0_i refclk_n_q0_i	pad_refclkp_i pad_refclkn_i	Input
lmmireset_n_q0_i	reset_n_i	Input
txoutgclk_pll0_q0_o	xg_tx_gclk_o[0]	Output
txoutgclk_pll1_q0_o	xg_tx_gclk_o[1]	Output

If any of these restrictions are violated, the Radiant software will not automatically merge the MPPHY instances into a single quad. If user constraints or lane assignment forces incompatible MPPHY instances into the same quad, an error message is issued, and the Radiant software flow will not continue past the Post-Synthesis stage.

2.3.4. PHY Management Block

The PHY Management block is accessed through the AXI4-Lite or APB interface. This block is responsible for register access of the PCS registers. While all the two interfaces (AXI4-Lite or APB) are available for PCS Module Registers

For PCS register access through AXI4-Lite, this soft IP requires the use of remapped addresses (0xA000 – 0xA0FC) as AXI4-Lite addresses must be DWORD-aligned (0xA000, 0xA004, 0xA008, 0xA00C, and so on). To get the corresponding AXI4-Lite addresses, you will need to left shift even-numbered PCS address once (or multiply by two) – the lower 2 bytes of the AXI4-Lite read/write data will be mapped to the even-numbered PCS register; while the upper 2 bytes of the AXI4-Lite read/write data will be mapped to the subsequent odd-numbered PCS register.

For details and illustrations on how the AXI4-Lite address remapping works, refer to Table 2.2 and Figure 2.21.

Note: The read/write data values used in the following illustrations are just examples and are not the suggested register values to be written. Because PCS registers are half-duplex, simultaneous write to/read from PCS registers through AXI4-Lite channels are not supported.

With APB interfaces, no address remapping is required. The actual PCS Module Registers addresses (0x5000 – 0x507F) are used.

Table 2.2. AXI4-Lite to PCS Address and Data Conversion

axi4l_awaddr_i[31:0] / axi4l_araddr_i[31:0]	axi4l_wdata_i[31:0] / axi4l_rdata_o[31:0]	PCS Register Address (16-bit)	Access Types	PCS Register Values (16-bit)
0xA000	0x22334455	0x5000	RW	0x4455 = axi4l_wdata_i[15:0]
		0x5001	—	Not available
0xA004	0xaabbccdd	0x5002	RW	0xccdd = axi4l_wdata_i[15:0]
		0x5003	RW	0xaabb = axi4l_wdata_i[31:16]
0xA008	0x11335577	0x5004	RW	0x5577 = axi4l_wdata_i[15:0]
		0x5005	RW	0x1133 = axi4l_wdata_i[31:16]
0xA0FC	0x8899eeff	0x507E	RO	Read-only register
		0x507F	RW	0x8899 = axi4l_wdata_i[31:16]

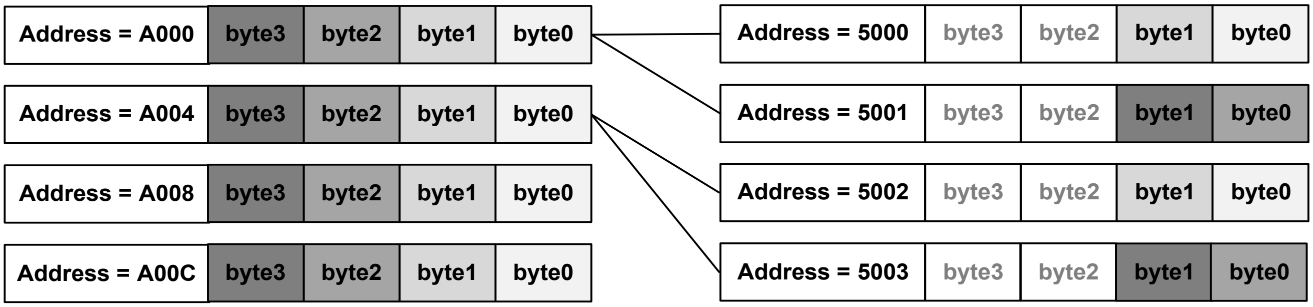


Figure 2.21. Remapped Addresses for PCS Register Address through the AXI4-Lite Interface

2.3.5. Loopback Modes

2.3.5.1. Far End Parallel Loopback

The input signal is driven at the serial RX port (RX PMA input) and the output is observed at the serial TX port (TX PMA output). In this loopback, user logic is not involved.

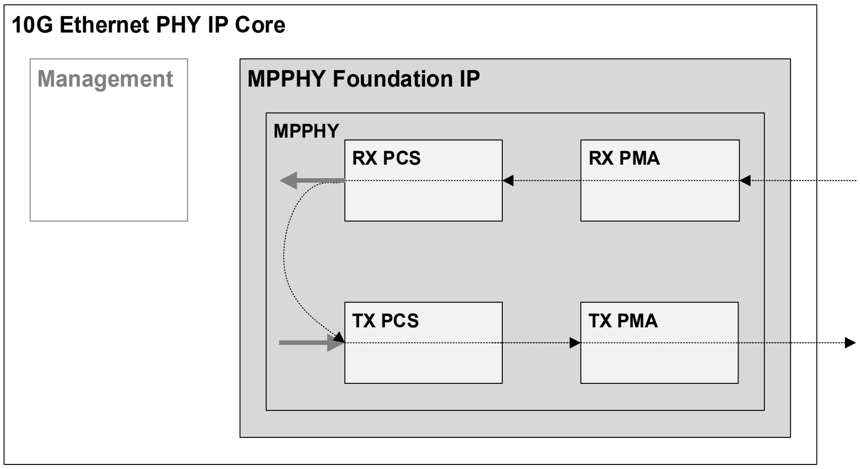


Figure 2.22. Far End Parallel Loopback

2.3.5.2. Near End Parallel Loopback

The input signal is driven at the parallel TX port (user logic side) and the output signal is observed at the parallel RX port (user logic side). In this loopback, PMA is not involved.

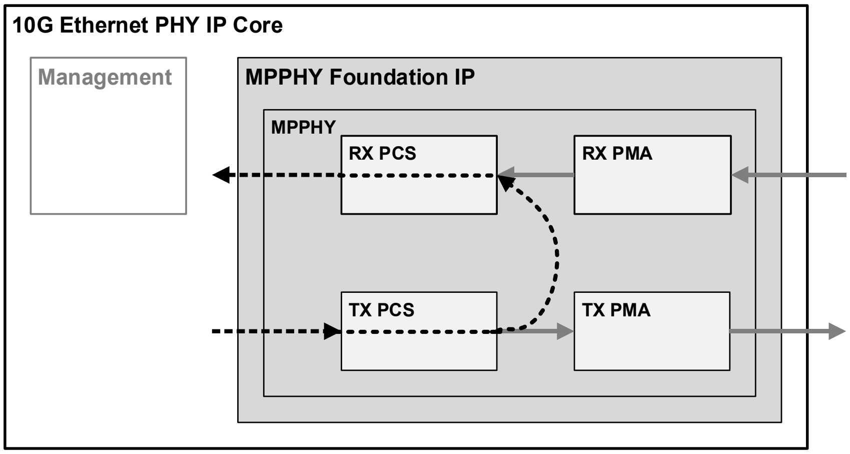


Figure 2.23. Near End Parallel Loopback

2.4. PHY (CertusPro-NX Devices)

2.4.1. IP Architecture Overview

The CertusPro-NX 10G Ethernet PHY IP core provides the XGMII interface to the MAC and follows the IEEE802.3 10GBASE-R standard. It supports 64-bit of data and 8-bit of control signals for both transmit and receive path. Due to hardware limitation, CertusPro-NX 10G Ethernet PHY IP core can only be placed on the following lanes:

- LFCPNX-100 supported lanes: Lane 2, Lane 3, Lane 6 and Lane 7 only.
- LFCPNX-50 supported lanes: Lane 2 and Lane 3 only.

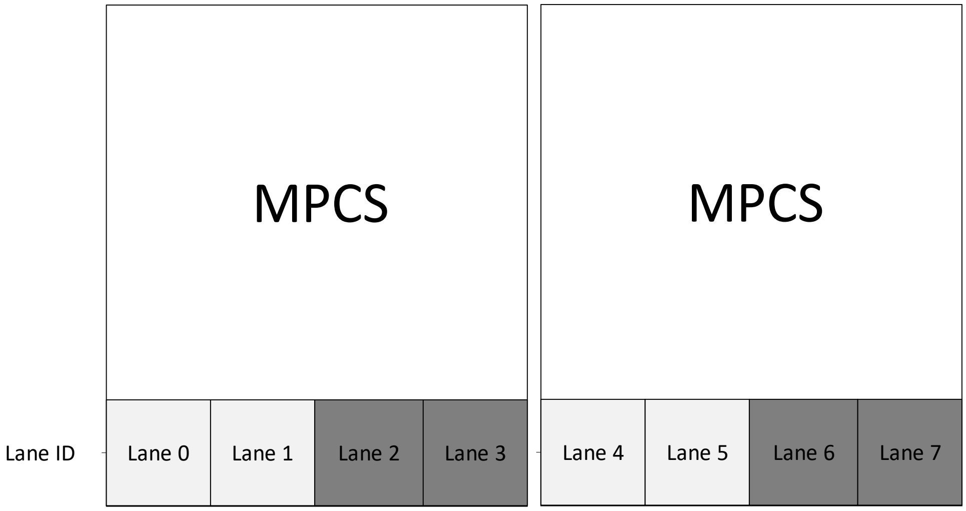


Figure 2.24. CertusPro-NX 10G Ethernet PHY IP Core Hardware Supported Lane

This IP core instantiates the MPCS Module foundation IP configured as 1-Lane 64b/66b PCS, and supports APB access to MPCS registers. For more information on MPCS Module foundation IP, refer to the MPCS Module User Guide (FPGA-IPUG-02118).

2.4.2. Block Diagram

The following figure shows the top-level block diagram of the 10G Ethernet PHY IP core.

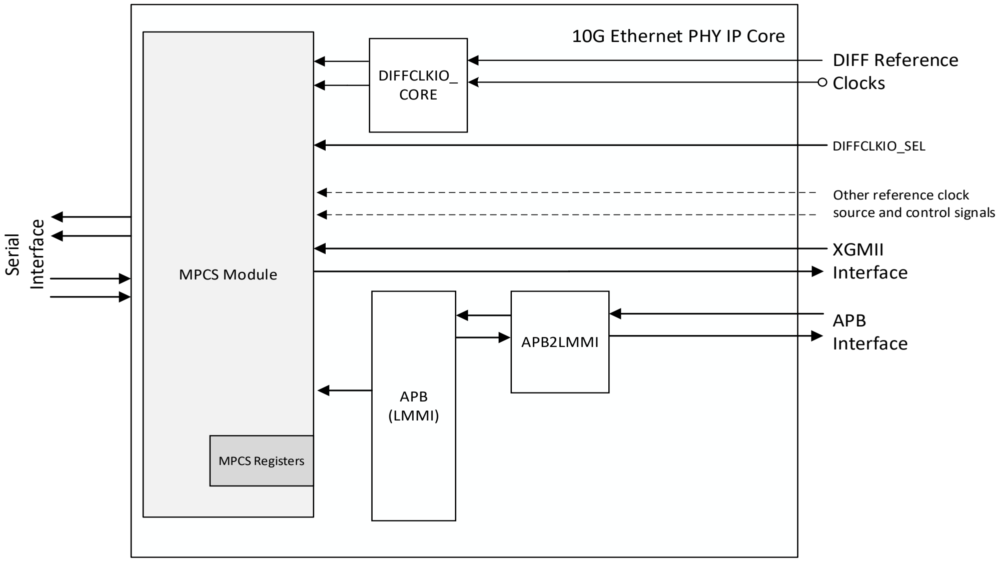
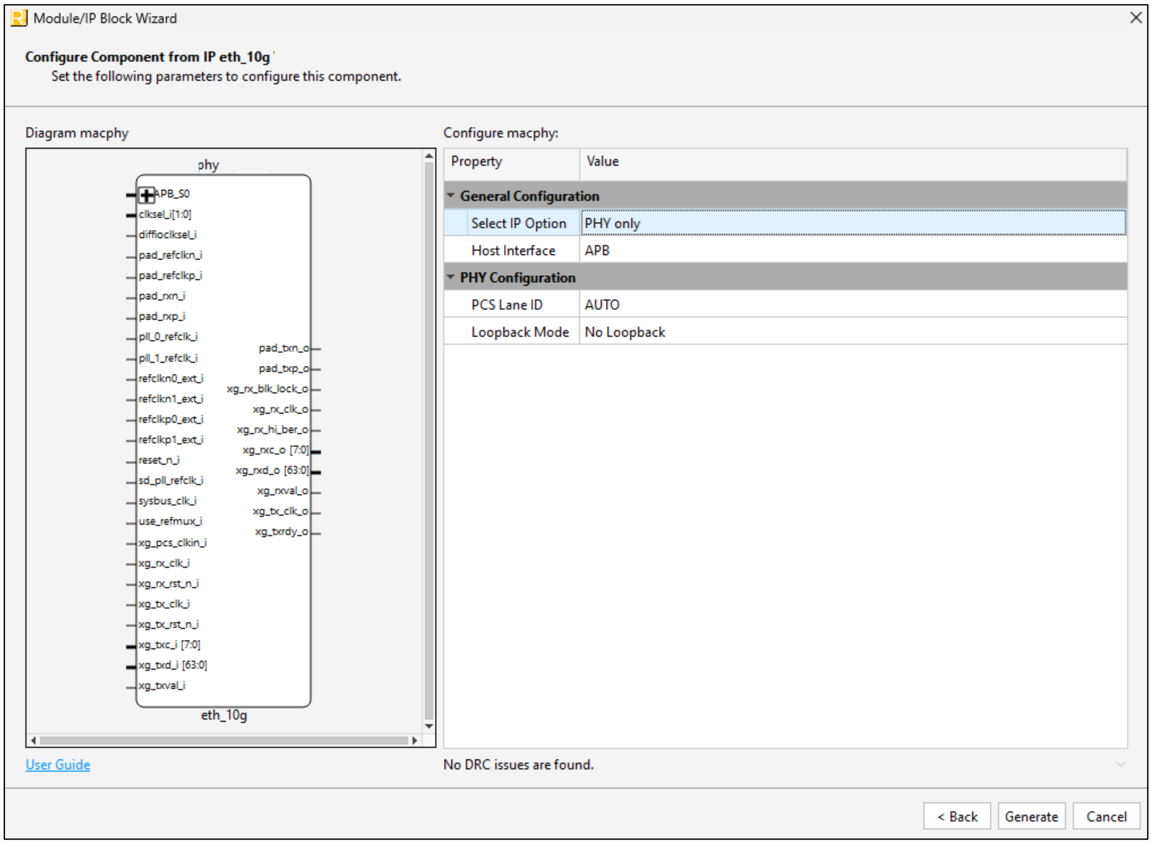


Figure 2.25. CertusPro-NX 10G Ethernet PHY IP Top-Level Block Diagram

2.4.3. Lane Merging (CertusPro-NX Devices)

Set the LANE ID configuration of the IP according to the number of quad you want to use. Follow these steps:

1. In the **Select IP Option** field, select **PHY only** or **MAC + PHY** or **MAC + PHY + 1588**.
2. In the **PCS Lane ID** field, set the lane ID configuration of the IP as shown in the figure below. By default, the ID is set to **AUTO**. For more details, refer to the IP Architecture Overview section on the supported lane.



Module/IP Block Wizard

Configure Component from IP eth_10g

Set the following parameters to configure this component.

Diagram macphy

Configure macphy:

Property	Value
General Configuration	
Select IP Option	PHY only
Host Interface	APB
PHY Configuration	
PCS Lane ID	AUTO
Loopback Mode	No Loopback

No DRC issues are found.

Figure 2.26. Lane ID Configuration

2.4.4. Reference Clocks

The CertusPro-NX 10G Ethernet PHY IP core provides other sources of reference clocks and can be used for dynamic switching. The pll_0/1_refclk_i must only be connected to PLL and must not be directly connected to the fabric. To use this feature, you must set the reference clock source control signals according to the following table.

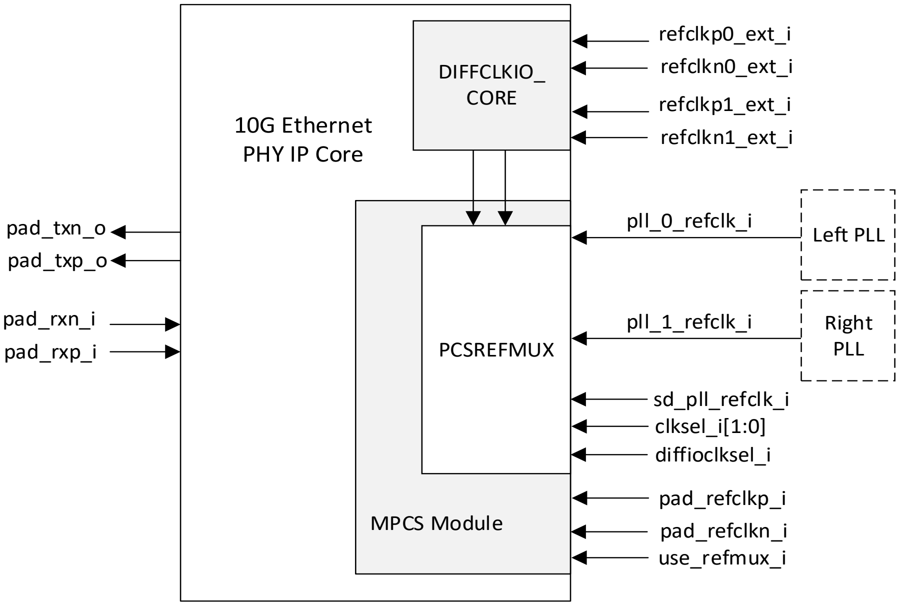


Figure 2.27. Reference Clock Block Diagram

Table 2.3. Reference Clock MUX Tree Control Signals

Reference Clock Control Signal	Usage
use_refmux_i	1'b1 – Use reference clock output from Clock MUX Tree (PCSREFMUX). 1'b0 – Use dedicated reference clocks (pad_refclkp/n_i).
diffioclksel_i	1'b1 – Use refclkp/n1_ext_i as reference clocks. 1'b0 – Use refclkp/n0_ext_i as reference clocks.
clksel_i	2'b00 – Use pll_0_refclk_i as reference clocks. 2'b01 – Use pll_1_refclk_i as reference clocks. 2'b10 – Use reference clock based on diffioclksel_i. 2'b11 – Use sd_pll_refclk_i as reference clock.

2.4.5. PCS Loopback

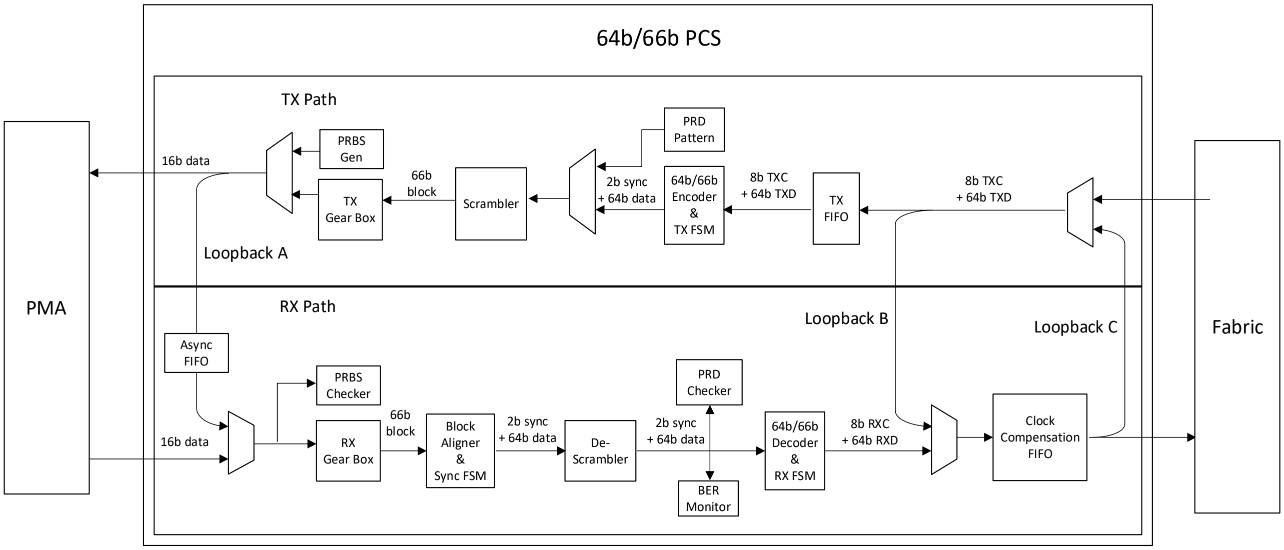


Figure 2.28. 64b66b PCS Loopback Diagram

The following lists the various types of PCS Loopback:

- Loopback A

  This loopback can be enabled when *Loopback* = *Near End Parallel Loopback* or by using bit 0 of the Loopback Mode Control (MPCS register).

  In this mode, the 16-bit input data of the RX path comes from the TX path. The TX path clock will drive both the TX and RX path. The asynchronous FIFO in between the TX path and RX path is for clock phase compensation.
- Loopback B

  This loopback can be enabled using bit 2 of the Loopback Mode Control (MPCS register).

  In this mode, the PCS accepts data on the transmit path from the XGMII and returns it on the receive path to the XGMII through the RX FIFO.
- Loopback C

  This loopback can be enabled when *Loopback* = *Far End Parallel Loopback* or by using bit 1 of the Loopback Mode Control (MPCS register).

  In this mode, the received data by the RX PCS is returned to the TX PCS.

2.5. MAC + PHY + 1588 (CertusPro-NX Devices)

2.5.1. Design Architecture

The timestamp unit (TSU) block is located between the 10G Ethernet MAC IP and the 10G Ethernet PHY IP block. The TSU will be enabled with MAC + PHY. This offering is only available for CertusPro-NX devices. The following blocks are present:

- TSU
- 10G Ethernet MAC IP cores. For more details, refer to the MAC section.
- 10G Ethernet PCS/PMA IP cores (CertusPro-NX PHY). For more details, refer to the PHY (CertusPro-NX Devices) section.
  For Lane Merging details, refer to the Lane Merging (CertusPro-NX Devices) section.

2.5.1.1. PTP Synchronization

The following diagram shows the 1588 PTP procedure to synchronize the clock.

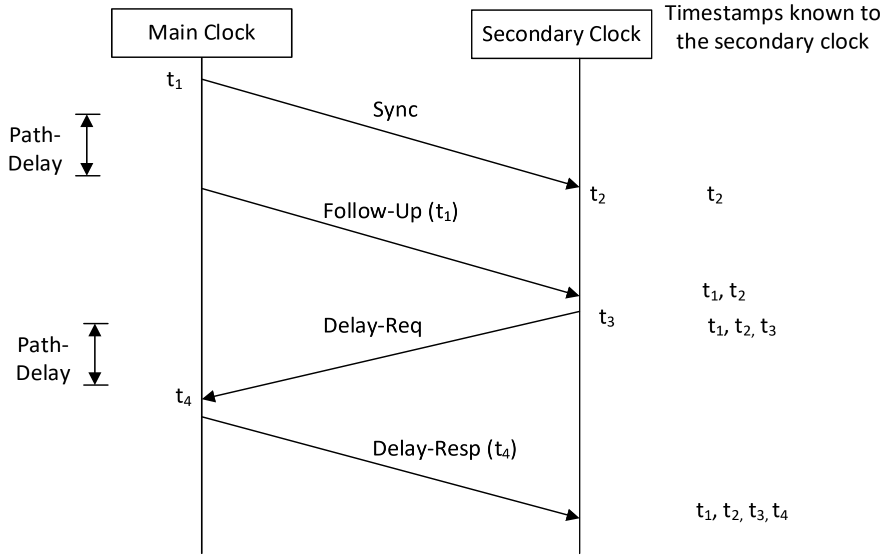


Figure 2.29. PTP Time Synchronization Flow

The following describes the procedure to synchronize the clock:

1. When the main clock sends a Sync message, the following steps are carried out:
   a. In 1-step synchronization, t_1 is sent in the Sync message.
   b. In 2-step synchronization, t_1 is sent in the Follow-Up and Sync messages.
2. The secondary clock receives a Sync message at timestamp t_2.
3. The secondary clock sends a Delay-Req message at t_3.
4. The main clock receives a Delay-Req message at t_4.
5. The main clock responds with a Delay-Resp message at t_4.
6. The secondary clock needs t_1, t_2, t_3, and t_4 timestamps for Delay estimation.

$$\text{Mean Path Delay} = ((t_2-t_1) + (t_4-t_3))/2$$

$$\text{Offset} = t_2-t_1\text{-Mean Path Delay}$$

2.5.2. Block Diagram

The following blocks are present in this design:

- 10G Ethernet PCS/PMA IP cores
- TSU
- 10G Ethernet MAC IP cores

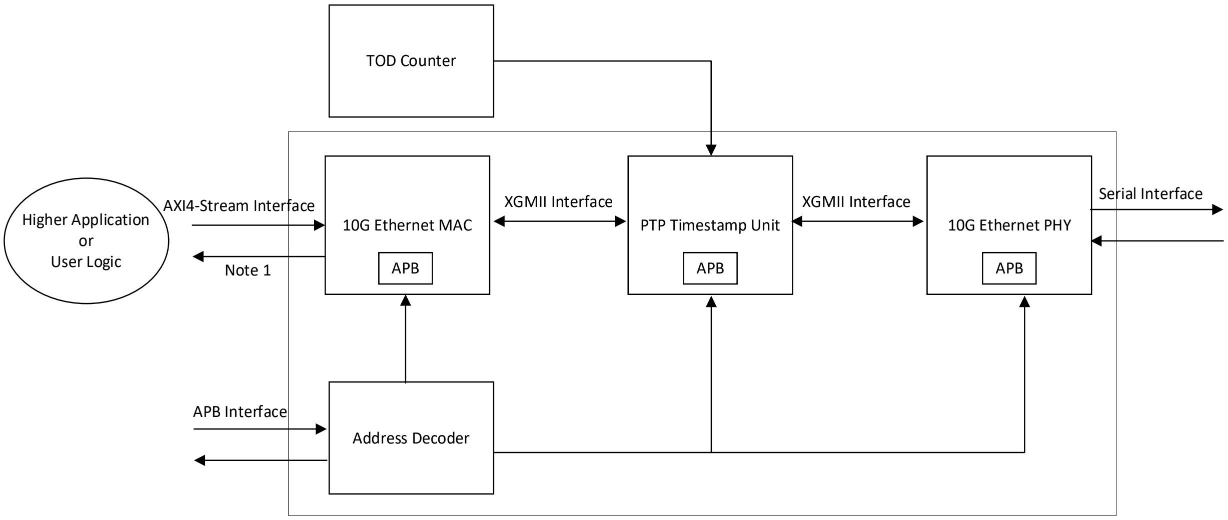


Figure 2.30. 10GbE MAC + PHY + 1588 IP Core Block Diagram

2.5.3. Timestamp Unit

The key features of this block are as follows:

- Detection of PTP and non PTP packets
- PTP packet parsing
- Timestamping for all PTP packets in the TX path.
- Timestamping for all PTP packets in the RX path.
- Correction field calculation for end-to-end transparent clock (E2E TC) (TC[2])
- Timestamp insertion for PTP Event Messages in 1-step[2] mode
- UDP Checksum and CRC32 recalculation for 1 step[2] PTP messages
- All the PTP messages remain unchanged for 2-step PTP.

Notes:

1. At the RX side, the MAC AXIS output contains PTP and non-PTP packets. Therefore, muxing logic for PTP and non-PTP must be handled by user logic if required. The ToD counter is done in user logic as well.
2. The PTP 1588v2 supports 2-step clock synchronization. Note that 1-step synchronization will be enabled in later milestone. The TC mode must be enabled with 1-step mode. In 2-step synchronization, the TSU function provides accurate timestamp and the related fingerprint for all PTP messages without modifying the packet content.

2.5.4. 10G Ethernet PHY

For more information on 10G Ethernet PHY, refer to the PHY (CertusPro-NX Devices) section.

2.5.5. 10G Ethernet MAC

For more information on 10G Ethernet MAC, refer to the MAC section.

2.5.6. PTP Packet Format

This section describes the following PTP packet formats:

- PTP over Ethernet
- PTP over UDP/IPv4
- PTP over UDP/IPv6

2.5.6.1. PTP Over Ethernet

The following section describes the location of the PTP message in an Ethernet frame.

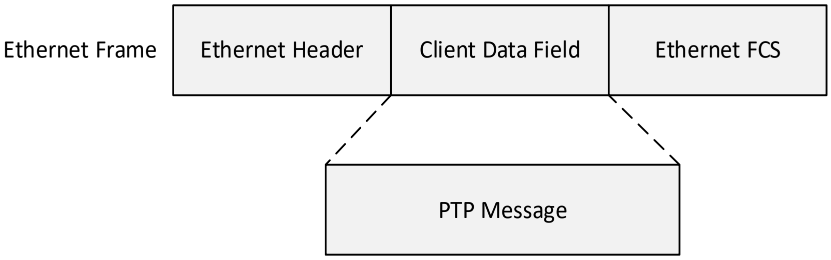


Figure 2.31. PTP Over Ethernet

Number of Bytes	7	1	6	6	2	44	0-1500	0-46	4
	Preamble	**Start of Frame Delimiter (SFD)**	**Destination Address (DA)**	**Source Address (SA)**	**Length/ Type (L/T)**	**PTP Header**	**Data**	**Pad**	**Frame Check Sequence (FCS)**

Figure 2.32. PTP Over Ethernet Frame Format

2.5.6.2. PTP over UDP/IPv4

The following section describes the location of the PTP message in the UDP/IPv4 packets.

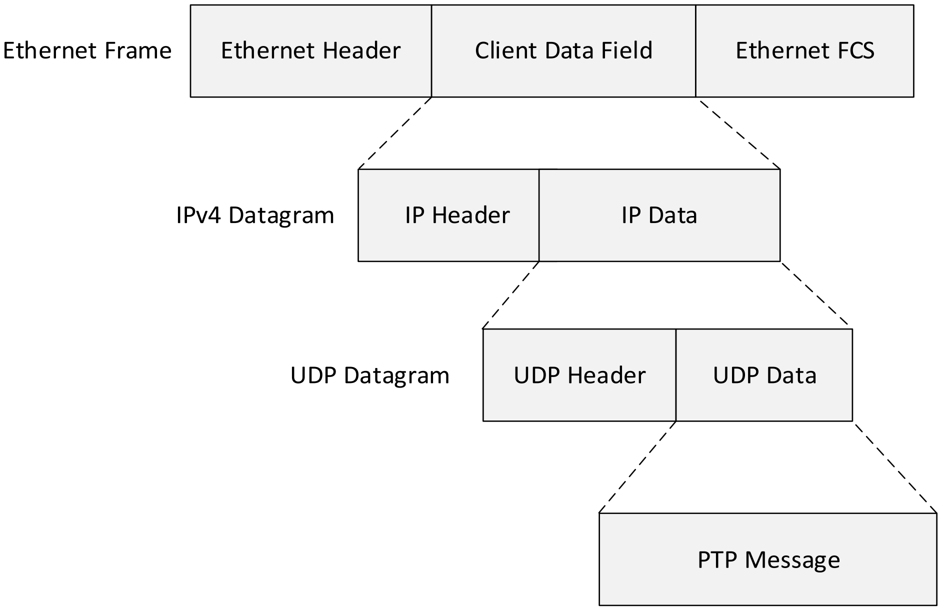


Figure 2.33. PTP Over UDP/IPV4

2.5.6.3. PTP Over UDP/IPv6

The following section describes the location of the PTP message in the UDP/IPv6 packets.

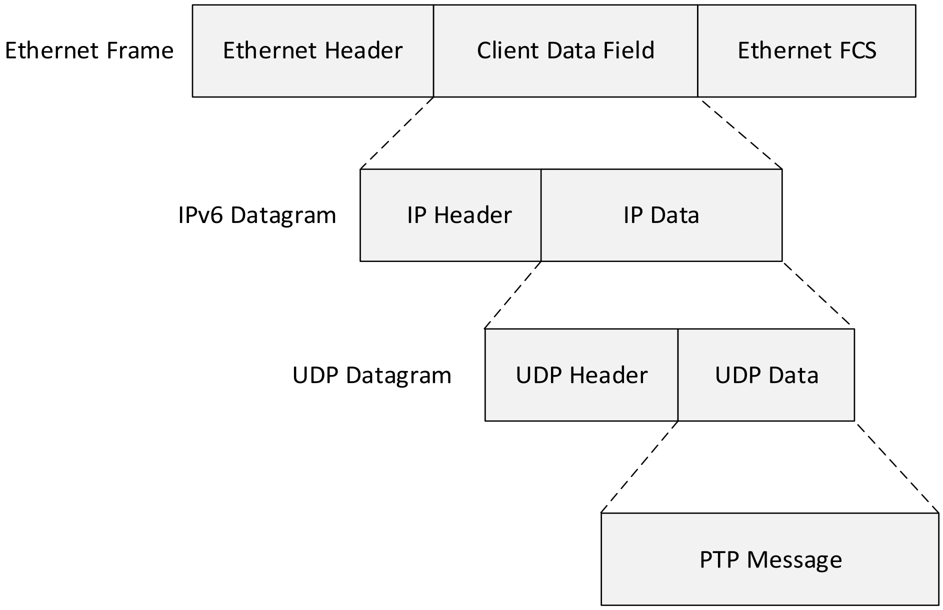


Figure 2.34. PTP Over UDP/IPv6

2.5.7. PTP Message Format

This section describes the PTP message formats. A PTP message consist of the following:

- 34-byte header
- Body
- Suffix (Optional)

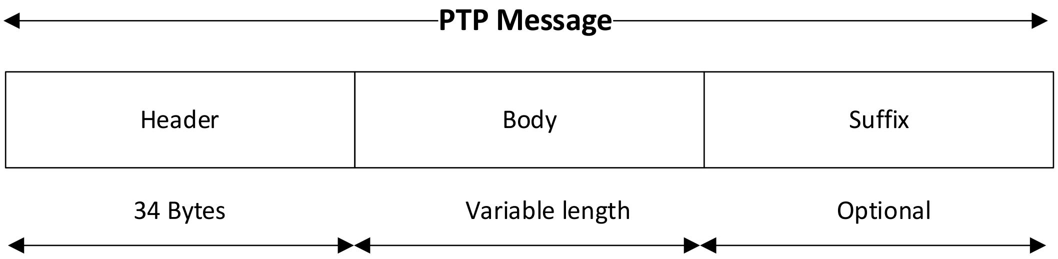


Figure 2.35. PTP Message Format

2.5.7.1. PTP Header

The following table lists the PTP Header message format.

Table 2.4. PTP Header

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
majorSdoId				messageType				1	0
minorVersionPTP				versionPTP				1	1
messageLength								2	2
domainNumber								1	4

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
minorSdoId								1	5
flagField								2	6
correctionField								8	8
messageTypeSpecific								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField								1	32
logMessageInterval								1	33

2.5.7.2. PTP Body

The PTP Body message format varies for the following PTP message types:

- PTP Sync Message
- PTP Follow-Up Message
- PTP Delay Request Message
- PTP Delay Response Message

PTP Body Format for Sync Message

The following table lists the PTP Body format for the Sync message type.

Table 2.5. PTP Body Format for Sync Message Type

Sync Message Format									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (Refer to Table 2.4. PTP Header)								34	0
originTimestamp								10	34

PTP Body Format for Follow-Up Message

The following table lists the PTP Body format for the Follow-Up message type.

Table 2.6. PTP Body Follow-Up Message Type

Follow_Up Message Format									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (Refer to Table 2.4. PTP Header)								34	0
preciseOriginTimestamp								10	34

PTP Body Format for Delay Request Message

The following table lists the PTP Body format for the Delay Request message type.

Table 2.7. PTP Body Format for the Delay Request Message Type

Delay_Req Message Format									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (Refer to Table 2.4. PTP Header)								34	0
originTimestamp								10	34

PTP Body Format for Delay Response Message

The following table lists the PTP Body format for the Delay Response message type.

Table 2.8. PTP Body Format for the Delay Response Message Type

Delay_Resp Message Format									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header (Refer to Table 2.4. PTP Header)								34	0
receiveTimestamp								10	34
requestingPortIdentity								10	44

2.5.8. PHY Delay Value

The following table lists the CertusPro-NX PHY (PCS - PMA) delay values. You must program the TX PHY delay value in the Egress Delay Register (0x18) and PHY RX delay value in the Ingress Delay Register (0x1C).

Table 2.9. PHY Delay Values

Value or Path	RX	TX
Delay	140 ns + (64B/66B PCS Block Align Shift[1] x 0.097) +/- 4 ns	94 ns +/- 4 ns[2]

Notes:

1. To obtain the value for the 64B/66B PCS Block Align Shift register, refer to the *Summary of MPCS Registers* table in the MPCS Module User Guide (FPGA-IPUG-02118).
2. Preliminary rounded value.

2.6. MAC + PHY (CertusPro-NX Devices)

2.6.1. Design Architecture

The following blocks are present:

- 10G Ethernet MAC IP cores. For more details, refer to the MAC section.
- 10G Ethernet PCS/PMA IP cores (CertusPro-NX PHY). For more details, refer to the PHY (CertusPro-NX Devices) section. For Lane Merging details, refer to the Lane Merging (CertusPro-NX Devices) section.

2.6.2. Block Diagram

The following blocks are present in this design:

- 10G Ethernet PCS/PMA IP cores
- 10G Ethernet MAC IP cores

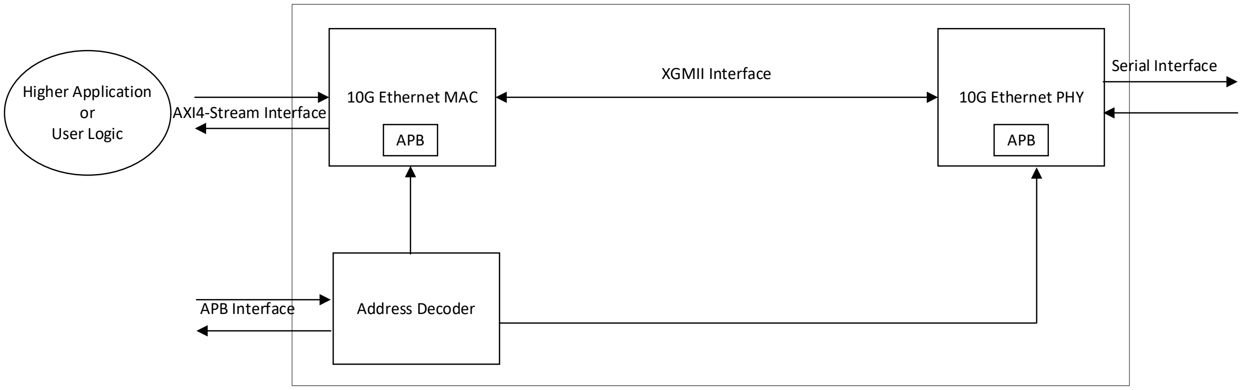


Figure 2.36. 10GbE MAC + PHY IP Core Block Diagram

2.7. Clocking

2.7.1. Clocking Overview

The following figure shows the clock network of the 10G Ethernet IP core for Avant devices. The clock frequency requirements are described in the Signal Description section according to the configuration selected—MAC only, PHY only, or MAC+PHY. In the MAC + PHY configuration, you must connect the txmac_clk_i and rxmac_clk_i output clock from the PHY block.

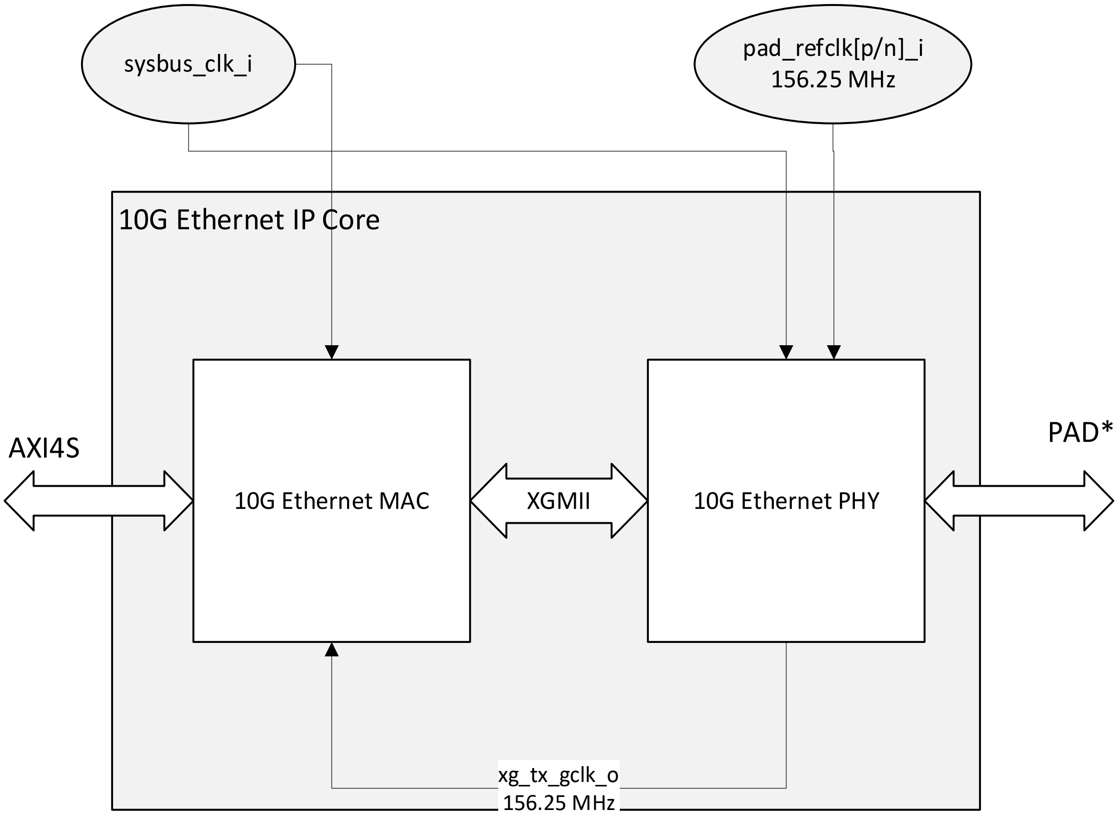


Figure 2.37. 10G Clock Network Diagram for Avant Devices

The following figure shows the clocking diagram for CertusPro-NX devices. The clock frequency requirements are described in the Signal Description section according to the configuration selected—MAC only, PHY only, or MAC+PHY+1588 (with TSU). In the MAC + PHY + 1588 configuration, the txmac_clk_i and rxmac_clk_i are connected internally to the output clock from the PHY block.

Note: Make sure that the GPLL being used does not have Fractional-N Divider mode enabled.

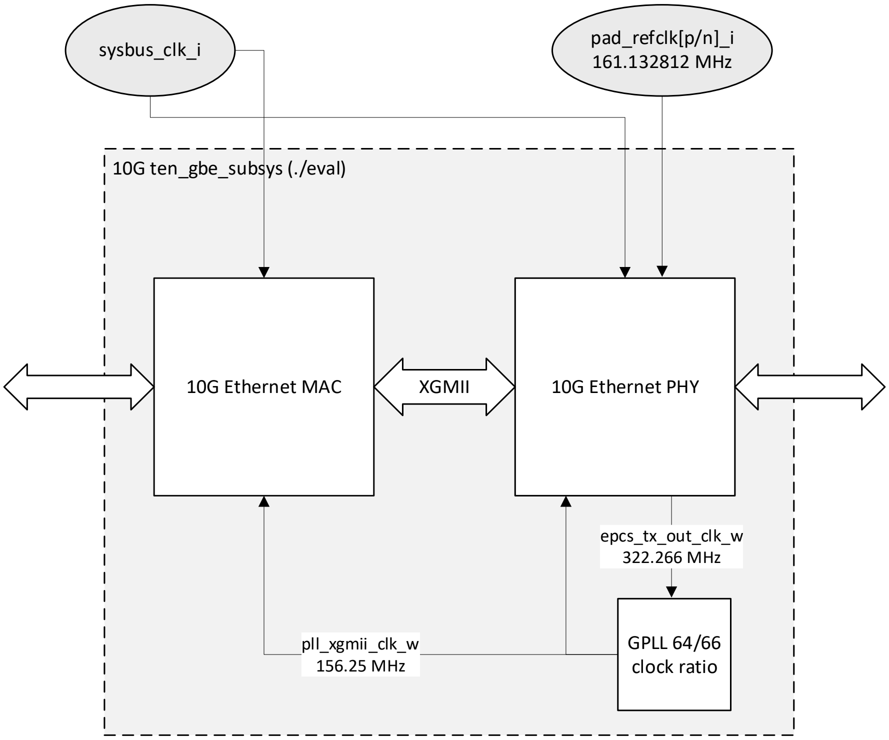


Figure 2.38. 10G Clock Network Diagram for CertusPro-NX Devices

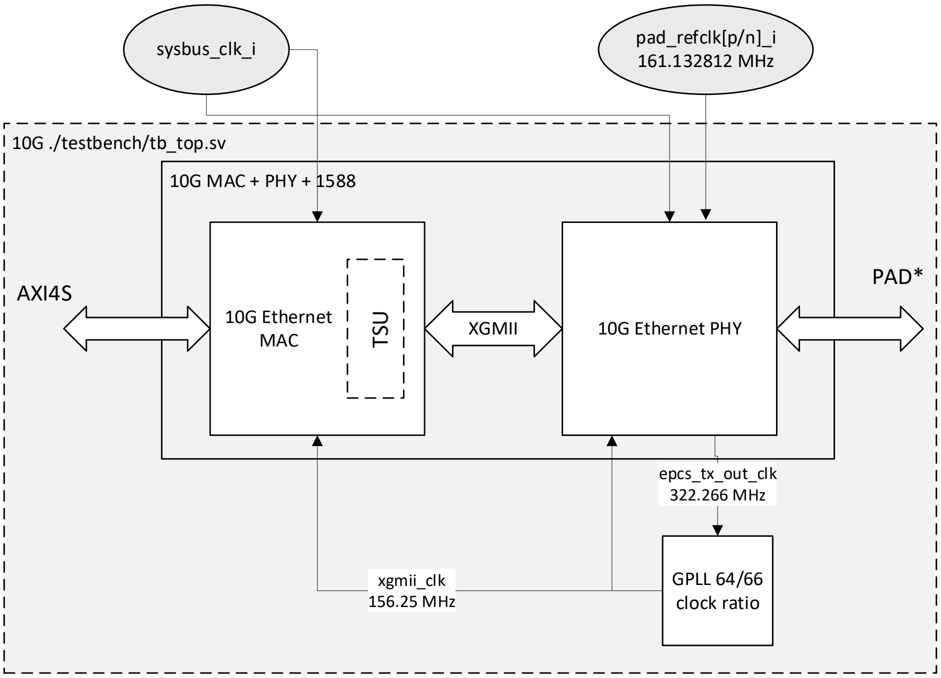


Figure 2.39. 10G Clock Network Diagram for CertusPro-NX Devices for MAC + PHY + 1588 Option

2.8. Reset

2.8.1. MAC Reset Sequence

2.8.1.1. Reset

An asynchronous reset pin (active low) as system reset is used for resetting the 10G Ethernet IP core. Internal reset logic is implemented to guarantee synchronous deassertion all throughout the different clock domain among soft logic blocks. The minimum assertion of reset is five clock cycles of the slowest clock*.

2.8.1.2. Power Up Sequence

The following lists the sequence after power-up of the chip or system:

1. Assert reset pin for five clock cycles of the slowest clock* of the system.
2. Wait for the 10G PHY to be in a ready state.
   For more information on the sequence, see the PHY (Avant Devices) section and the PHY (CertusPro-NX Devices) section.
3. Send settings through the AXI4-Lite interface to configure the system. For a list of MAC registers, refer to the Configuration Registers for MAC section.
4. The MAC is ready to receive transfers.

* **Note**: The slowest clock refers to the sysbus_clk_i with frequency that can be as low as 20 MHz.

The following figure shows a sample sequence when the RX MAC is configured to pass the FCS field to the client.

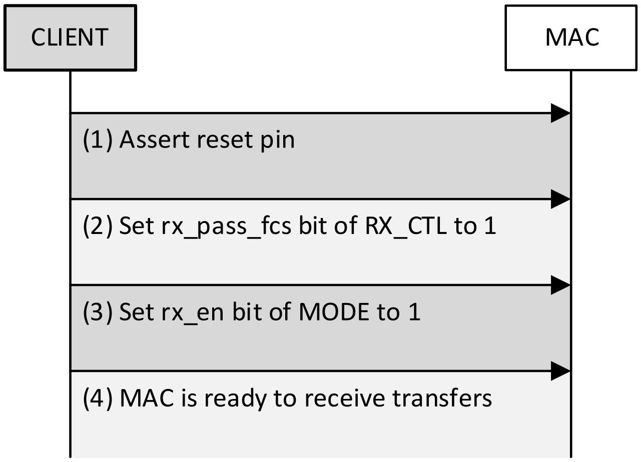

CLIENT
MAC
(1) Assert reset pin
(2) Set rx_pass_fcs bit of RX_CTL to 1
(3) Set rx_en bit of MODE to 1
(4) MAC is ready to receive transfers

Figure 2.40. Sequence to Configure RX MAC In-Band FCS Passing

2.8.2. PHY Reset Sequence (Avant Devices)

2.8.2.1. Reset

An asynchronous reset pin (active low) as system reset is used for resetting the 10GbE PHY IP core. Internal reset logic is implemented to guarantee synchronous deassertion all throughout the different clock domain among soft logic blocks. The minimum assertion of reset is five clock cycles of the slowest clock*.

2.8.2.2. Sequence

The following lists the sequence after power-up of the chip or system:

1. Assert reset pin, reset_n_i, for five clock cycles of the slowest clock* of the system.
2. Wait for phy_init_done_o to assert. It is expected during this time (before T0), xg_tx_out_clk_o and xg_tx_out_clk_o are not toggling yet.
3. After T0, user drives continuous IDLE patterns until xg_pcsrdy_o asserts.
4. After T1, the PHY is ready to transmit data packets.

* **Note**: The slowest clock refers to the sysbus_clk_i with frequency that can be as low as 20 MHz.

The following figure shows the PHY initialization sequence.

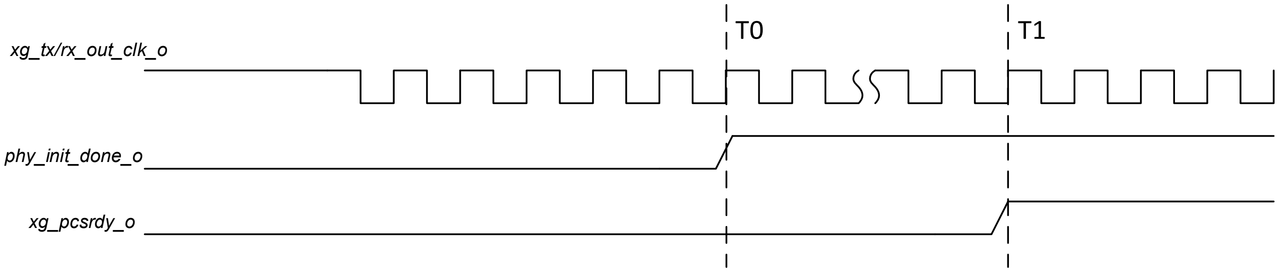


Figure 2.41. PHY Initialization Sequence

2.8.3. PHY + MAC + 1588 Reset Sequence (CertusPro-NX Devices)

This reset sequence applies to both the PHY-only configuration and the PHY+MAC+1588 configuration for CertusPro-NX devices.

2.8.3.1. Reset

An asynchronous reset pin (active low) as system reset is used for resetting the 10G Ethernet IP core. Internal reset logic is implemented to guarantee synchronous deassertion throughout the different clock domain among soft logic blocks. The minimum assertion of reset is five clock cycles of the slowest clock*.

2.8.3.2. Sequence

The following lists the sequence after power-up of the chip or system:

1. Assert reset_n_i pin for five clock cycles of the slowest clock* of the system.
2. Wait for the 10G PHY to be in the ready state.
3. Refer to the Appendix B. Clock and Reset Requirements (CertusPro-NX) section.
   Note: pll_lock_w is an external PLL to generate the 156.25 MHz clock input to xg_tx/rx_clk_i.
4. Send settings through the APB interface to configure the system.
5. The IP is ready to transmit and receive packets.

* **Note**: The slowest clock refers to the sysbus_clk_i with frequency that can be as low as 20 MHz.

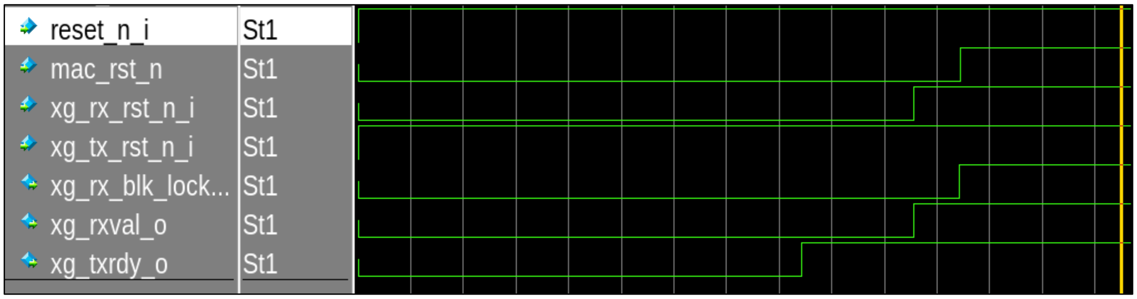


Figure 2.42. Example Testbench Reset Waveform

2.9. Latency

The following table provides the measured latency information for 10G Ethernet IP.

Table 2.10. MAC + PHY Attributes (Avant Devices)

Core	Core Configuration	Latency (ns)	User Bus Width (bits)	Core Clock Frequency (MHz)
MAC + PHY	10G BASE-R TX	252	64	156.25
MAC + PHY	10G BASE-R RX	350	64	156.25

Table 2.11. MAC + PHY Attributes (CertusPro-NX Devices)

Core	Core Configuration	Latency (ns)	User Bus Width (bits)	Core Clock Frequency (MHz)
MAC + PHY	10G BASE-R TX	190*	64	156.25
MAC + PHY	10G BASE-R RX	287*	64	156.25

* Note: This is a nominal value. For CertusPro-NX PHY latency value, refer to the PHY Delay Value section.

3. IP Parameter Description

The configurable attributes of the 10G Ethernet IP core are shown in the following tables. You can configure the IP core by setting the attributes accordingly in the IP Catalog's Module/IP wizard of the Lattice Radiant software.

3.1. MAC + PHY (Avant Devices)

The following table lists the 10G Ethernet IP core configurable attributes for the *MAC + PHY* option. The values set for attributes with corresponding registers serve as the maximum values and cannot be set higher during dynamic reconfiguration. Select IP Option—*MAC + PHY* option.

Table 3.1. MAC + PHY Attributes (Avant Devices)

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General Configuration				
Select IP Option	• MAC + PHY • PHY Only • MAC Only	MAC + PHY	IP option.	—
Host Interface	• AXI4-Lite • APB	AXI4-Lite	Set the register interface.	—
MAC Configuration				
Multicast Address Filtering	• Enabled • Disabled	Disabled	Enables or disables address filtering for multicast frames.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Pause Frame Generation via Ports	• Enabled • Disabled	Disabled	Enables or disables TX pause frame generation via ports.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Statistics Counter Configuration				
Statistic Counter Registers	• Enabled • Disabled	Disabled	Enables or disables statistics counter registers	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Counter Width	• 32 • 64	32	Statistics counters register size.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Statistics	• Enabled • Disabled	Disabled	TX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
RX Statistics	• Enabled • Disabled	Disabled	RX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
PMA Setup-Transmitter Subgroup (default values are recommended)				
TX Amplitude Control [0-63]	0-63	24	Transmitter amplitude adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
TX Pre Cursor [0-31]	0-31	0	Transmitter pre-emphasis level adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
TX Post Cursor [0-31]	0-31	0	Transmitter post-emphasis level adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
PMA Setup-Receiver Subgroup (default values are recommended)				
RX Coupling Mode	• AC Coupling • DC Coupling	AC Coupling	PMA Coupling mode	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
RX Loss of Sig port Enable	• Enabled • Disabled	Enabled	RX Loss of Sig capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
DFE Enable	• Enabled • Disabled	Enabled	DFE capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
RX Adaptive Equalization Enable	• Enabled • Disabled	Enabled	RX adaptive EQ capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY* Selectable only when *DFE Enable == Disabled*
RX Attenuation Block	• -13dB • -2dB	-13dB	RX Attenuation setting. Higher dB loss reflects higher capability of the RX to handle a lossy signal.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY* Selectable only when *RX Adaptive Equalization Enable == Disabled*
PHY Configuration				
FEC Mode	• FCFEC • Disabled	Disabled	Enables the FCFEC or disables the FEC mode.	Enabled when *Select IP Option == PHY only* or *MAC + PHY*
Fast Simulation Mode	• Enabled • Disabled	Disabled	Enables or disables the fast sim mode.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
PCS Lane ID	• AUTO • 0-27	AUTO	Specifies the Lane ID.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
Loopback Mode	• Far End Parallel Loopback • Near End Parallel Loopback • No Loopback	No Loopback	Enables the Far End Parallel Loopback or Near End Parallel Loopback or No Loopback.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*

3.2. MAC Only

The following table lists the 10G Ethernet IP core configurable attributes for the *MAC Only* option. The values set for attributes with corresponding registers serves as the maximum values and cannot set higher than these values during dynamic reconfiguration. Select IP Option—*MAC Only* option.

Table 3.2. MAC Only Attributes

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General Configuration				
Select IP Option	For Avant-AT-G/X devices: • MAC + PHY • PHY Only • MAC Only For CertusPro-NX devices: • MAC Only • PHY only • MAC + PHY + 1588 • MAC + PHY	For Avant-AT-G/X devices: • MAC + PHY For CertusPro-NX devices: • MAC Only	IP option.	—
PHY Interface	• XGMII • 8-bit GMII • 16-bit GMII • MII	XGMII	Set the default PHY interface.	Enabled when *Select IP Option == MAC Only*
Host Interface	For Avant-AT-G/X devices: • APB • AXI4-Lite For CertusPro-NX devices: • APB	For Avant-AT-G/X devices: • AXI4-Lite For CertusPro-NX devices: • APB	Set the register interface.	—
MAC Configuration				
Dynamic Speed Selection	• Enabled • Disabled	Disabled	Enables or disables dynamic speed selection.	Enabled when *Select IP Option == MAC Only.*
Multicast Address Filtering	• Enabled • Disabled	Disabled	Enables or disables address filtering for Multicast frames.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Pause Frame Generation via Ports	• Enabled • Disabled	Disabled	Enables or disables TX pause frame generation via ports.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Statistics Counter Configuration				
Statistics Counter Registers	• Enabled • Disabled	Disabled	Enables or disables statistics counter registers.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Counter Width	• 32 • 64	32	Statistics counters register size.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Statistics	• Enabled • Disabled	Disabled	TX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
RX Statistics	• Enabled • Disabled	Disabled	RX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Non-XGMII Configuration				
Frame Type	• Standard • Jumbo • Super Jumbo	Super Jumbo	Set Ethernet frame type to determine FIFO depth for 8-bit/16-bit GMII or MII configurations or when *Dynamic Speed Selection* is enabled.	Enabled when *Dynamic Speed Selection == Enabled, or PHY Interface == XGMII*

3.3. PHY Only (Avant Devices)

The following table lists the 10G Ethernet IP core configurable attributes for the *PHY Only* option in Avant devices. Select IP Option—*PHY Only* option.

Table 3.3. PHY Only Attributes (Avant Devices)

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General Configuration				
Select IP Option	• MAC + PHY • PHY Only • MAC Only	MAC + PHY	IP option.	—
Host Interface	• AXI4-Lite • APB	AXI4-Lite	Set the register interface.	—
PMA Setup-Transmitter Subgroup (default values are recommended)				
TX Amplitude Control [0-63]	0-63	24	Transmitter amplitude adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
TX Pre Cursor [0-31]	0-31	0	Transmitter pre-emphasis level adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
TX Post Cursor [0-31]	0-31	0	Transmitter post-emphasis level adjustment control.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
PMA Setup-Receiver Subgroup (default values are recommended)				
RX Coupling Mode	• AC Coupling • DC Coupling	AC Coupling	PMA Coupling mode.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
RX Loss of Sig port Enable	• Enabled • Disabled	Enabled	RX Loss of Sig capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
DFE Enable	• Enabled • Disabled	Enabled	DFE capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
RX Adaptive Equalization Enable	- Enabled - Disabled	Enabled	RX adaptive EQ capability.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY* Selectable only when *DFE Enable == Disabled*
RX Attenuation Block	- -13dB - -2dB	-13dB	RX Attenuation setting. Higher dB loss reflects higher capability of the RX to handle a lossy signal.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY* Selectable only when *RX Adaptive Equalization Enable == Disabled*
PHY Configuration				
FEC Mode	- FCFEC - Disabled	Disabled	Enables the FCFEC or disables the FEC mode.	Enabled when *Select IP Option == PHY only* or *MAC + PHY*
Fast Simulation Mode	- Enabled - Disabled	Disabled	Enables or disables the fast sim mode	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
PCS Lane ID	For Avant-AT-G/X devices: - AUTO - 0-27	AUTO	Specifies the Lane ID.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*
Loopback Mode	- Far End Parallel Loopback - Near End Parallel Loopback - No Loopback	No Loopback	Enables the Far End Parallel Loopback or Near End Parallel Loopback or No Loopback.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY*

3.4. PHY Only (CertusPro-NX Devices)

The following table lists the 10G Ethernet IP core configurable attributes for the *PHY Only* option in CertusPro-NX devices. Select IP Option—*PHY Only* option.

Table 3.4. PHY Only Attributes (CertusPro-NX Devices Devices)

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General				
Select IP Option	- MAC Only - PHY Only - MAC + PHY + 1588 - MAC + PHY	MAC Only	IP option.	—
Host Interface	APB	APB	Specifies the preferred interface for register access.	—
PHY Configuration				
PCS Lane ID	For CertusPro-NX devices[1]: - AUTO	AUTO	Specifies the Lane ID. For more details, refer to the PHY (CertusPro-	Enabled when *Select IP Option == PHY Only or MAC + PHY or MAC + PHY + 1588*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
	• 2 • 3 • 6 • 7		NX Devices) section.	
Loopback Mode	• No Loopback • Near End Parallel Loopback • Far End Parallel Loopback	No Loopback	PCS loopback. For more information, refer to the PCS Loopback section.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY or MAC+PHY+1588*

Note:

1. CertusPro-NX devices supported lanes:
    - LFCPNX-100: Lane 2, Lane 3, Lane 6, and Lane 7 only.
    - LFCPNX-50: Lane 2 and Lane 3 only.

3.5. MAC + PHY + 1588 (CertusPro-NX Devices)

The following table lists the 10G Ethernet IP core configurable attributes for the *MAC + PHY + 1588* option. The values set for attributes with corresponding registers serves as the maximum values and cannot set higher than these values during dynamic reconfiguration. Select IP Option— *MAC + PHY + 1588* option.

Table 3.5. MAC + PHY + 1588 Attributes (CertusPro-NX Devices)

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General Configuration				
Select IP Option	• MAC Only • PHY Only • MAC + PHY + 1588 • MAC + PHY	MAC Only	IP option.	—
Host Interface	APB	APB	Set the register interface.	—
MAC Configuration				
Multicast Address Filtering	• Enabled • Disabled	Disabled	Enables or disables address filtering for multicast frames.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Pause Frame Generation via Ports	• Enabled • Disabled	Disabled	Enables or disables TX pause frame generation via ports.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Statistics Counter Configuration				
Statistics Counter Registers	• Enabled • Disabled	Disabled	Enables or disables statistics counter registers.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Counter Width	• 32 • 64	32	Statistics counters register size.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Statistics	• Enabled • Disabled	Disabled	TX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
RX Statistics	• Enabled • Disabled	Disabled	RX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
PHY Configuration				
PCS Lane ID	For CertusPro-NX	AUTO	Specifies the Lane ID.	Enabled when *Select IP*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
	devices[1]: • AUTO • 2 • 3 • 6 • 7		For more details, refer to the PHY (CertusPro-NX Devices) section.	*Option == PHY Only* or *MAC + PHY or MAC + PHY + 1588*
Loopback Mode	• No Loopback • Near End Parallel Loopback • Far End Parallel Loopback	No Loopback	PCS loopback. For more information, refer to the PCS Loopback section.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY or MAC+PHY+1588*

Note:

1. CertusPro-NX devices supported lanes:
    - LFCPNX-100: Lane 2, Lane 3, Lane 6, and Lane 7 only.
    - LFCPNX-50: Lane 2 and Lane 3 only.

3.6. MAC + PHY (CertusPro-NX Devices)

The following table lists the 10G Ethernet IP core configurable attributes for the *MAC + PHY* option. The values set for attributes with corresponding registers serves as the maximum values and cannot set higher than these values during dynamic reconfiguration. Select IP Option— *MAC + PHY* option.

Table 3.6. MAC + PHY Attributes (CertusPro-NX Devices)

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
General Configuration				
Select IP Option	• MAC Only • PHY Only • MAC + PHY + 1588 • MAC + PHY	MAC Only	IP option.	—
Host Interface	APB	APB	Set the register interface.	—
MAC Configuration				
Multicast Address Filtering	• Enabled • Disabled	Disabled	Enables or disables address filtering for multicast frames.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Pause Frame Generation via Ports	• Enabled • Disabled	Disabled	Enables or disables TX pause frame generation via ports.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Statistics Counter Configuration				
Statistics Counter Registers	• Enabled • Disabled	Disabled	Enables or disables statistics counter registers.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
Counter Width	• 32 • 64	32	Statistics counters register size.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
TX Statistics	• Enabled • Disabled	Disabled	TX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*
RX Statistics	• Enabled • Disabled	Disabled	RX statistics.	Enabled when *Select IP Option == MAC Only* or *MAC + PHY*

Attribute	Selectable Values	Default	Description	Dependency on Other Attributes
PHY Configuration				
PCS Lane ID	For CertusPro-NX devices[1]: • AUTO • 2 • 3 • 6 • 7	AUTO	Specifies the Lane ID. For more details, refer to the PHY (CertusPro-NX Devices) section.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY or MAC + PHY + 1588*
Loopback Mode	• No Loopback • Near End Parallel Loopback • Far End Parallel Loopback	No Loopback	PCS loopback. For more information, refer to the PCS Loopback section.	Enabled when *Select IP Option == PHY Only* or *MAC + PHY or MAC+PHY+1588*

Note:

1. CertusPro-NX devices supported lanes:
    - LFCPNX-100: Lane 2, Lane 3, Lane 6, and Lane 7 only.
    - LFCPNX-50: Lane 2 and Lane 3 only.

4. Signal Description

This section describes the 10G Ethernet IP core ports.

4.1. MAC + PHY Signals (Avant Devices)

Table 4.1. Signal Description—MAC + PHY

Port Name	Clock Domain	I/O	Width	Description
Clock and Reset				
reset_n_i	Asynchronous	In	1	Active-low asynchronous reset.
xg_tx_clk_o	—	Out	1	TX clock output (156.25 MHz).
xg_rx_clk_o	—	Out	1	RX clock output (156.25 MHz).
xg_tx_gclk_o	—	Out	2	TX clock forwarded to global clock distribution (156.25 MHz).
xg_rx_gclk_o	—	Out	1	RX clock forwarded to global clock distribution (156.25 MHz).
txmac_clk_i	-	In	1	156.25 MHz clock for MAC interface. It is recommended to use xg_tx_gclk_o[0] as input.
rxmac_clk_i	-	In	1	156.25 MHz clock for MAC interface. It is recommended to use xg_tx_gclk_o[0] as input.
sysbus_clk_i	—	In	1	Clock for the Management module (AXI4-Lite interface). You must use between 100 MHz to 150 MHz clock for XGMII interface and 100 MHz for LMMI interface.
Serial I/O				
pad_refclkn_i	—	In	1	156.25 MHz SERDES PLL reference clock signal (-).
pad_refclkp_i	—	In	1	156.25 MHz SERDES PLL reference clock signal (+).
pad_rxn_i	pad_refclkn_i	In	1	RX- differential signal.
pad_rxp_i	pad_refclkp_i	In	1	RX+ differential signal.
pad_txn_o	pad_refclkn_i	Out	1	TX- differential signal.
pad_txp_o	pad_refclkp_i	Out	1	TX+ differential signal.
Client-Side Interface				
AXI4-Stream Receive Interface				
axis_rx_tdata_o	xg_tx_gclk_o[0][4]	Out	64	AXI4-Stream data from PHY to the client.
axis_rx_tkeep_o	xg_tx_gclk_o[0][4]	Out	8	AXI4-Stream control that indicates which data is valid on axis_rx_tdata_o[ ]. • axis_rx_tkeep_o[0]: axis_rx_tdata_o [7:0] • axis_rx_tkeep_o[1]: axis_rx_tdata_o [15:8] • axis_rx_tkeep_o[2]: axis_rx_tdata_o [23:16] • axis_rx_tkeep_o[3]: axis_rx_tdata_o [31:24] • axis_rx_tkeep_o[4]: axis_rx_tdata_o [39:32] • axis_rx_tkeep_o[5]: axis_rx_tdata_o [47:40] • axis_rx_tkeep_o[6]: axis_rx_tdata_o [55:48] • axis_rx_tkeep_o[7]: axis_rx_tdata_o [63:56]
axis_rx_tvalid_o	xg_tx_gclk_o[0][4]	Out	1	AXI4-Stream data valid.
axis_rx_tuser_o	xg_tx_gclk_o[0][4]	Out	1	AXI4-Stream user signal used to indicate that the frame had a length error, termination error, or a cyclic redundancy check (CRC) error. This signal is qualified with the axis_rx_tlast_o signal.
axis_rx_tlast_o	xg_tx_gclk_o[0][4]	Out	1	AXI4-Stream signal indicating the end of a packet.
AXI4-Stream Transmit Interface				
axis_tx_tready_o	xg_tx_gclk_o[0][4]	Out	1	AXI4-Stream signal indicating that the core can accept data transfer.

Port Name	Clock Domain	I/O	Width	Description
axis_tx_tdata_i	xg_tx_gclk_o[0][4]	In	64	AXI4-Stream data from the client.
axis_tx_tkeep_i	xg_tx_gclk_o[0][4]	In	8	AXI4-Stream control that indicates which data is valid on axis_tx_tdata_i [ ]. • axis_tx_tkeep_i[0]: axis_tx_tdata_i[7:0] • axis_tx_tkeep_i[1]: axis_tx_tdata_i[15:8] • axis_tx_tkeep_i[2]: axis_tx_tdata_i[23:16] • axis_tx_tkeep_i[3]: axis_tx_tdata_i[31:24] • axis_tx_tkeep_i[4]: axis_tx_tdata_i[39:32] • axis_tx_tkeep_i[5]: axis_tx_tdata_i[47:40] • axis_tx_tkeep_i[6]: axis_tx_tdata_i[55:48] • axis_tx_tkeep_i[7]: axis_tx_tdata_i[63:56]
axis_tx_tvalid_i	xg_tx_gclk_o[0][4]	In	1	AXI4-Stream data valid.
axis_tx_tuser_i	xg_tx_gclk_o[0][4]	In	1	AXI4-Stream user signal to indicate an error in the frame. This signal is qualified with the axis_tx_tlast_i signal.
axis_tx_tlast_i	xg_tx_gclk_o[0][4]	In	1	AXI4-Stream signal indicating the end of a packet.
Non-Standard RX/TX Signals				
xg_pcsrdy_o	xg_tx_gclk_o[0][4]	Out	1	The high level of this signal indicates alignment is achieved and RX PCS is ready for transaction.
phy_init_done_o	xg_tx_gclk_o[0][4]	Out	1	The high level of this signal indicates alignment is PHY initialization or reset is done.
pma_rx0_sigdet_hf_o	xg_tx_gclk_o[0][4]	Out	1	Receive high-frequency signal detection output. When asserted, indicates that the receiver is receiving high-frequency signals.
pma_rx0_sigdet_lf_o	xg_tx_gclk_o[0][4]	Out	1	Receive low-frequency signal detection output. When asserted, indicates that the receiver is receiving low-frequency signals.
APB Interface[1]				
apb_psel_i	sysbus_clk_i	In	1	Select signal. Indicates that the completer device is selected, and a data transfer is required.
apb_paddr_i	sysbus_clk_i	In	32	Address signal.
apb_pwdata_i	sysbus_clk_i	In	32	Write data signal.
apb_pwrite_i	sysbus_clk_i	In	1	Direction signal. Write = 1, Read = 0.
apb_penable_i	sysbus_clk_i	In	1	Enable signal. Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	sysbus_clk_i	Out	1	Ready signal. Indicates transfer completion. The completer uses this signal to extend an APB transfer.
apb_prdata_o	sysbus_clk_i	Out	32	Read data signal.
apb_pslverr_o	sysbus_clk_i	Out	1	Transfer success/failure signal. Tie to low.
apb_psel_i	sysbus_clk_i	In	1	Select signal. Indicates that the completer device is selected, and a data transfer is required.
apb_paddr_i	sysbus_clk_i	In	32	Address signal.
apb_pwdata_i	sysbus_clk_i	In	32	Write data signal.
apb_pwrite_i	sysbus_clk_i	In	1	Direction signal. Write = 1, Read = 0.
AXI4-Lite Interface[2]				
axi4l_awaddr_i	sysbus_clk_i	In	32	Write address bus.
axi4l_awvalid_i	sysbus_clk_i	In	1	Write address valid.
axi4l_awready_o	sysbus_clk_i	Out	1	Write address acknowledge.
axi4l_wdata_i	sysbus_clk_i	In	32	Write data bus.
axi4l_wvalid_i	sysbus_clk_i	In	1	Write data valid.
axi4l_wready_o	sysbus_clk_i	Out	1	Write data acknowledge.

Port Name	Clock Domain	I/O	Width	Description
axi4l_wstrb_i	sysbus_clk_i	In	4	AXI4-Lite write strobe. This feature is not supported so tied to 0.
axi4l_bresp_o	sysbus_clk_i	Out	2	Write transaction response.
axi4l_bvalid_o	sysbus_clk_i	Out	1	Write response valid.
axi4l_bready_i	sysbus_clk_i	In	1	Write response acknowledge.
axi4l_araddr_i	sysbus_clk_i	In	32	Read address bus.
axi4l_arvalid_i	sysbus_clk_i	In	1	Read address valid.
axi4l_arready_o	sysbus_clk_i	Out	1	Read address acknowledge.
axi4l_rdata_o	sysbus_clk_i	Out	32	Read data output.
axi4l_rresp_o	sysbus_clk_i	Out	2	Read data response.
axi4l_rvalid_o	sysbus_clk_i	Out	1	Read data/response valid.
axi4l_rready_i	sysbus_clk_i	In	1	Read data acknowledge.
Statistics Vector Interface				
tx_statvec_o	xg_tx_gclk_o[0][4]	Out	28	Contains information on the frame transmitted. This bus is qualified by the tx_staten_o signal. • tx_statvec_o[13:0]: Frame byte count. • tx_statvec_o[14]: Transmit is OK. • tx_statvec_o[15]: MAC control inserted by MAC. • tx_statvec_o[16]: MAC control inserted by client. • tx_statvec_o[17]: Jumbo frame. • tx_statvec_o[18]: Tagged frame. • tx_statvec_o[19]: Broadcast address. • tx_statvec_o[20]: Multicast address. • tx_statvec_o[21]: Underrun error. • tx_statvec_o[22]: CRC error. • tx_statvec_o[23]: Length check error. • tx_statvec_o[24]: Terminate error. • tx_statvec_o[25]: Long frame error. • tx_statvec_o[26]: PTP1588 frame. • tx_statvec_o[27]: Reserved for future use.
tx_staten_o	xg_tx_gclk_o[0][4]	Out	1	When asserted, indicates that the contents of the tx_statvec_o bus are valid. This signal is asserted for three txmac_clk_i periods.
rx_statvec_o	xg_tx_gclk_o[0][4]	Out	28	Contains information on the frame received. This bus is qualified by the rx_staten_o signal. • rx_statvec_o[13:0]: Frame byte count. • rx_statvec_o[14]: Frame dropped. • rx_statvec_o[15]: Broadcast frame received. • rx_statvec_o[16]: Multicast frame received. • rx_statvec_o[17]: CRC error. • rx_statvec_o[18]: VLAN tag detected. • rx_statvec_o[19]: Pause frame. • rx_statvec_o[20]: Length check error. • rx_statvec_o[21]: Frame is too long. • rx_statvec_o[22]: MAC address mismatch. • rx_statvec_o[23]: Unsupported opcode. Only the opcode for pause frame is supported. • rx_statvec_o[24]: Minimum IPG violated. • rx_statvec_o[25]: Receive packet ignored. • rx_statvec_o[26]: PTP1588 frame received. • rx_statvec_o[27]: Reserved for future use.
rx_staten_o	xg_tx_gclk_o[0][4]	Out	1	When asserted, indicates that the contents of the rx_statvec_o bus are valid. This signal is asserted for three rxmac_clk_i periods.

Port Name	Clock Domain	I/O	Width	Description
TX Pause Frame[3]				
tx_pausreq_i	xg_tx_gclk_o[0][4]	In	1	Transmit pause frame. 1 = send request, 0 = do not send request. This is a positive edge-triggered bit. The tx_fc_en bit of TX_CTL register should be set to 1 to enable this feature. When this port is enabled, the tx_pausreq bit of MAC_CTL register is not accessible. Subsequent pause request will not be serviced if the pause frame of the first request is not transmitted yet.
tx_paustm_i	xg_tx_gclk_o[0][4]	In	16	Pause Quanta time. This is equivalent to the tx_paustim bit of PAUSE_TM register.

Notes:

1. Available when APB is selected as the Host Interface.
2. Available when AXI4-Lite is selected as the Host Interface.
3. Available when TX Pause Frame Generation via Ports is enabled.
4. For XGMII configuration, it is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.

4.2. MAC Only Signals

Table 4.2. Signal Description—MAC Only

Port Name	Clock Domain	I/O	Width	Description
Clock and Reset				
reset_n_i	Asynchronous	In	1	Active-low asynchronous reset.
rxmac_clk_i	—	In	1	Clock for Receive AXI4-Stream and XGMII or 8-bit/16-bit GMII or MII data path. 156.25 MHz clock for XGMII interface. 125 MHz clock for 8-bit GMII interface. 156.25 MHz clock for 16-bit interface. 125 MHz clock for MII RX data. For XGMII configuration, it is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.
txmac_clk_i	—	In	1	Clock for Transmit AXI4-Stream and XGMII/GMII/MII data path. 156.25 MHz clock for XGMII interface. 125 MHz clock for 8-bit GMII interface. 156.25 MHz clock for 16-bit interface. 125 MHz clock for MII interface. For XGMII configuration, it is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.
sysbus_clk_i	—	In	1	Clock for Management module (APB interface). It is recommended to use between 20 MHz to 156.25 MHz clock for XGMII interface and 20 to 125 MHz for MII/GMII interface.
XGMII Interface[1]				
xgmii_rxd_i	xg_tx_gclk_o[0][7]	In	64	8-lane Receive SDR XGMII data from PHY. • Lane 0: xgmii_rxd_i[7:0] • Lane 1: xgmii_rxd_i[15:8] • Lane 2: xgmii_rxd_i[23:16] • Lane 3: xgmii_rxd_i[31:24] • Lane 4: xgmii_rxd_i[39:32] • Lane 5: xgmii_rxd_i[47:40] • Lane 6: xgmii_rxd_i[55:48] • Lane 7: xgmii_rxd_i[63:56]

Port Name	Clock Domain	I/O	Width	Description
xgmii_rxc_i	xg_tx_gclk_o[0][7]	In	8	Control bits for each lane in xgmii_rxd_i[ ]. • Lane 0: xgmii_rxc_i[0] • Lane 1: xgmii_rxc_i[1] • Lane 2: xgmii_rxc_i[2] • Lane 3: xgmii_rxc_i[3] • Lane 4: xgmii_rxc_i[4] • Lane 5: xgmii_rxc_i[5] • Lane 6: xgmii_rxc_i[6] • Lane 7: xgmii_rxc_i[7]
xgmii_txd_o	xg_tx_gclk_o[0][7]	Out	64	8-lane Transmit SDR XGMII data to PHY. • Lane 0: xgmii_txd_o[7:0] • Lane 1: xgmii_txd_o[15:8] • Lane 2: xgmii_txd_o[23:16] • Lane 3: xgmii_txd_o[31:24] • Lane 4: xgmii_txd_o[39:32] • Lane 5: xgmii_txd_o[47:40] • Lane 6: xgmii_txd_o[55:48] • Lane 7: xgmii_txd_o[63:56]
xgmii_txc_o	xg_tx_gclk_o[0][7]	Out	8	Control bits for each lane in xgmii_txd_o[ ]. • Lane 0: xgmii_txc_o[0] • Lane 1: xgmii_txc_o[1] • Lane 2: xgmii_txc_o[2] • Lane 3: xgmii_txc_o[3] • Lane 4: xgmii_txc_o[4] • Lane 5: xgmii_txc_o[5] • Lane 6: xgmii_txc_o[6] • Lane 7: xgmii_txc_o[7]
GMII Interface[2]				
gmii_rx_d_i	rxmac_clk_i	In	8	8-bit GMII RX data.
gmii_rx_dv_i	rxmac_clk_i	In	1	Indicates the GMII RX data is valid when asserted.
gmii_rx_err_i	rxmac_clk_i	In	1	Indicates the GMII RX data contains error.
gmii_tx_d_o	txmac_clk_i	Out	8	8-bit GMII TX data.
gmii_tx_en_o	txmac_clk_i	Out	1	Indicates the GMII TX data is valid when asserted.
gmii_tx_err_o	txmac_clk_i	Out	1	Indicates the GMII RX data contains error.
gmii_16_rx_d_i	rxmac_clk_i	In	16	16-bit GMII RX data.
gmii_16_rx_dv_i	rxmac_clk_i	In	2	Indicates the GMII RX data is valid when asserted.
gmii_16_rx_err_i	rxmac_clk_i	In	2	Indicates the GMII RX data contains error.
gmii_16_tx_d_o	txmac_clk_i	Out	16	16-bit GMII TX data.
gmii_16_tx_en_o	txmac_clk_i	Out	2	Indicates the GMII TX data is valid when asserted.
gmii_16_tx_err_o	txmac_clk_i	Out	2	Indicates the GMII RX data contains error.
MII Interface[3]				
mii_rx_d_i	rxmac_clk_i	In	4	4-bit MII RX data.
mii_rx_dv_i	rxmac_clk_i	In	1	Indicates the MII RX data is valid when asserted.
mii_rx_err_i	rxmac_clk_i	In	1	Indicates the MII RX data contains error.
mii_tx_d_o	txmac_clk_i	Out	4	4-bit MII TX data.
mii_tx_en_o	txmac_clk_i	Out	1	Indicates the MII TX data is valid when asserted.

Port Name	Clock Domain	I/O	Width	Description
mii_tx_err_o	txmac_clk_i	Out	1	Indicates the MII RX data contains error.
AXI4-Stream Receive Interface				
axis_rx_tdata_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	64	AXI4-Stream data from PHY to client.
axis_rx_tkeep_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	8	AXI4-Stream control that indicates which data is valid on axis_rx_tdata_o[ ]. • axis_rx_tkeep_o[0]: axis_rx_tdata_o [7:0] • axis_rx_tkeep_o[1]: axis_rx_tdata_o [15:8] • axis_rx_tkeep_o[2]: axis_rx_tdata_o [23:16] • axis_rx_tkeep_o[3]: axis_rx_tdata_o [31:24] • axis_rx_tkeep_o[4]: axis_rx_tdata_o [39:32] • axis_rx_tkeep_o[5]: axis_rx_tdata_o [47:40] • axis_rx_tkeep_o[6]: axis_rx_tdata_o [55:48] • axis_rx_tkeep_o[7]: axis_rx_tdata_o [63:56]
axis_rx_tvalid_o	xg_x_gclk_o[0] [7], rxmac_clk_i	Out	1	AXI4-Stream data valid.
axis_rx_tuser_o	xg_tx_gclk_o[0][7], rxmac_clk_i	Out	1	AXI4-Stream user signal used to indicate that the frame had a length error, termination error, or a CRC error. This signal is qualified with the axis_rx_tlast_o signal.
axis_rx_tlast_o	xg_tx_gclk_o[0][7], rxmac_clk_i	Out	1	AXI4-Stream signal indicating the end of a packet.
AXI4-Stream Transmit Interface				
axis_tx_tready_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	1	AXI4-Stream signal indicating that the core can accept data transfer.
axis_tx_tdata_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	64	AXI4-Stream data from client.
axis_tx_tkeep_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	8	AXI4-Stream control that indicates which data is valid on axis_tx_tdata_i [ ]. • axis_tx_tkeep_i[0]: axis_tx_tdata_i[7:0] • axis_tx_tkeep_i[1]: axis_tx_tdata_i[15:8] • axis_tx_tkeep_i[2]: axis_tx_tdata_i[23:16] • axis_tx_tkeep_i[3]: axis_tx_tdata_i[31:24] • axis_tx_tkeep_i[4]: axis_tx_tdata_i[39:32] • axis_tx_tkeep_i[5]: axis_tx_tdata_i[47:40] • axis_tx_tkeep_i[6]: axis_tx_tdata_i[55:48] • axis_tx_tkeep_i[7]: axis_tx_tdata_i[63:56]
axis_tx_tvalid_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	1	AXI4-Stream data valid.
axis_tx_tuser_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	1	AXI4-Stream user signal used to indicate an error in the frame. This signal is qualified with the axis_tx_tlast_i signal.
axis_tx_tlast_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	1	AXI4-Stream signal indicating the end of a packet.
APB Interface[5]				
apb_psel_i	sysbus_clk_i	In	1	Select signal. Indicates that the completer device is selected and a data transfer is required.
apb_paddr_i	sysbus_clk_i	In	32	Address signal.
apb_pwdata_i	sysbus_clk_i	In	32	Write data signal.
apb_pwrite_i	sysbus_clk_i	In	1	Direction signal. Write = 1, Read = 0
apb_penable_i	sysbus_clk_i	In	1	Enable signal. Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	sysbus_clk_i	Out	1	Ready signal. Indicates transfer completion. The completer uses this signal to extend an APB transfer.
apb_prdata_o	sysbus_clk_i	Out	32	Read data signal.

Port Name	Clock Domain	I/O	Width	Description
apb_pslverr_o	sysbus_clk_i	Out	1	Transfer success or failure signal. Tie to low.
AXI4-Lite Interface				
axi4l_awaddr_i	sysbus_clk_i	In	32	Write address bus.
axi4l_awvalid_i	sysbus_clk_i	In	1	Write address valid.
axi4l_awready_o	sysbus_clk_i	Out	1	Write address acknowledge.
axi4l_wdata_i	sysbus_clk_i	In	32	Write data bus.
axi4l_wvalid_i	sysbus_clk_i	In	1	Write data valid.
axi4l_wready_o	sysbus_clk_i	Out	1	Write data acknowledge.
axi4l_wstrb_i	sysbus_clk_i	In	4	AXI4-Lite write strobe. This feature is not supported so tied to 0.
axi4l_bresp_o	sysbus_clk_i	Out	2	Write transaction response.
axi4l_bvalid_o	sysbus_clk_i	Out	1	Write response valid.
axi4l_bready_i	sysbus_clk_i	In	1	Write response acknowledge.
axi4l_araddr_i	sysbus_clk_i	In	32	Read address bus.
axi4l_arvalid_i	sysbus_clk_i	In	1	Read address valid.
axi4l_arready_o	sysbus_clk_i	Out	1	Read address acknowledge.
axi4l_rdata_o	sysbus_clk_i	Out	32	Read data output.
axi4l_rresp_o	sysbus_clk_i	Out	2	Read data response.
axi4l_rvalid_o	sysbus_clk_i	Out	1	Read data/response valid.
axi4l_rready_i	sysbus_clk_i	In	1	Read data acknowledge.
Statistics Vector Interface				
tx_statvec_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	28	Contains information on the frame transmitted. This bus is qualified by the tx_staten_o signal. • tx_statvec_o[13:0]: Frame byte count • tx_statvec_o[14]: Transmit is OK • tx_statvec_o[15]: MAC control inserted by MAC • tx_statvec_o[16]: MAC control inserted by Client • tx_statvec_o[17]: Jumbo frame • tx_statvec_o[18]: Tagged frame • tx_statvec_o[19]: Broadcast address • tx_statvec_o[20]: Multicast address • tx_statvec_o[21]: Underrun error • tx_statvec_o[22]: CRC error • tx_statvec_o[23]: Length check error • tx_statvec_o[24]: Terminate error • tx_statvec_o[25]: Long Frame error • tx_statvec_o[26]: PTP1588 frame • tx_statvec_o[27]: Reserved for future use
tx_staten_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	1	When asserted, indicates that the contents of the tx_statvec_o bus are valid. This signal is asserted for three txmac_clk_i periods.

Port Name	Clock Domain	I/O	Width	Description
rx_statvec_o	xg_tx_gclk_o[0][7], rxmac_clk_i	Out	28	Contains information on the frame received. This bus is qualified by the rx_staten_o signal. • rx_statvec_o[13:0]: Frame byte count • rx_statvec_o[14]: Frame dropped • rx_statvec_o[15]: Broadcast frame received • rx_statvec_o[16]: Multicast frame received • rx_statvec_o[17]: CRC error • rx_statvec_o[18]: VLAN Tag detected • rx_statvec_o[19]: PAUSE frame • rx_statvec_o[20]: Length check error • rx_statvec_o[21]: Frame is too long • rx_statvec_o[22]: MAC Address mismatch • rx_statvec_o[23]: Unsupported opcode. Only the opcode for PAUSE frame is supported. • rx_statvec_o[24]: Minimum IPG violated • rx_statvec_o[25]: Receive packet ignored • rx_statvec_o[26]: PTP1588 frame received • rx_statvec_o[27]: Reserved for future use
rx_staten_o	xg_tx_gclk_o[0][7], txmac_clk_i	Out	1	When asserted, indicates that the contents of the rx_statvec_o bus are valid. This signal is asserted for three rxmac_clk_i periods.
xg_pcsrdy_i	xg_tx_gclk_o[0][7]	In	1	This is an active-high level status signal from external PHY. Asserted when the PHY link status is ready. In cases where the local fault/remote fault is sending an earlier signal such as during the reset state, then this signal is used by MAC to determine the PHY link status when out of reset. As a result, the RX_STAT_LINK_OK counter is increased.
LINTR				
int_o	sysbus_clk_i	Out	1	Interrupt request. For more information on the assertion of this signal, refer to the Interrupt Registers section.
Miscellaneous				
speed_sel_i[4]	xg_tx_gclk_o[0][7], txmac_clk_i, rxmac_clk_i	In	2	PHY speed selection. The following lists the speed selection for different interfaces: • 0x0: XGMII interface • 0x1: 8-bit GMII interface • 0x2: 16-bit GMII interface • 0x3: MII interface Ensure that the MAC are idle with no packet transmission. After modifying this signal, the client is required to trigger the reset.
TX Pause Frame[6]				
tx_pausreq_i	xg_tx_gclk_o[0][7], txmac_clk_i]	In	1	Transmit PAUSE frame. 1 = send request, 0 = do not send request. This is a positive edge-triggered bit. The tx_fc_en bit of TX_CTL register should be set to 1 to enable this feature. When this port is enabled, the tx_pausreq bit of MAC_CTL register is not accessible. Subsequent pause request will not be serviced if the pause frame of the first request is not transmitted yet.
tx_paustm_i	xg_tx_gclk_o[0][7], txmac_clk_i	In	16	Pause Quanta time. This is equivalent to the tx_paustim bit of the PAUSE_TM register.
rx_pause_cntr_o	xg_tx_gclk_o[0][7], rxmac_clk_i	Out	19	Received Pause counter. This signal is only available in IP option = *MAC only*. Each Pause Quanta corresponds to 512 bit times. In 10G Ethernet IP, the 512 bit times is 8 valid clock cycles. Hence, the counter value is equivalent to quanta time

Port Name	Clock Domain	I/O	Width	Description
				multiplied by 8.
rx_pause_req_o	xg_tx_gclk_o[0][7],rxmac_clk_i	Out	1	Received pause request. This signal is only available in IP option = *MAC only*. Used to indicate there is a pause frame detected.
rx_pause_time_o	xg_tx_gclk_o[0][7], rxmac_clk_i	Out	16	Received Pause Quanta time. This signal is only available in IP option = *MAC only*. This value represents the requested pause time from the pause frame. Each pause quanta corresponds to 512 bit times.

Notes:

1. Available when PHY Interface== XGMII or Dynamic Speed Selection == Enabled.
2. 8-bit GMII is available when PHY Interface== 8-bit GMI or Dynamic Speed Selection == Enabled; 16-bit GMII signals is available when PHY Interface== 16-bit GMII or Dynamic Speed Selection == Enabled.
3. Available when PHY Interface==MII or Dynamic Speed Selection == Enabled.
4. Available when Dynamic Speed Selection == Enabled.
5. Available when APB is selected as Host Interface.
6. Available when TX Pause Frame Generation via Ports is enabled.
7. For XGMII configuration, it is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.

4.3. PHY Only Signals (Avant Devices)

Table 4.3. Signal Description—PHY Only (Avant Devices)

Port Name	Clock Domain	I/O	Width	Description
Serial I/O				
pad_refclkn_i	—	In	1	156.25 MHz SERDES PLL reference clock signal (-).
pad_refclkp_i	—	In	1	156.25 MHz SERDES PLL reference clock signal (+).
pad_rxn_i	pad_refclkn_i	In	1	RX- differential signal.
pad_rxp_i	pad_refclkp_i	In	1	RX+ differential signal.
pad_txn_o	pad_refclkn_i	Out	1	TX- differential signal.
pad_txp_o	pad_refclkp_i	Out	1	TX+ differential signal.
Clock and Reset				
xg_tx_clk_i	—	In	1	TX clock input – 156.25 MHz clock for transmit data path. It is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.
xg_rx_clk_i	—	In	1	RX clock input – 156.25 MHz clock for receive data path. It is recommended to use the xg_tx_gclk_o[0] or xg_tx_clk_o.
xg_tx_clk_o	—	Out	1	TX clock output (156.25 MHz).
xg_rx_clk_o	—	Out	1	RX clock output (156.25 MHz).
xg_tx_gclk_o	—	Out	2	TX clock forwarded to global clock distribution (156.25 MHz).
xg_rx_gclk_o	—	Out	1	RX clock forwarded to global clock distribution (156.25 MHz).
reset_n_i	Asynchronous	In	1	Active-low asynchronous reset.
sysbus_clk_i	—	In	1	Clock for Management module. It is recommended to use 100 MHz.
XGMII				
xg_txc_i	xg_tx_gclk_o[0]	In	8	8-bit TX control signal. bit[0] is the control signal for xg_txd_i [7:0] bit[1] is the control signal for xg_txd_i [15:8] … bit[7] is the control signal for xg_txd_i [63:56] When control bit is high, indicates a control byte, otherwise it is a data byte.
xg_txd_i	xg_tx_gclk_o[0]	In	64	64-bit TX data signal.
xg_rxc_o	xg_tx_gclk_o[0]	Out	8	8-bit RX control signal.

Port Name	Clock Domain	I/O	Width	Description
				bit[0] is the control signal for xg_rxd_o [7:0] bit[1] is the control signal for xg_rxd_o [15:8] … bit[7] is the control signal for xg_rxd_o [63:56] When control bit is high, indicates a control byte, otherwise it is a data byte.
xg_rxd_o	xg_tx_gclk_o[0]	Out	64	64-bit RX data signal.
Non-Standard RX/TX Signals				
xg_rxval_o	xg_tx_gclk_o[0]	Out	1	RX valid signal. When high, indicates that the corresponding values on signals xg_rxc_o and xg_rxd_o are valid.
xg_rxrdy_i	xg_tx_gclk_o[0]	In	1	RX ready signal. This is a flow-control feedback signal. This signal is only available in Avant ES devices.
xg_pcsrdy_o	xg_tx_gclk_o[0]	Out	1	The high level of this signal indicates alignment is achieved and RX PCS is ready for transaction.
phy_init_done_o	xg_tx_gclk_o[0]	Out	1	The high level of this signal indicates alignment is PHY initialization or reset is done.
pma_rx0_sigdet_hf_o	xg_tx_gclk_o[0]	Out	1	Receive high-frequency signal detection output. When asserted, indicates that the receiver is receiving high-frequency signals.
pma_rx0_sigdet_lf_o	xg_tx_gclk_o[0]	Out	1	Receive low-frequency signal detection output. When asserted, indicates that the receiver is receiving low-frequency signals.
AXI4-Lite Interface[1]				
axi4l_awaddr_i	sysbus_clk_i	In	32	Write address bus.
axi4l_awvalid_i	sysbus_clk_i	In	1	Write address valid.
axi4l_awready_o	sysbus_clk_i	Out	1	Write address acknowledge.
axi4l_wdata_i	sysbus_clk_i	In	32	Write data bus.
axi4l_wvalid_i	sysbus_clk_i	In	1	Write data valid.
axi4l_wready_o	sysbus_clk_i	Out	1	Write data acknowledge.
axi4l_wstrb_i	sysbus_clk_i	In	4	AXI4-Lite write strobe. This feature is not supported so tied to 0.
axi4l_bresp_o	sysbus_clk_i	Out	2	Write transaction response.
axi4l_bvalid_o	sysbus_clk_i	Out	1	Write response valid.
axi4l_bready_i	sysbus_clk_i	In	1	Write response acknowledge.
axi4l_araddr_i	sysbus_clk_i	In	32	Read address bus.
axi4l_arvalid_i	sysbus_clk_i	In	1	Read address valid.
axi4l_arready_o	sysbus_clk_i	Out	1	Read address acknowledge.
axi4l_rdata_o	sysbus_clk_i	Out	32	Read data output.
axi4l_rresp_o	sysbus_clk_i	Out	2	Read data response.
axi4l_rvalid_o	sysbus_clk_i	Out	1	Read data/response valid.
axi4l_rready_i	sysbus_clk_i	In	1	Read data acknowledge.
APB Interface[1]				
apb_psel_i	sysbus_clk_i	In	1	Select signal. Indicates that the completer device is selected and a data transfer is required.
apb_paddr_i	sysbus_clk_i	In	32	Address signal.
apb_pwdata_i	sysbus_clk_i	In	32	Write data signal.
apb_pwrite_i	sysbus_clk_i	In	1	Direction signal. Write = 1, Read = 0.
apb_penable_i	sysbus_clk_i	In	1	Enable signal. Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	sysbus_clk_i	Out	1	Ready signal. Indicates transfer completion. The completer uses this signal to extend an APB transfer.

Port Name	Clock Domain	I/O	Width	Description
apb_prdata_o	sysbus_clk_i	Out	32	Read data signal.
apb_pslverr_o	sysbus_clk_i	Out	1	Transfer success/failure signal. Tie to low.

Note:

1. Only one of the two interfaces is available as selected by the *Host Interface.*

4.4. PHY Only Signals (CertusPro-NX Devices)

Table 4.4. Signal Description—PHY Only (CertusPro-NX Devices)

Port Name	Clock Domain	I/O	Width	Description
Serial I/O				
pad_refclkn_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (-).
pad_refclkp_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (+).
pad_rxn_i	pad_refclkn_i	In	1	RX- differential signal.
pad_rxp_i	pad_refclkp	In	1	RX+ differential signal.
pad_txn_o	pad_refclkn_i	Out	1	TX- differential signal.
pad_txp_o	pad_refclkp_i	Out	1	TX+ differential signal.
Other Reference Clock Source				
refclkn0_ext_i	refclkn0_ext_i	In	1	161.132812 MHz reference clock from external I/O pad0 (-).
refclkp0_ext_i	refclkp0_ext_i	In	1	161.132812 MHz reference clock from external I/O pad0 (+)
refclkn1_ext_i	refclkn1_ext_i	In	1	161.132812 MHz reference clock from external I/O pad1 (-).
refclkp1_ext_i	refclkp1_ext_i	In	1	161.132812 MHz reference clock from external I/O pad1 (+).
pll_0_refclk_i	pll_0_refclk_i	In	1	161.132812 MHz generated reference clock from the Left PLL.
pll_1_refclk_i	pll_1_refclk_i	In	1	161.132812 MHz generated reference clock from the Right PLL.
sd_pll_refclk_i	sd_pll_refclk_i	In	1	161.132812 MHz reference clock from fabric.
Reference Clock MUX Tree Control Signals				
use_refmux_i	—	In	1	See Reference Clocks section.
diffioclksel_i	—	In	1	See Reference Clocks section.
clksel_i	—	In	2	See Reference Clocks section.
Clock and Reset				
xg_tx_clk_i	—	In	1	TX clock input – 156.25 MHz clock for transmit data path. *Note: User needs to ensure the clock must be precisely generating 156.25 MHz clock from 322.265625 MHz xg_tx_clk_o.*
xg_tx_rst_n_i	Asynchronous	In	1	TX active low reset. Asynchronous assert, synchronous (xg_tx_clk_i) deassert.
xg_rx_clk_i	—	In	1	RX clock input – 156.25 MHz clock for receive data path. *Note: User needs to ensure the clock must be precisely generating 156.25 MHz clock from 322.265625 MHz xg_tx_clk_o.*
xg_rx_rst_n_i	Asynchronous	In	1	RX active low reset. Asynchronous assert, synchronous (xg_rx_clk_i) deassert.
xg_pcs_clkin_i	—	In	1	Low speed clock input to PMA. Use 156.25 MHz clock.
xg_tx_clk_o	—	Out	1	TX clock output – 322.265625 MHz.
xg_rx_clk_o	—	Out	1	RX clock output – 322.265625 MHz.
XGMII				
xg_txc_i	xg_tx_clk_i	In	8	8-bit TX control signal. bit[0] is the control signal for xg_txd_i [7:0] bit[1] is the control signal for xg_txd_i [15:8] …

Port Name	Clock Domain	I/O	Width	Description
				bit[7] is the control signal for xg_txd_i [63:56] When control bit is high, indicates a control byte, otherwise it is a data byte.
xg_txd_i	xg_tx_clk_i	In	64	64-bit TX data signal.
xg_rxc_o	xg_rx_clk_i	Out	8	8-bit RX control signal. bit[0] is the control signal for xg_rxd_o [7:0] bit[1] is the control signal for xg_rxd_o [15:8] … bit[7] is the control signal for xg_rxd_o [63:56] When control bit is high, indicates a control byte, otherwise it is a data byte.
xg_rxd_o	xg_rx_clk_i	Out	64	64-bit RX data signal.
Non-Standard RX/TX Signals				
xg_rxval_o	xg_rx_clk_i	Out	1	RX valid signal. When high, indicates that the corresponding values on signals xg_rxc_o and xg_rxd_o are valid.
xg_txval_i	xg_tx_clk_i	In	1	TX valid signal. When high, indicates that the corresponding values on signals xg_txc_i and xg_txd_i are valid.
xg_txrdy_o	xg_tx_clk_i	Out	1	TX ready signal. When high, indicates that XGMII PCS is ready to accept the user data and control signal (xg_txc_i and xg_txd_i).
xg_rx_hi_ber_o	xg_rx_clk_i	Out	1	The high level of this signal indicates high bit error rate (BER).
xg_rx_blk_lock_o	xg_rx_clk_i	Out	1	The high level of this signal indicates the block lock is achieved.
APB Interface[1]				
apb_pclk_i	sysbus_clk_i	In	1	Clock for Management module (APB interface). It is recommended to use between 20 MHz to 156.25 MHz clock.
apb_preset_n_i	sysbus_clk_i	In	1	Active low reset. Asynchronous assert, synchronous (apb_pclk_i) deassert.
apb_psel_i	sysbus_clk_i	In	1	Completer select.
apb_penable_i	sysbus_clk_i	In	1	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
apb_paddr_i	sysbus_clk_i	In	16	Address.
apb_pwdata_i	sysbus_clk_i	In	16	Write Data.
apb_pwrite_i	sysbus_clk_i	In	1	Indicates write or read access. 0 – Read. 1 – Write.
apb_prdata_o	sysbus_clk_i	Out	16	Read Data.
apb_pready_o	sysbus_clk_i	Out	1	Ready. The completer uses this signal to extend an APB transfer.

4.5. MAC + PHY + 1588 Signals (CertusPro-NX Devices)

Table 4.5. Signal Description—MAC + PHY + 1588 (CertusPro-NX Devices)

Port Name	Clock Domain	I/O	Width	Description
Serial I/O				
pad_refclkp_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (+).
pad_refclkn_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (-).
pad_rxn_i	pad_refclkn_i	In	1	RX- differential signal.
pad_rxp_i	pad_refclkp_i	In	1	RX+ differential signal.
pad_txn_o	pad_refclkn_i	Out	1	TX- differential signal.
pad_txp_o	pad_refclkp_i	Out	1	TX+ differential signal.
Other Reference Clock Source				

Port Name	Clock Domain	I/O	Width	Description
refclkp0_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad0 (+) .
refclkn0_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad0 (-).
refclkp1_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad1 (+).
refclkn1_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad1 (-).
pll_0_refclk_i	—	In	1	161.132812 MHz generated reference clock from Left PLL.
pll_1_refclk_i	—	In	1	161.132812 MHz generated reference clock from Right PLL.
sd_pll_refclk_i	—	In	1	161.132812 MHz reference clock from fabric.
use_refmux_i	—	In	1	1'b1 – Use reference clock output from Clock MUX Tree (PCSREFMUX). 1'b0 – Use dedicated reference clocks (pad_refclkp/n_i).
diffioclksel_i	—	In	1	1'b1 – Use refclkp/n1_ext_i as reference clocks. 1'b0 – Use refclkp/n0_ext_i as reference clocks.
clksel_i	—	In	2	2'b00 – Use pll_0_refclk_i as reference clocks. 2'b01 – Use pll_1_refclk_i as reference clocks. 2'b10 – Use reference clock based on diffioclksel_i. 2'b11 – Use sd_pll_refclk_i as reference clock.
xg_pcs_clkin_i	—	In	1	Low-speed clock input to PMA. Use 156.25 MHz clock.
Clock and Reset				
sysbus_clk_i	—	In	1	Clock for Management module. It is recommended to use between 20 MHz to 156.25 MHz clock for XGMII interface.
mac_clk	—	In	1	Input clock for MAC and TSU. 156.25 MHz clock for XGMII interface.
mac_rst_n	—	In	1	This active-low mac_rst_n is used to reset MAC and TSU block.
reset_n_i	Asynchronous	In	1	Active low Async global reset.
xg_tx_clk_o	—	Out	1	TX clock output (322.265625 MHz).
xg_rx_clk_o	—	Out	1	RX clock output (322.265625 MHz).
xg_rx_rst_n_i	Asynchronous	In	1	RX Active Low reset. Asynchronous assert, synchronous (xg_rx_clk_i) deassert.
xg_tx_rst_n_i	Asynchronous	In	1	TX Active Low reset. Asynchronous assert, synchronous (xg_tx_clk_i) deassert.
xg_tx_clk_i	—	In	1	TX clock input – 156.25 MHz clock for receive data path.
xg_rx_clk_i	—	In	1	RX clock input – 156.25 MHz clock for receive data path.
AXI4-Stream Receive Interface				
axis_rx_tdata_o	mac_clk	Out	64	AXI4-Stream data from PHY to client.
axis_rx_tkeep_o	mac_clk	Out	8	AXI4-Stream control that indicates which data is valid on AXIs_rx_tdata_o.
axis_rx_tvalid_o	mac_clk	Out	1	AXI4-Stream data valid.
axis_rx_tuser_o	mac_clk	Out	1	AXI4-Stream user signal used to indicate that the frame had a length error, termination error, or a CRC error. This signal is qualified with the AXIs_rx_tlast_o signal.
axis_rx_tlast_o	mac_clk	Out	1	AXI4-Stream signal indicating the end of a packet.
AXI4-Stream Transmit Interface				
axis_tx_tready_o	mac_clk	Out	64	AXI4-Stream signal indicating that the core can accept data transfer.
axis_tx_tdata_i	mac_clk	In	64	AXI4-Stream data from client.
axis_tx_tkeep_i	mac_clk	In	8	AXI4-Stream control that indicates which data is valid on AXIs_rx_tdata_o.
axis_tx_tvalid_i	mac_clk	In	1	AXI4-Stream data valid.

Port Name	Clock Domain	I/O	Width	Description
axis_tx_tuser_i	mac_clk	In	1	AXI4-Stream user signal used to indicate an error in the frame.
axis_tx_tlast_i	mac_clk	In	1	AXI4-Stream signal indicating the end of a packet.
Non-Standard RX/TX Signals				
tx_statvec_o	mac_clk	Out	28	Contains information on the frame transmitted. This bus is qualified by the tx_staten_o signal. tx_statvec_o[13:0]: Frame byte count. tx_statvec_o[14]: Transmit is OK. tx_statvec_o[15]: MAC control inserted by MAC. tx_statvec_o[16]: MAC control inserted by Client. tx_statvec_o[17]: Jumbo frame. tx_statvec_o[18]: Tagged frame. tx_statvec_o[19]: Broadcast address. tx_statvec_o[20]: Multicast address. tx_statvec_o[21]: Underrun error. tx_statvec_o[22]: CRC error. tx_statvec_o[23]: Length check error. tx_statvec_o[24]: Terminate error. tx_statvec_o[25]: Long Frame error. tx_statvec_o[26]: PTP1588 frame. tx_statvec_o[27]: Reserved for future use.
tx_staten_o	mac_clk	Out	1	When asserted, indicates that the contents of the tx_statvec_o bus are valid. This signal is asserted for 3 txmac_clk_i periods.
rx_statvec_o	mac_clk	Out	28	Contains information on the frame received. This bus is qualified by the rx_staten_o signal. rx_statvec_o[13:0]: Frame byte count. rx_statvec_o[14]: Frame dropped. rx_statvec_o[15]: Broadcast frame received. rx_statvec_o[16]: Multicast frame received. rx_statvec_o[17]: CRC error. rx_statvec_o[18]: VLAN Tag detected. rx_statvec_o[19]: PAUSE frame. rx_statvec_o[20]: Length check error. rx_statvec_o[21]: Frame is too long. rx_statvec_o[22]: MAC address mismatch. rx_statvec_o[23]: Unsupported opcode. Only the opcode for PAUSE frame is supported. rx_statvec_o[24]: Minimum IPG violated. rx_statvec_o[25]: Receive packet ignored. rx_statvec_o[26]: PTP1588 frame received. rx_statvec_o[27]: Reserved for future use.
rx_staten_o	mac_clk	Out	1	When asserted, indicates that the contents of the rx_statvec_o bus are valid. This signal is asserted for 3 rxmac_clk_i periods.
int_o	mac_clk	Out	1	Interrupt request. See Interrupt Registers for the details of the assertion of this signal.
xg_txrdy_o	mac_clk	Out	1	TX ready signal. When high, indicates that XGMII PCS is ready to accept the user data and control signal (xg_txc_i and xg_txd_i).
xg_rx_hi_ber_o	mac_clk	Out	1	The high level of this signal indicates high bit error rate (BER).
xg_rx_blk_lock_o	mac_clk	Out	1	The high level of this signal indicates the block lock is

Port Name	Clock Domain	I/O	Width	Description
				achieved.
xg_rxval_o	mac_clk	Out	1	RX valid signal. When high, indicates that the corresponding values on signals xg_rxc_o and xg_rxd_o are valid.
xg_txval_i	mac_clk	In	1	TX valid signal. When high, indicates that the corresponding values on signals xg_txc_i and xg_txd_i are valid.
APB Interfaces				
apb_psel_i	sysbus_clk_i	In	1	Completer select.
apb_penable_i	sysbus_clk_i	In	1	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
apb_paddr_i	sysbus_clk_i	In	32	Address.
apb_pwdata_i	sysbus_clk_i	In	32	Write data.
apb_pwrite_i	sysbus_clk_i	In	1	Indicates write or read access. 0 – Read. 1 – Write.
apb_prdata_o	sysbus_clk_i	Out	32	Read data.
apb_pready_o	sysbus_clk_i	Out	1	Ready. The completer uses this signal to extend an APB transfer.
apb_pslverr_o	sysbus_clk_i	Out	1	Transfer success/failure signal. Tie to low.
Timestamp Signals				
tx_ToD_timestamp	External 125 MHz clock frequency.	In	80	80-bits TX ToD timestamp generated from external ToD counter. This timestamp does not include the fraction nanosecond. The recommended clock frequency to generate ToD counter is 125 MHz. Bit 32 to 79: 48-bit seconds field. Bit 0 to 31: 32-bit nanoseconds field.
rx_ToD_timestamp	External 125 MHz clock frequency.	In	80	80-bits RX ToD timestamp generated from external ToD counter. This timestamp does not include the fraction nanosecond. The recommended clock frequency to generate ToD counter is 125 MHz. Bit 32 to 79: 48-bit seconds field. Bit 0 to 31: 32-bit nanoseconds field.
rx_timestamp_rd_en	mac_clk	In	1	To enable read operation to RX timestamp FIFO.
rx_timestamp_rd_data	mac_clk	Out	128	Data out from the RX timestamp FIFO. The data contains timestamp + sequence number. rx_timestamp_rd_data[79:0]: timestamp rx_timestamp_rd_data[95:80]: sequence number rx_timestamp_rd_data[127:96]: reserved
rx_timestamp_rd_data_valid	mac_clk	Out	1	Data out valid from the RX timestamp FIFO.
rx_timestamp_buffer_data_count	mac_clk	Out	32	RX FIFO timestamp count. Indicates the number of timestamps stored in the FIFO for you to read.
rx_timestamp_buffer_empty_n	mac_clk	Out	1	Active-low signal. This signal indicates the RX timestamp buffer status during 2-step PTP.
tx_timestamp_rd_en	mac_clk	In	1	To enable read operation to TX timestamp FIFO.
tx_timestamp_rd_data	mac_clk	Out	128	Data out from the TX timestamp FIFO. The data contains timestamp + sequence number. tx_timestamp_rd_data[79:0]: timestamp tx_timestamp_rd_data[95:80]: sequence number tx_timestamp_rd_data[127:96]: reserved
tx_timestamp_rd_data_valid	mac_clk	Out	1	Data out valid from the TX timestamp FIFO.
tx_timestamp_buffer_data_	mac_clk	Out	32	TX FIFO timestamp count.

Port Name	Clock Domain	I/O	Width	Description
count				
tx_timestamp_buffer_empty_n	mac_clk	Out	1	Active-low signal. This signal indicates the TX timestamp buffer status during 2-step PTP.

4.5.1. Clock Signal for MAC + PHY + 1588 Signals

Table 4.6. Clock Signal for MAC + PHY + 1588 Signals

Interface Signal	Clock Signal
tx_ToD_timestamp/rx_ToD_timestamp	External 125 MHz clock frequency.
rx_timestamp_rd* / tx_timestamp_rd* rx_timestamp_rd_data* / tx_timestamp_rd_data* rx_timestamp_buffer* / tx_timestamp_buffer*	Mac_clk

4.6. MAC + PHY Signals (CertusPro-NX Devices)

Table 4.7. Signal Description—MAC + PHY (CertusPro-NX Devices)

Port Name	Clock Domain	I/O	Width	Description
Serial I/O				
pad_refclkp_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (+).
pad_refclkn_i	—	In	1	161.132812 MHz SERDES PLL reference clock signal (-).
pad_rxn_i	pad_refclkn_i	In	1	RX- differential signal.
pad_rxp_i	pad_refclkp_i	In	1	RX+ differential signal.
pad_txn_o	pad_refclkn_i	Out	1	TX- differential signal.
pad_txp_o	pad_refclkp_i	Out	1	TX+ differential signal.
Other Reference Clock Source				
refclkp0_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad0 (+) .
refclkn0_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad0 (-).
refclkp1_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad1 (+).
refclkn1_ext_i	—	In	1	161.132812 MHz reference clock from external I/O pad1 (-).
pll_0_refclk_i	—	In	1	161.132812 MHz generated reference clock from Left PLL.
pll_1_refclk_i	—	In	1	161.132812 MHz generated reference clock from Right PLL.
sd_pll_refclk_i	—	In	1	161.132812 MHz reference clock from fabric.
use_refmux_i	—	In	1	1’b1 – Use reference clock output from Clock MUX Tree (PCSREFMUX). 1’b0 – Use dedicated reference clocks (pad_refclkp/n_i).
diffioclksel_i	—	In	1	1’b1 – Use refclkp/n1_ext_i as reference clocks. 1’b0 – Use refclkp/n0_ext_i as reference clocks.
clksel_i	—	In	2	2’b00 – Use pll_0_refclk_i as reference clocks. 2’b01 – Use pll_1_refclk_i as reference clocks. 2’b10 – Use reference clock based on diffioclksel_i. 2’b11 – Use sd_pll_refclk_i as reference clock.
xg_pcs_clkin_i	—	In	1	Low-speed clock input to PMA. Use 156.25 MHz clock.
Clock and Reset				
sysbus_clk_i	—	In	1	Clock for the Management module. It is recommended to use between 20 MHz to 156.25 MHz clock for XGMII interface.
mac_clk	—	In	1	Input clock for MAC. 156.25 MHz clock for XGMII interface.
mac_rst_n	—	In	1	This active-low mac_rst_n is used to reset MAC block.

Port Name	Clock Domain	I/O	Width	Description
reset_n_i	Asynchronous	In	1	Active low Async global reset.
xg_tx_clk_o	—	Out	1	TX clock output (322.265625 MHz).
xg_rx_clk_o	—	Out	1	RX clock output (322.265625 MHz).
xg_rx_rst_n_i	Asynchronous	In	1	RX Active Low reset. Asynchronous assert, synchronous (xg_rx_clk_i) deassert.
xg_tx_rst_n_i	Asynchronous	In	1	TX Active Low reset. Asynchronous assert, synchronous (xg_tx_clk_i) deassert.
xg_tx_clk_i	—	In	1	TX clock input – 156.25 MHz clock for receive data path.
xg_rx_clk_i	—	In	1	RX clock input – 156.25 MHz clock for receive data path.
AXI4-Stream Receive Interface				
axis_rx_tdata_o	mac_clk	Out	64	AXI4-Stream data from PHY to client.
axis_rx_tkeep_o	mac_clk	Out	8	AXI4-Stream control that indicates which data is valid on AXIs_rx_tdata_o.
axis_rx_tvalid_o	mac_clk	Out	1	AXI4-Stream data valid.
axis_rx_tuser_o	mac_clk	Out	1	AXI4-Stream user signal used to indicate that the frame had a length error, termination error, or a CRC error. This signal is qualified with the AXIs_rx_tlast_o signal.
axis_rx_tlast_o	mac_clk	Out	1	AXI4-Stream signal indicating the end of a packet.
AXI4-Stream Transmit Interface				
axis_tx_tready_o	mac_clk	Out	64	AXI4-Stream signal indicating that the core can accept data transfer.
axis_tx_tdata_i	mac_clk	In	64	AXI4-Stream data from client.
axis_tx_tkeep_i	mac_clk	In	8	AXI4-Stream control that indicates which data is valid on AXIs_rx_tdata_o.
axis_tx_tvalid_i	mac_clk	In	1	AXI4-Stream data valid.
axis_tx_tuser_i	mac_clk	In	1	AXI4-Stream user signal used to indicate an error in the frame.
axis_tx_tlast_i	mac_clk	In	1	AXI4-Stream signal indicating the end of a packet.
Non-Standard RX/TX Signals				
tx_statvec_o	mac_clk	Out	28	Contains information on the frame transmitted. This bus is qualified by the tx_staten_o signal. tx_statvec_o[13:0]: Frame byte count. tx_statvec_o[14]: Transmit is OK. tx_statvec_o[15]: MAC control inserted by MAC. tx_statvec_o[16]: MAC control inserted by Client. tx_statvec_o[17]: Jumbo frame. tx_statvec_o[18]: Tagged frame. tx_statvec_o[19]: Broadcast address. tx_statvec_o[20]: Multicast address. tx_statvec_o[21]: Underrun error. tx_statvec_o[22]: CRC error. tx_statvec_o[23]: Length check error. tx_statvec_o[24]: Terminate error. tx_statvec_o[25]: Long Frame error. tx_statvec_o[26]: PTP1588 frame. tx_statvec_o[27]: Reserved for future use.
tx_staten_o	mac_clk	Out	1	When asserted, indicates that the contents of the tx_statvec_o bus are valid. This signal is asserted for 3 txmac_clk_i periods.
rx_statvec_o	mac_clk	Out	28	Contains information on the frame received. This bus is qualified by the rx_staten_o signal. rx_statvec_o[13:0]: Frame byte count.

Port Name	Clock Domain	I/O	Width	Description
				rx_statvec_o[14]: Frame dropped. rx_statvec_o[15]: Broadcast frame received. rx_statvec_o[16]: Multicast frame received. rx_statvec_o[17]: CRC error. rx_statvec_o[18]: VLAN Tag detected. rx_statvec_o[19]: PAUSE frame. rx_statvec_o[20]: Length check error. rx_statvec_o[21]: Frame is too long. rx_statvec_o[22]: MAC address mismatch. rx_statvec_o[23]: Unsupported opcode. Only the opcode for PAUSE frame is supported. rx_statvec_o[24]: Minimum IPG violated. rx_statvec_o[25]: Receive packet ignored. rx_statvec_o[26]: PTP1588 frame received. rx_statvec_o[27]: Reserved for future use.
rx_staten_o	mac_clk	Out	1	When asserted, indicates that the contents of the rx_statvec_o bus are valid. This signal is asserted for 3 rxmac_clk_i periods.
int_o	mac_clk	Out	1	Interrupt request. See Interrupt Registers for the details of the assertion of this signal.
xg_txrdy_o	mac_clk	Out	1	TX ready signal. When high, indicates that XGMII PCS is ready to accept the user data and control signal (xg_txc_i and xg_txd_i).
xg_rx_hi_ber_o	mac_clk	Out	1	The high level of this signal indicates high bit error rate (BER).
xg_rx_blk_lock_o	mac_clk	Out	1	The high level of this signal indicates the block lock is achieved.
xg_rxval_o	mac_clk	Out	1	RX valid signal. When high, indicates that the corresponding values on signals xg_rxc_o and xg_rxd_o are valid.
xg_txval_i	mac_clk	In	1	TX valid signal. When high, indicates that the corresponding values on signals xg_txc_i and xg_txd_i are valid.
APB Interfaces				
apb_psel_i	sysbus_clk_i	In	1	Completer select.
apb_penable_i	sysbus_clk_i	In	1	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
apb_paddr_i	sysbus_clk_i	In	32	Address.
apb_pwdata_i	sysbus_clk_i	In	32	Write data.
apb_pwrite_i	sysbus_clk_i	In	1	Indicates write or read access. 0 – Read. 1 – Write.
apb_prdata_o	sysbus_clk_i	Out	32	Read data.
apb_pready_o	sysbus_clk_i	Out	1	Ready. The completer uses this signal to extend an APB transfer.
apb_pslverr_o	sysbus_clk_i	Out	1	Transfer success or failure signal. Tie to low.

5. Register Description

This section describes the registers for the 10G Ethernet MAC + PHY IP core.

5.1. MAC + PHY Registers (Avant Devices)

The registers available in the Ethernet MAC + PHY option include all registers from the MAC and PHY.

For register descriptions in the 10GbE MAC and PHY, refer to the MAC Registers and PHY Registers sections.

Note: With the MAC + PHY selected as the IP option, only the AXI4-Lite or APB interface is available for access to both PCS and MAC registers.

5.2. MAC Registers

All registers are accessed through the APB interface when the IP is configured as *MAC Only*.

Table 5.1. Access Types for MAC Only

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access.
WO	Returns 0	Updates register value.
RW	Returns register value	Updates register value.
RW1C	Returns register value	Writing 1’b1 on register bit clears the bit to 1’b0. Writing 1’b0 on register bit is ignored.
Reserved	Returns 0	Ignores write access.

5.2.1. Configuration Registers for MAC

The following table lists the 10GbE MAC configuration registers.

Table 5.2. Configuration Registers for MAC

Offset	Register Name	Access	Description
0x000	MODE	RW	Mode of Operation Register
0x004	TX_CTL	RW	Transmit MAC Control Register
0x008	RX_CTL	RW	Receive MAC Control Register
0x00C	MAX_PKT_LNGTH	RW	Maximum Packet Size Register
0x010	IPG_VAL	RW	IPG Value Register
0x014	MAC_ADDR_0	RW	MAC Address Register Word 0
0x018	MAC_ADDR_1	RW	MAC Address Register Word 1
0x01C	TX_RX_STS	RO	Transmit and Receive Status Register
0x020	VLAN_TAG	RO	VLAN Tag Register
0x024	MC_TABLE_0	RW	Multicast Tables Register Word 0
0x028	MC_TABLE_1	RW	Multicast Tables Register Word 1
0x02C	PAUSE_OPCODE	RW	Pause Opcode
0x030	MAC_CTL	RW	MAC Control Register
0x034	PAUSE_TM	RW	Pause Time Register

5.2.1.1. MODE Register

This register enables the operation of the MAC. It can be written at any time.

Table 5.3. MODE Register

Bit	Label	Description	Default
[31:2]	Reserved	Reserved bits.	0
[1]	tx_en	TX MAC Enable When set to 1, the TX MAC is enabled to transmit frames.	0
[0]	rx_en	RX MAC Enable When set to 1, the RX MAC is enabled to receive frames.	0

5.2.1.2. TX_CTL Register

This register should be overwritten only when the TX MAC is disabled. Writing this register while TX MAC is active results in unpredictable behavior.

Table 5.4. TX_CTL Register

Bit	Label	Description	Default
[31:5]	Reserved	Reserved bits.	0
[4]	tx_pass_pream	TX Custom Preamble Mode. When set to 1, the TX MAC operates in custom preamble mode where it preserves the preamble field presented on the client interface. Note: Custom preamble mode is only supported by XGMII interface.	0
[3]	transmit_short	Transmit Short Frames. When set to 1, the TX MAC transmits frames shorter than 64 bytes without adding padding bytes. When set to 0, TX MAC adds padding bytes when frames are shorter than 64 bytes before transmitted to the PHY.	0
[2]	tx_ipg_stretch	IFG Stretch Mode. When set to 1, the TX MAC operates in the IFG stretch mode to match the data rates of OC-192. The IPG required to match OC-192 is added to the value specified in the IPG_VAL register.	0
[1]	tx_fc_en	Flow-control Enable. When set to 1, this enables the flow control functionality of the TX MAC. This bit must be set for the TX MAC to transmit a pause frame.	0
[0]	tx_pass_fcs	In-band FCS Enable. When set to 1, the FCS field generation is disabled in the TX MAC, and the client is responsible to generate the appropriate FCS field.	0

5.2.1.3. RX_CTL Register

This register should be overwritten only when the RX MAC is disabled. Writing this register while RX MAC is active results in unpredictable behavior.

Table 5.5. RX_CTL Register

Bit	Label	Description	Default
[15:8]	Reserved	Reserved bits.	0
[7]	rx_pass_pream	RX Custom Preamble Mode. When set to 1, the RX MAC operates in custom preamble mode where it preserves the preamble field of the received frame. Note: Custom preamble mode is only supported by XGMII interface.	0
[6]	drop_mac_ctrl	Drop MAC Control Frames. When set to 1, all MAC control frames are not passed on to the client interface.	0
[5]	receive_short	Receive Short Frames. When set to 1, enables the RX MAC to receive frames shorter than 64 bytes.	0*

Bit	Label	Description	Default
[4]	receive_bc	Receive Broadcast Frames. When set to 1, it enables the RX MAC to receive broadcast frames.	0
[3]	receive_all_mc	Receive Multicast Frames. When set to 1, the multicast frames are received per the filtering rules for such frames. When set to 0, no multicast (except pause frames) frames are received.	0
[2]	rx_pause_en	Receive Pause Frames. When set to 1, the RX MAC indicates the pause frame reception to the TX MAC. Pause frames are received and transferred to the AXI4-Stream interface only when the drop_mac_ctrl bit is not set.	0
[1]	rx_pass_fcs	In-band FCS Passing. When set to 1, the FCS and any of the padding bytes are passed to the AXI4-Stream interface. When set to 0, the MAC strips off the FCS and any padding bytes before transferring it to the AXI4-Stream interface.	0
[0]	prms	Promiscuous Mode. When set to 1, all filtering schemes are abandoned, and the RX MAC receives frames with any address.	0

* **Note**: If the L/T value is less than 46 bytes, it detects as short frame, and it will not be dropped. If the L/T value is less than 46 bytes but the payload is more than the defined value, then the extra payload will be treated as the padded byte.

5.2.1.4. MAX_PKT_LNGTH Register

This register should be overwritten only when the MAC is disabled. All frames longer than the value (number of bytes) in this register is tagged as long frames. Writing this register while the MAC is active results in unpredictable behavior.

Table 5.6. MAX_PKT_LNGTH Register

Bit	Label	Description	Default
[31:14]	Reserved	Reserved bits.	0
[13:0]	max_pkt_len	Maximum Frame Length. Used only for statistical purposes, all frames longer than the value given here are marked as long. This value does not affect the frame's reception.	0

5.2.1.5. IPG_VAL Register

This register contains the IPG value to be used by the TX MAC. Back-to-back packets in the transmit buffer is sent out with the IPG setting programmed in this register with DIC.

Table 5.7. IPG_VAL Register

Bit	Label	Description	Default
[15:5]	Reserved	Reserved bits.	—
[4:0]	tx_ipg	Transmit Inter-Packet Gap. Specifies the amount of inter-frame gap in increments of 4 bytes. For details, refer to the Transmit MAC section. A value of 0 of this register is prohibited.	5'h1

5.2.1.6. MAC_ADDR_0 and MAC_ADDR_1 Register

The MAC address is stored in the registers in hexadecimal form. For example, to set the MAC address to: AC-DE-48-00-00-80 would require writing 0x48_00_00_80 to address 0x014 (MAC_ADDR_0). 0xAC_DE to address 0x018 (MAC_ADDR_1).

Table 5.8. MAC_ADDR_0 Register

Bit	Label	Description	Default
[31:0]	mac_addr_0	First four bytes of the MAC address. Ethernet address assigned to the port that is supported by the MAC.	0

Table 5.9. MAC_ADDR_1 Register

Bit	Label	Description	Default
[31:16]	Reserved	Reserved bits.	0
[15:0]	mac_addr_1	Last two bytes of the MAC address. Ethernet address assigned to the port that is supported by the MAC.	0

5.2.1.7. TX_RX_STS Register

Table 5.10. TX_RX_STS Register

Bit	Label	Description	Default
[31:5]	Reserved	Reserved bits.	0
[4]	link_ok	Link is OK. When set to 1, this indicates that no fault symbols were received on the link.	0
[3]	remote_fault	Remote fault. When set to 1, this indicates that remote fault symbols were received on the link.	0
[2]	local_fault	Local fault. When set to 1, this indicates that local fault symbols were received on the link.	0
[1]	rx_idle	Receive MAC idle. When set to 1, this indicates that the RX MAC is inactive.	0
[0]	tx_idle	Transmit MAC idle. When set to 1, this indicates that the TX MAC is inactive.	0

5.2.1.8. VLAN_TAG Register

This register has the VLAN tag field of the most recent tagged frame that was received.

Table 5.11. VLAN_TAG Register

Bit	Label	Description	Default
[31:16]	Reserved	Reserved bits.	0
[15:0]	vlan_tag	VLAN tag ID.	0

5.2.1.9. MC_TABLE_0 and MC_TABLE_1 Register

When the core is programmed to receive multicast frames, a filtering scheme is used to decide whether the frame should be received or not. This 64-bit matrix forms the hash table that is used to filter out the incoming multicast frames. For details, refer to the Receive MAC section.

Table 5.12. MC_TABLE_0 Register

Bit	Label	Description	Default
[31:0]	mc_table_0	Multicast Table Word 0. First 4-bytes of the 64-bit hash.	0

Table 5.13. MC_TABLE_1 Register

Bit	Label	Description	Default
[31:0]	mc_table_1	Multicast Table Word 1. Last 4-bytes of the 64-bit hash.	0

5.2.1.10. PAUSE_OPCODE Register

This register contains the pause opcode. This is compared against the opcode in the received pause frame. This value is also included in any pause frame transmitted by the MAC.

Table 5.14. PAUSE_OPCODE Register

Bit	Label	Description	Default
[31:16]	Reserved	Reserved bits.	0
[15:0]	pause_opcode	Pause opcode.	16’h01

5.2.1.11. MAC_CTL Register

Table 5.15. MAC_CTL Register

Bit	Label	Description	Default
[31:5]	Reserved	Reserved bits.	—
[4]	ignore_pkt	Ignore packet. When set to 1, the RX MAC ignores or drops incoming packets.	0
[3:1]	Reserved	Reserved bits.	—
[0]	tx_pausreq	Transmit pause frame. 1 = send request, 0 = do not send request. This is a positive edge-triggered bit. The tx_fc_en bit of TX_CTL register should be set to 1 to enable this feature. Set this register to 1 during send request, maintaining it for at least the hold time specified below before resetting it to 0. Standard maximum frame size of 1,518 bytes requires a hold time of 1,214.4 ns (1,518 bytes divided by 8 bytes per clock cycle = 1,214.4 ns). Super jumbo frame size of 9,600 bytes requires a hold time of 7,680 ns (9,600 bytes divided by 8 bytes per clock cycle = 7,680 ns). Clock cycle of 6.4 ns (per 8 bytes).	0

5.2.1.12. PAUSE_TM Register

This register has the pause time for a flow control packet sourced by the 10 Gb MAC transmitter.

Table 5.16. PAUSE_TM Register

Bit	Label	Description	Default
[31:16]	Reserved	Reserved bits.	—
[15:0]	tx_paustim	Pause duration.	0

5.2.2. Interrupt Registers

For details of these registers, refer to the *Lattice Interrupt Interface* section of the Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide (FPGA-UG-02039).

Table 5.17. Summary of Interrupt Registers

Offset	Register Name	Access	Description
0x038	INT_STATUS	RW1C	Interrupt Status Register.
0x03C	INT_ENABLE	RW	Interrupt Enable Register.
0x040	INT_SET	WO	Interrupt Set Register.

5.2.2.1. INT_STATUS Register

Table 5.18. INT_STATUS Register

Bit	Label	Description	Default
[31:2]	Reserved	Reserved bits.	0
[1]	remote_fault_int	Remote fault symbols received.	0
[0]	local_fault_int	Local fault symbols received.	0

5.2.2.2. INT_ENABLE Register

Table 5.19. INT_ENABLE Register

Bit	Label	Description	Default
[31:2]	Reserved	Reserved bits.	0
[1]	remote_fault_en	Enables remote_fault_int.	0
[0]	local_fault_en	Enables local_fault_int.	0

5.2.2.3. INT_SET Register

Table 5.20. INT_SET Register

Bit	Label	Description	Default
[31:2]	Reserved	Reserved bits.	0
[1]	remote_fault_set	Sets remote_fault_int.	0
[0]	local_fault_set	Sets local_fault_int.	0

5.2.3. Statistics Counters

These statistic counters are wraparound counters and can only be reset when the system reset is asserted. The default value of these counters is 0.

The register name with "_0" refers to the least significant word of the counter and "_1" refers to the most significant word.

Table 5.21. Summary of Statistics Counters

Offset	Register Name	Access	Description
0x044	TX_STAT_PKT_LNGTH_0	RO	Transmit Packet Length Statistics Counter. Indicates the total number of octets transmitted in a particular frame. tx_statvec_o[13:0] is used to implement this counter.
0x048	TX_STAT_PKT_LNGTH_1	RO	
0x04C	TX_STAT_ERR_0	RO	Transmit TX Error Statistics Counter. Counts the total number of PHY terminated packet. The tx_statvec_o[24] is used to implement this counter.
0x050	TX_STAT_ERR_1	RO	
0x054	TX_STAT_UNDER_RUN_0	RO	Transmit Underrun Error Statistics Counter. Counts the total number of underrun packets transmitted. tx_statvec_o[21] is used to implement this counter.
0x058	TX_STAT_UNDER_RUN_1	RO	
0x05C	TX_STAT_CRC_ERR_0	RO	Transmit CRC Error Statistics Counter. Counts the total number of packets transmitted with CRC error. tx_statvec_o[22] is used to implement this counter.
0x060	TX_STAT_CRC_ERR_1	RO	
0x064	TX_STAT_LNGTH_ERR_0	RO	Transmit Length Error Statistics Counter. Counts the total number of packets transmitted with

Offset	Register Name	Access	Description
0x068	TX_STAT_LNGTH_ERR_1	RO	length of the packet and length in the Length/Type field mismatch. tx_statvec_o[23] is used to implement this counter.
0x06C	TX_STAT_LNG_PKT_0	RO	Transmit Long packet Statistics Counter. Counts the total number of packets transmitted with length of the packet longer than the max_frm_size. tx_statvec_o[25] is used to implement this counter.
0x070	TX_STAT_LNG_PKT_1	RO	
0x074	TX_STAT_MULTCST_0	RO	Transmit Multicast Packet Statistics Counter. Counts the total number of multicast packets transmitted. tx_statvec_o[20] is used to implement this counter.
0x078	TX_STAT_MULTCST_1	RO	
0x07C	TX_STAT_BRDCST_0	RO	Transmit Broadcast Packet Statistics Counter. Counts the total number of broadcast packets transmitted. tx_statvec_o[19] is used to implement this counter.
0x080	TX_STAT_BRDCST_1	RO	
0x084	TX_STAT_CNT_0	RO	Transmit Control Packet Statistics Counter. Counts the total number of control packets (pause frame) transmitted by the AXI4-Stream interface. tx_statvec_o[16] is used to implement this counter.
0x088	TX_STAT_CNT_1	RO	
0x08C	TX_STAT_JMBO_0	RO	Transmit Jumbo Packet Statistics Counter. Counts the total number of jumbo packets transmitted. tx_statvec_o[17] is used to implement this counter.
0x090	TX_STAT_JMBO_1	RO	
0x094	TX_STAT_PAUSE_0	RO	Transmit Pause Packet Statistics Counter. Counts the total number of pause packets inserted by the MAC core (enabled through MAC_CTL register). tx_statvec_o[15] is used to implement this counter.
0x098	TX_STAT_PAUSE_1	RO	
0x09C	TX_STAT_VLN_TG_0	RO	Transmit VLAN Tag Statistics Counter. Counts the total number of tagged packets transmitted. tx_statvec_o[18] is used to implement this counter.
0x0A0	TX_STAT_VLN_TG_1	RO	
0x0A4	TX_STAT_PKT_OK_0	RO	Transmit Packet OK Statistics Counter. Counts the total number of packets transmitted without any errors. tx_statvec_o[14] is used to implement this counter.
0x0A8	TX_STAT_PKT_OK_1	RO	
0x0AC	TX_STAT_PKT_64_0	RO	Transmit Packet 64 Statistics Counter. Counts the total number of packets transmitted with length equal to 64.
0x0B0	TX_STAT_PKT_64_1	RO	
0x0B4	TX_STAT_PKT_65_127_0	RO	Transmit Packet 65 - 127 Statistics Counter. Counts the total number of packets transmitted with lengths between 65 and 127.
0x0B8	TX_STAT_PKT_65_127_1	RO	
0x0BC	TX_STAT_PKT_128_255_0	RO	Transmit Packet 128-255 Statistics Counter. Counts the total number of packets transmitted with lengths between 128 and 255.
0x0C0	TX_STAT_PKT_128_255_1	RO	
0x0C4	TX_STAT_PKT_256_511_0	RO	Transmit Packet 256-511 Statistics Counter. Counts the total number of packets transmitted with lengths between 256 and 511.
0x0C8	TX_STAT_PKT_256_511_1	RO	

Offset	Register Name	Access	Description
0x0CC	TX_STAT_PKT_512_1023_0	RO	Transmit Packet 512-1023 Statistics Counter. Counts the total number of packets transmitted with lengths between 512 and 1,023.
0x0D0	TX_STAT_PKT_512_1023_1	RO	
0x0D4	TX_STAT_PKT_1024_1518_0	RO	Transmit Packet 1024-1518 Statistics Counter. Counts the total number of packets transmitted with lengths between 1,024 and 1,518.
0x0D8	TX_STAT_PKT_1024_1518_1	RO	
0x0DC	TX_STAT_PKT_1518_0	RO	Transmit Packet 1518 Statistics Counter. Counts the total number of packets transmitted with length greater than 1,518.
0x0E0	TX_STAT_PKT_1518_1	RO	
0x0E4	TX_STAT_FRM_ERR_0	RO	Transmit Frame Error Statistics Counter. Counts the total number of packets transmitted with error. tx_statvec_o[14] is used to implement this counter.
0x0E8	TX_STAT_FRM_ERR_1	RO	
0x0EC	TX_STAT_PKT_1519_2047_0	RO	Transmit Packet 1519-2047 Statistics Counter. Counts the total number of packets transmitted with lengths between 1,024 and 2,047.
0x0F0	TX_STAT_PKT_1519_2047_1	RO	
0x0F4	TX_STAT_PKT_2048_4095_0	RO	Transmit Packet 2048-4095 Statistics Counter. Counts the total number of packets transmitted with lengths between 2,048 and 4,095.
0x0F8	TX_STAT_PKT_2048_4095_1	RO	
0x0FC	TX_STAT_PKT_4096_9216_0	RO	Transmit Packet 4096-9216 Statistics Counter. Counts the total number of packets transmitted with lengths between 4,096 and 9,216.
0x100	TX_STAT_PKT_4096_9216_1	RO	
0x104	TX_STAT_PKT_9217_16383_0	RO	Transmit Packet 9217-16383 Statistics Counter. Counts the total number of packets transmitted with lengths between 9,217 and 16,383.
0x108	TX_STAT_PKT_9217_16383_1	RO	
0x10C	RX_STAT_PKT_LNGTH_0	RO	Receive Packet Length Statistics Counter. Indicated the length of the packet received. rx_statvec_o[13:0] is used to implement this counter.
0x110	RX_STAT_PKT_LNGTH_1	RO	
0x114	RX_STAT_VLN_TG_0	RO	Receive VLAN Tag Statistics Counter. Counts the total number of tagged packets received. rx_statvec_o[18] is used to implement this counter.
0x118	RX_STAT_VLN_TG_1	RO	
0x11C	RX_STAT_PAUSE_0	RO	Receive Pause Packet Statistics Counter. Counts the total number of pause packets received. rx_statvec_o[19] is used to implement this counter.
0x120	RX_STAT_PAUSE_1	RO	
0x124	RX_STAT_FLT_0	RO	Receive Filtered Packet Statistics Counter. rx_statvec_o[22] is used to implement this counter.
0x128	RX_STAT_FLT_1	RO	
0x12C	RX_STAT_UNSP_OPCODE_0	RO	Receive Unsupported Opcode Statistics Counter. Counts the number of packets received with unsupported Opcode. rx_statvec_o[23] is used to implement this counter.
0x130	RX_STAT_UNSP_OPCODE_1	RO	

Offset	Register Name	Access	Description
0x134	RX_STAT_BRDCST_0	RO	Receive Broadcast Packet Statistics Counter. Counts the number of packets received that were directed to the broadcast address. This does not include multicast packets. rx_statvec_o[15] is used to implement this counter.
0x138	RX_STAT_BRDCST_1	RO	
0x13C	RX_STAT_MULTCST_0	RO	Receive Multicast Packet Statistics Counter. Counts the number of packets received that were directed to the multicast address. This does not include broadcast packets. rx_statvec_o[16] is used to implement this counter.
0x140	RX_STAT_MULTCST_1	RO	
0x144	RX_STAT_LNGTH_ERR_0	RO	Receive Length Error Statistics Counter. Counts the total number of packets received with length of the packet and length in the Length/Type field mismatch. rx_statvec_o[20] is used to implement this counter.
0x148	RX_STAT_LNGTH_ERR_1	RO	
0x14C	RX_STAT_LNG_PKT_0	RO	Receive Long Packet Statistics Counter. Counts the number of packets received longer than the max_pkt_len. rx_statvec_o[21] is used to implement this counter.
0x150	RX_STAT_LNG_PKT_1	RO	
0x154	RX_STAT_CRC_ERR_0	RO	Receive CRC Error Statistics Counter. Counts the number of packets received with CRC error. rx_statvec_o[17] is used to implement this counter.
0x158	RX_STAT_CRC_ERR_1	RO	
0x15C	RX_STAT_PKT_DISCARD_0	RO	Receive Packet Discard Statistics Counter. Counts the number of packets discarded at the receive end. rx_statvec_o[14] is used to implement this counter.
0x160	RX_STAT_PKT_DISCARD_1	RO	
0x164	RX_STAT_PKT_IGNORE_0	RO	Receive Packet Ignored Statistics Counter. Counts the number of packets ignored when you request using the ignore_pkt. rx_statvec_o[25] is used to implement this counter.
0x168	RX_STAT_PKT_IGNORE_1	RO	
0x16C	RX_STAT_PKT_FRAGMENTS_0	RO	Receive Packet Fragments Statistics Counter. Counts the number of packets received with less than 64 octets in length and has either an FCS error or an alignment error. rx_statvec_o[13:6] along with rx_statvec[17] are used to implement this counter.
0x170	RX_STAT_PKT_FRAGMENTS_1	RO	
0x174	RX_STAT_PKT_JABBERS_0	RO	Receive Packet Jabbers Statistics Counter. Counts the number of packets received with length longer than 1,518 octets and has either an FCS error or an alignment error. rx_statvec_o[13:0] along with rx_statvec_o[17] are used to implement this counter.
0x178	RX_STAT_PKT_JABBERS_1	RO	
0x17C	RX_STAT_PKT_64_0	RO	Receive Packet 64 Statistics Counter. Counts the number of packets received that were 64 octets in length (including bad packets). rx_statvec_o[13:0] is used to implement this counter.
0x180	RX_STAT_PKT_64_1	RO	
0x184	RX_STAT_PKT_65_127_0	RO	Receive Packet 65-127 Statistics Counter. Counts the number of packets received that were between 65-127 octets in length (including bad packets).
0x188	RX_STAT_PKT_65_127_1	RO	
0x18C	RX_STAT_PKT_128_255_0	RO	Receive Packet 128-255 Statistics Counter. Counts the number of packets received that were between 128-255 octets in length (including bad packets).
0x190	RX_STAT_PKT_128_255_1	RO	

Offset	Register Name	Access	Description
0x194	RX_STAT_PKT_256_511_0	RO	Receive Packet 256-511 Statistics Counter. Counts the number of packets received that were between 256-511 octets in length (including bad packets).
0x198	RX_STAT_PKT_256_511_1	RO	
0x19C	RX_STAT_PKT_512_1023_0	RO	Receive Packet 512-1023 Statistics Counter. Counts the number of packets received that were between 512-1,023 octets in length (including bad packets).
0x1A0	RX_STAT_PKT_512_1023_1	RO	
0x1A4	RX_STAT_PKT_1024_1518_0	RO	Receive Packet 1024-1518 Statistics Counter. Counts the number of packets received that were between 1,024-1,518 octets in length (including bad packets).
0x1A8	RX_STAT_PKT_1024_1518_1	RO	
0x1AC	RX_STAT_PKT_UNDERSIZE_0	RO	Receive Packet Undersize Statistics Counter. Counts the number of packets received that were less than 64 octets long and were otherwise well formed.
0x1B0	RX_STAT_PKT_UNDERSIZE_1	RO	
0x1B4	RX_STAT_PKT_UNICAST_0	RO	Receive Packet Unicast Statistics Counter. Counts the number of good packets received that were directed to a single address.
0x1B8	RX_STAT_PKT_UNICAST_1	RO	
0x1BC	RX_STAT_PKT_RCVD_0	RO	Packets Received Statistics Counter. Counts the number of packets received (including bad packet, broadcast, and multicast packets). rx_statvec[15] and rx_statvec_o[16] are used to implement this counter.
0x1C0	RX_STAT_PKT_RCVD_1	RO	
0x1C4	RX_STAT_PKT_64_GOOD_CRC_0	RO	Receive Packet 64 with Good CRC Statistics Counter. Counts the number of packets received with length less than 64 and with a good CRC. rx_statvec_o[13:6] and rx_statvec_o[17] are used to implement this counter.
0x1C8	RX_STAT_PKT_64_GOOD_CRC_1	RO	
0x1CC	RX_STAT_PKT_1518_GOOD_CRC_0	RO	Receive Packet 1518 with Good CRC Statistics Counter. Counts the number of packets received with length more than 1,518 and with a good CRC. rx_statvec_o[13:0] and rx_statvec_o[17] are used to implement this counter.
0x1D0	RX_STAT_PKT_1518_GOOD_CRC_1	RO	
0x1D4	RX_STAT_PKT_1519_2047_0	RO	Receive Packet 1519-2047 Statistics Counter. Counts the number of packets received that were between 1,519 - 2,047 octets in length (including bad packets).
0x1D8	RX_STAT_PKT_1519_2047_1	RO	
0x1DC	RX_STAT_PKT_2048_4095_0	RO	Receive Packet 2048-4095 Statistics Counter. Counts the number of packets received that were between 2,048 – 4,095 octets in length (including bad packets).
0x1E0	RX_STAT_PKT_2048_4095_1	RO	
0x1E4	RX_STAT_PKT_4096_9216_0	RO	Receive Packet 4096-9216 Statistics Counter. Counts the number of packets received that were between 4,096 – 9,216 octets in length (including bad packets).
0x1E8	RX_STAT_PKT_4096_9216_1	RO	
0x1EC	RX_STAT_PKT_9217_16383_0	RO	Receive Packet 9217-16383 Statistics Counter. Counts the number of packets received that were between 9,217 – 16,383 octets in length (including bad packets).
0x1F0	RX_STAT_PKT_9217_16383_1	RO	
0X204	TX_STAT_PTP1588_PKT_0	RO	Transmit PTP1588 Statistics Counter. Counts the number of PTP1588 packets transmitted. The

Offset	Register Name	Access	Description
0x208	TX_STAT_PTP1588_PKT_1	RO	tx_statvec_o[26] is used to implement this counter.
0x20C	RX_STAT_PTP1588_PKT_0	RO	Receive PTP1588 Statistics Counter. Counts the number of PTP1588 packets received. The rx_statvec_o[26] is used to implement this counter.
0x210	RX_STAT_PTP1588_PKT_1	RO	
0x21C	RX_STAT_LINK_OK	RO	Receive link_ok Statistics Counter. Counts the number of link_ok bit received. Note: If you wish to use the RX link_ok statistics counter, you cannot reset the MAC signals, such as xg_rxval_o and xg_rx_blk_lock_o signals when the RX PHY is down.
0x224	TX_STAT_PKT_LNGTH_ACCU_0	RO	Transmit Accumulation Byte Statistic Counter. Count and accumulate the total packet length that will receive from AXIS interface and transmits to XGMII interface. The counter will roll over to 0 if full.
0x228	TX_STAT_PKT_LNGTH_ACCU_1	RO	
0x22C	RX_STAT_PKT_LNGTH_ACCU_0	RO	Receive Accumulation Byte Statistic Counter. Count and accumulate the total packet length that will receive from XGMII interface and transmit to AXIS interface. The counter will not increase if the packet received is discarded by MAC. The counter will roll over to 0 if full.
0x230	RX_STAT_PKT_LNGTH_ACCU_1	RO	

5.3. PHY Registers (Avant Devices)

All registers are accessed through the APB interface when the IP is configured as *PHY Only*. Access Types of each register are defined in the following table.

Table 5.22. Access Types for PHY

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access.
WO	Returns 0	Updates register value.
RW	Returns register value	Updates register value.
RW1C	Returns register value	Writing 1’b1 on register bit clears the bit to 1’b0. Writing 1’b0 on register bit is ignored.
Reserved	Returns 0	Ignores write access.

5.3.1. Configuration Registers for PHY

The following table shows the register address map for the 10GbE PHY IP core.

Table 5.23. Register Address Map for PHY

AXI4-Lite Offset	APB Offset	Register Name	Bit Width	Description
PCS Registers				
0xA000 – 0xA0FC	0x5000 – 0x507F	PCS Module Registers	16	Refer to the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).

For more information on the address mapping, refer to the PHY Management section.

5.4. PHY Registers (CertusPro-NX Devices)

This section provides detailed descriptions of 10G Ethernet PCS data registers.

The following register address map specifies the available IP core registers.

Table 5.24. Register Address Map for PHY (CertusPro-NX Devices)

APB Offset	Register Name	Bit Width	Description
0x7A00 – 0x7AFF	PMA Registers	16	Refer to the Appendix A section of the CertusPro-NX SerDes/PCS User Guide (FPGA-TN-02245).
0x7B00 – 0x7BEA	MPCS Module Registers	16	Refer to the Multi-Protocol PCS Module User Guide (FPGA-IPUG-02118).

5.5. MAC + PHY+ 1588 Registers (CertusPro-NX Devices)

This section provides detailed descriptions of the MAC + PHY + 1588 registers.

5.5.1. Register Address Mapping

The following table lists the address decoding for the MAC, PHY, and TSU blocks. The base address must be adjusted accordingly to access each individual registers in the respective blocks.

Table 5.25. Register Address Mapping

Block	APB Base Address
MAC	0x0000 – 0x3FFF
PHY	0x5000 onwards
TSU	0x4000 – 0x4FFF

For example, to write to the TSU block Debug register (offset = 0x0), you need to drive the Address = 0x4000 with the desired DATA. To read from the TSU Debug register, you need to read from Address = 0x4000.

5.5.2. TSU Configuration Registers

TSU registers are included in the MAC. Address decoding is needed for the TSU block configuration.

Table 5.26. TSU Configuration Registers

Offset	Register Name	Mode	Description
0x000	Debug Register	RW	Used as a debug register.
0x004	Version Number Register	RO	Version number.
0x008	PTP RX Packet Counter Register	RO	This register is used to count the number of RX PTP packets received.
0x00C	PTP TX Packet Counter Register	RO	This register is used to count the number of TX PTP packets transmitted.
0x010	Non-PTP RX Packet Counter Register	RO	This register is used to count the number of non-PTP RX packets received.
0x014	Non-PTP TX Packet Counter Register	RO	This register is used to count the number of non-PTP TX packets transmitted.
0x018	Egress Delay Register	RW	This register is used to configure TX transceiver delay.
0x01C	Ingress Delay Register	RW	This register is used to configure RX transceiver delay.
0x02C	RX PTP Timestamp Buffer Register	RO	This register is used to keep track of the number of PTP packets in the RX timestamp buffer.
0x030	TX PTP Timestamp Buffer Register	RO	This register is used to keep track of the number of PTP packets in the TX timestamp buffer.
0x038	MAC Address 0 Register	RW	This register is used to configure MAC Address Word 0. This MAC address is used by TSU to identify the PTP packet. MAC_ADDR_0 and MAC_ADDR_1 Register

Offset	Register Name	Mode	Description
0x03C	MAC Address 1 Register	RW	This register is used to configure MAC Address Word 1. This MAC address is used by TSU to identify the PTP packet. MAC_ADDR_0 and MAC_ADDR_1 Register
0x040	Packet Reset Register	RW	This register is used to reset all packet counters. Therefore, the following registers are reset: • PTP RX Packet Counter Register (0x008) • PTP TX Packet Counter Register (0x00C) • Non-PTP RX Packet Counter Register (0x010) • Non-PTP TX Packet Counter Register (0x014)
0x044	Timestamp Buffer Register	RO	This register is used to report the TX/RX timestamp and residence buffer status.

Table 5.27. [0x0000] Debug Register

Field	Name	Description	Access	Default
[31:0]	Debug_reg	Used as debugging purpose.	RW	32'h28101997

Table 5.28. [0x0004] Version Number Register

Field	Name	Description	Access	Default
[31:0]	Version_reg	Version number	RO	32'h00000001

Table 5.29. [0x0008] PTP RX Packet Counter Register

Field	Name	Description	Access	Default
[31:0]	rx_ptp_packets_count _reg	PTP RX Packet Counter—Number of PTP packets received.	RO	32'h0

Table 5.30. [0x000C] PTP TX Packet Counter Register

Field	Name	Description	Access	Default
[31:0]	tx_ptp_packets_count _reg	PTP TX Packet Counter—Number of PTP packets transmitted.	RO	32'h0

Table 5.31 [0x0010] Non-PTP RX Packet Counter Register

Field	Name	Description	Access	Default
[31:0]	rx_non_ptp_packets_count_reg	Non-PTP RX Packet Counter—Number of non-PTP packets received.	RO	32'h0

Table 5.32 [0x0014] Non-PTP TX Packet Counter Register

Field	Name	Description	Access	Default
[31:0]	tx_non_ptp_packets_count _reg	Non-PTP TX Packet Counter—Number of non-PTP packets transmitted.	RO	32'h0

Table 5.33 [0x0018] Egress Delay Register

Field	Name	Description	Access	Default
[31:0]	egress_delay_reg	Egress Delay—The delay between the timestamping point of TX packet parser and the TX (out-going) serial pad. For more information, refer to the PHY Delay Value section.	RW	32'h0

Table 5.34 [0x001C] Ingress Delay Register

Field	Name	Description	Access	Default
[31:0]	Ingress_delay_reg	Ingress Delay—The delay between the timestamping point of RX packet parser and the RX (incoming) SFP serial pad. For more information, refer to the PHY Delay Value section.	RW	32'h0

Table 5.35 [0x002C] RX PTP Timestamp Buffer Register

Field	Name	Description	Access	Default
[31:0]	rx_timestamp_buffer_data_count_reg	Number of PTP packets in RX timestamp buffer.	RO	32'h0

Table 5.36 [0x0030] TX PTP Timestamp Buffer Register

Field	Name	Description	Access	Default
[31:0]	tx_timestamp_buffer_data_count_reg	Number of PTP packets in TX timestamp buffer.	RO	32'h0

Table 5.37 [0x0038] MAC Address 0 Register

Field	Name	Description	Access	Default
[31:0]	mac_addr_low_reg	MAC Address 0. Used by the TSU to identify the PTP packet. If the source address does not match with the MAC address, then it will not be considered as a PTP packet and is treated as a normal packet. This register has the same programming sequence as the MAC_ADDR_0(0x014) and MAC_ADDR_1(0x018) registers in the MAC_ADDR_0 and MAC_ADDR_1 Register section. For example: MAC_ADDR: AC-DE-48-00-00-80. You must assign MAC address 0[31:0] = 48-00-00-80.	RW	32'h0

Table 5.38 [0x003C] MAC Address 1 Register

Field	Name	Description	Access	Default
[31:16]	Reserved	—	RO	16’h00
[15:0]	mac_addr_high_reg	MAC Address 1. Used by the TSU to identify the PTP packet. If the source address does not match with the MAC address, then it will not be considered as a PTP packet and is treated as a normal packet. This register has the same programming sequence as the MAC_ADDR_0(0x014) and MAC_ADDR_1(0x018) registers in the MAC_ADDR_0 and MAC_ADDR_1 Register section. For example: MAC_ADDR: AC-DE-48-00-00-80. You must assign MAC address 1[15:0] =	RO	32'h0

Field	Name	Description	Access	Default
		AC-DE.		

Table 5.39 [0x0040] Packet Reset Register

Field	Name	Description	Access	Default
[31:1]	Reserved	—	RO	31’h0
[0]	Packet_reset_reg	Parser Packet Count Reset. This resets both PTP [TX,RX] Packet Counter and Non-PTP [TX,RX] Packet Counter. 0—No reset. 1—Reset.	RW	1'b0

Table 5.40 [0x0044] Timestamp Buffer Register

Field	Name	Description	Access	Default
[31:8]	Reserved	—	RO	24’h0
[7]	rx_timestamp_buffer_empty_reg	RX timestamp buffer empty status.	RO	1'b0
[6]	rx_timestamp_buffer_full_reg	RX timestamp buffer full status.	RO	1'b0
[5]	rx_residence_buffer_empty_reg	RX residence buffer empty status.	RO	1'b0
[4]	rx_residence_buffer_full_reg	RX residence buffer full status.	RO	1'b0
[3]	tx_timestamp_buffer_empty_reg	TX timestamp buffer empty status.	RO	1'b0
[2]	tx_timestamp_buffer_full_reg	TX timestamp buffer full status.	RO	1'b0
[1]	tx_residence_buffer_full_reg	TX residence buffer full status.	RO	1'b0
[0]	Tx_residence_buffer_empty_reg	TX residence buffer empty status.	RO	1'b0

6. Example Design

The following example designs are generated along with the 10G Ethernet MAC IP core in <Component Name>/eval folder:

- Eval example design (Certus-Pro NX devices)
  This example design integrates the 10G Ethernet MAC IP core with a pre-generated CertuxPro-NX 10G Ethernet PHY IP core and GPLL into an Ethernet sub-system. You can simulate the example design using the QuestaSim™ software. For more information, refer to the Eval Example Design Components (CertusPro-NX Devices) section.
- Versa example design (Avant devices and CertusPro-NX devices)
  The versa example design integrates the 10G Ethernet MAC and Avant or CertusPro-NX PHY IP core with the AXI4-Lite and data checker into an Ethernet sub-system. The 10G Ethernet MAC + PHY IP Versa example design allows you to compile, simulate, and test the 10G Ethernet MAC + PHY IP on the following Lattice evaluation boards:
  - Avant G/X Versa board
  - CertusPro-NX Versa board

  For more information, refer to the Versa Example Design Components (Avant Devices and CertusPro-NX Devices) section.

  For more information on testing the IP with the evaluation board, refer to the Hardware Testing section.

6.1. Example Design Configuration

The following table shows the IP configuration for the 10G Ethernet MAC IP example design.

Table 6.1. IP Configuration for the 10G Ethernet MAC IP Example Design

GUI Parameter	IP Configuration
PHY Interface	XGMII
Dynamic Speed Selection	Disabled
Multicast Address Filtering	Don’t Care
Statistics Counters Registers	Don’t Care
Frame Type	Don’t Care

6.2. Overview of Example Design and Features

Key features of the 10G Ethernet MAC IP example design include:

- Ethernet packet generator to generate and compare packets
- Integrated 10G Ethernet MAC with 10G Ethernet PCS and GPLL to demonstrate the expected clocking scheme
- Loopback transmitted packet at the serial interface
- Supports continuous and non-continuous mode packet transmission (Versa example design only)

6.3. Example Design Components

6.3.1. Eval Example Design Components (CertusPro-NX Devices)

The 10G Ethernet MAC IP eval example design includes the following blocks:

- 10 Gb Ethernet MAC core
- 10 Gb Ethernet PCS core
- GPLL
- APB driver
- AXI-Stream driver
- Data checker

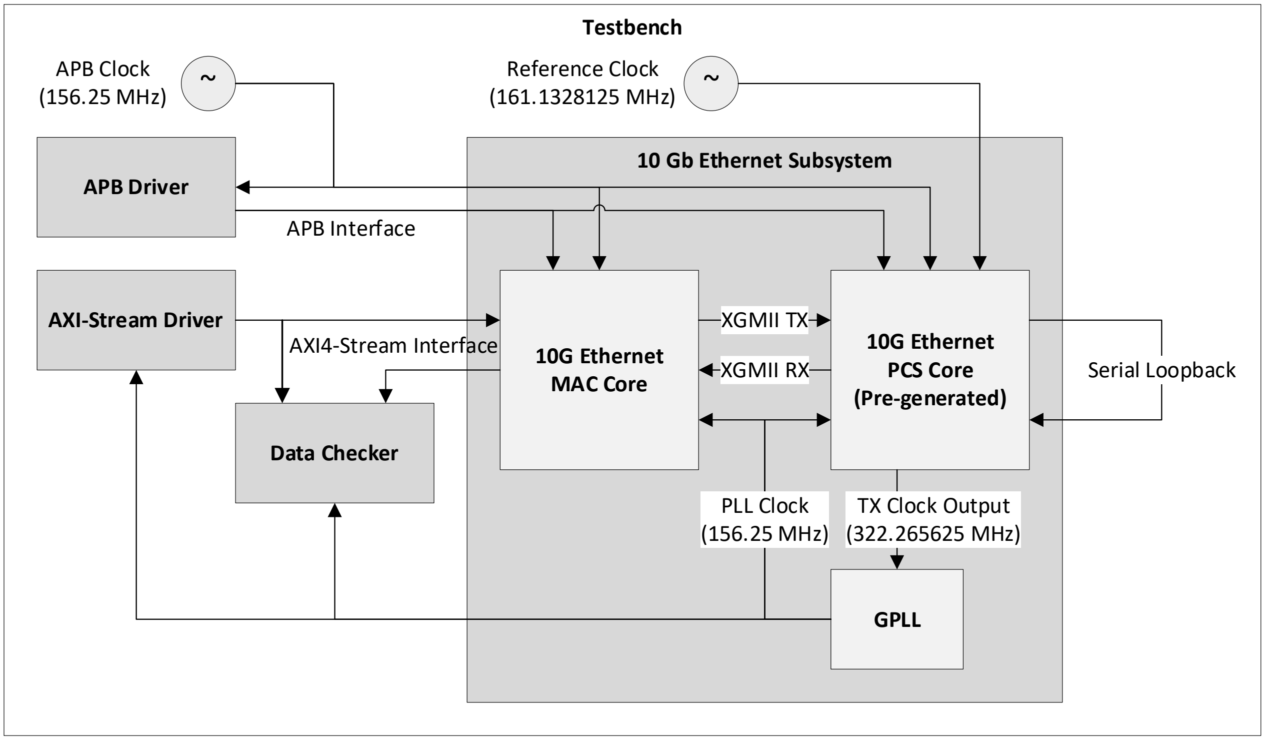


Figure 6.1 10 Gb Ethernet MAC IP Eval Example Design Block Diagram

6.3.1.1. 10G Ethernet MAC Core

The 10G Ethernet MAC core block instantiates 10G Ethernet MAC IP core with selected configuration. In transmit data path, it receives packets from the AXI-Stream driver and sends them to the 10G Ethernet PCS. In the receiving data path, it forwards packets from the 10G Ethernet PCS core to the data checker.

6.3.1.2. 10G Ethernet PCS Core

The 10G Ethernet PCS core block instantiates the CertusPro-NX 10G Ethernet PHY IP core with pre-generated configurations. In transmit data path, it sends packets from the 10G Ethernet MAC IP core to serial interface signals. In receive data path, it forwards received packet at serial interface signals to the 10G Ethernet MAC IP core. For more information on the features of the CertusPro-NX 10G Ethernet PHY IP core, refer to the CertusPro-NX 10G Ethernet PHY IP Core User Guide (FPGA-IPUG-02163).

6.3.1.3. GPLL

The GPLL receives 322.265625 MHz input clock from the 10G Ethernet PCS core and generates 156.25 MHz output clock to drive transmit and receive data path logics. The GPLL must have Fractional-N Divider mode disabled.

6.3.1.4. APB Driver

The APB driver is an example design component that configures the register of the 10G Ethernet MAC as shown the following table to demonstrate usage of the APB interface of the 10G Ethernet MAC. You must modify the registers and their value to match with your target application.

Table 6.2. APB Driver Configuration Registers

10G Ethernet MAC Register Address Offset	10G Ethernet MAC Register Name	Configuration Value
0x000	MODE	0x3
0x004	TX_CTL	0x0
0x008	RX_CTL	0x1
0x010	IPG_VAL	0x10

6.3.1.5. AXI-Stream Driver

The AXI-Stream driver generates packets for the AXI-Stream Transmit Interface of the 10G Ethernet MAC. The contents of the packets are randomized.

6.3.1.6. Data Checker

The Data checker monitors generated packets from the AXI-Stream driver and compared them with packets from the AXI-Stream Receive Interface of the 10G Ethernet MAC core. An error signal is flagged if the packets are mismatched.

6.3.2. Versa Example Design Components (Avant Devices and CertusPro-NX Devices)

The 10G Ethernet MAC IP versa example design includes the following blocks:

- 10 Gb Ethernet MAC + PHY IP core
- AXI4-Lite module (Avant devices only)
- APB module (CertusPro-NX devices only)
- AXI-Stream driver
- Data checker

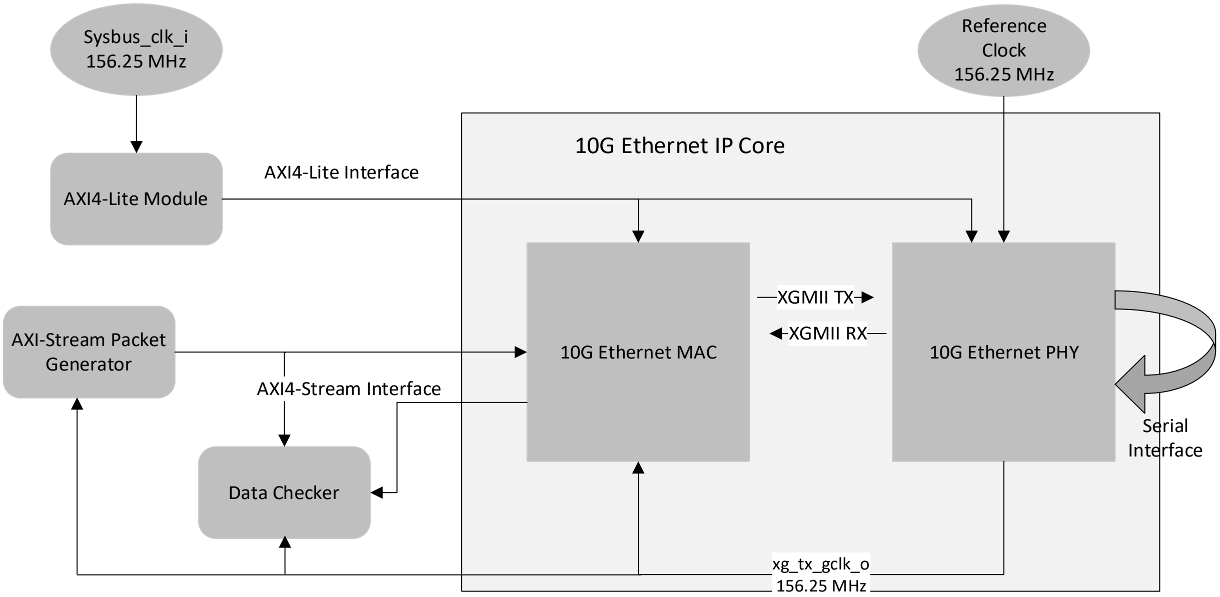


Figure 6.2 10 Gb Ethernet MAC IP Versa Example Design Block Diagram

6.3.2.1. 10 Gb Ethernet MAC + PHY IP Core

The 10G Ethernet MAC instantiates the 10G Ethernet MAC IP core based on the configuration value described in Table 6.3. In the transmit data path, the 10G Ethernet MAC receives packets from AXI-Stream Packet Generator and sends the packets to the 10G Ethernet PHY. In the receive data path, the 10G Ethernet MAC forwards packets from the 10G Ethernet PHY core to the Data Checker. The 10G Ethernet MAC is connected to the 10G Ethernet PHY within the 10G Ethernet IP core through the XGMII interface. The clock source to the Ethernet MAC is from xg_tx_gclk_o, which is the clock output from the Ethernet PHY. In the transmit data path, the 10G Ethernet PHY sends packets from 10G Ethernet MAC core to serial interface signals. In the receive data path, it forwards the received packet at serial interface signals to the 10G Ethernet MAC core.

6.3.2.2. AXI4-Lite Module or APB Module

The AXI4-Lite module (Avant devices) or APB module (CertusPro-NX devices) is the example design component that configures the register of the 10G Ethernet MAC. The example design component demonstrates the usage of the AXI4-Lite interface or APB interface of the 10G Ethernet and initializes the MAC to enable transmit and receive data path. You must modify the registers and the values to match your target application.

Table 6.3. AXI4-Lite Module or APB Module Configuration Registers

10 Gb Ethernet MAC Register Address Offset	10 Gb Ethernet MAC Register Name	Configuration Value
0x000	MODE	0x3
0x004	TX_CTL	0x0
0x008	RX_CTL	0x1
0x010	IPG_VAL	0x10

6.3.2.3. AXI-Stream Packet Generator

The AXI-Stream Packet Generator generates packets for the AXI-Stream Transmit Interface of the 10G Ethernet MAC. The contents of the packets are randomized.

6.3.2.4. Data Checker

The Data Checker monitors the generated packets from the AXI-Stream Packet Generator and compares them with the packets from the AXI-Stream Receive Interface of the 10G Ethernet MAC core. An error signal is flagged if the packets are mismatched.

6.4. Generating Example Design

6.4.1. Create New Radiant Project

1. In the Radiant software, go to **File → New → Project…** or click on the **New Project** icon.

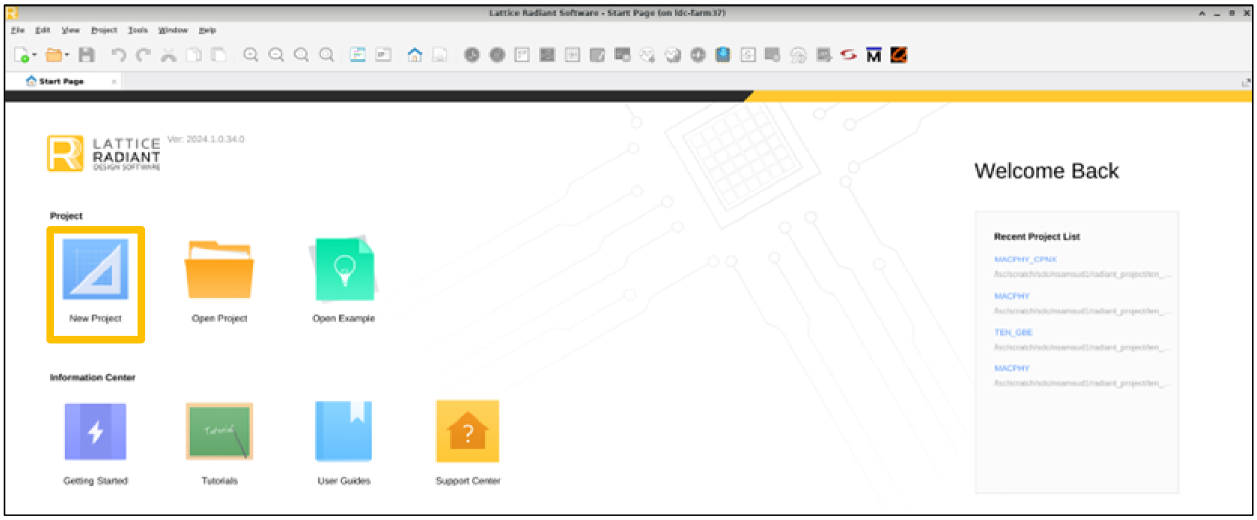
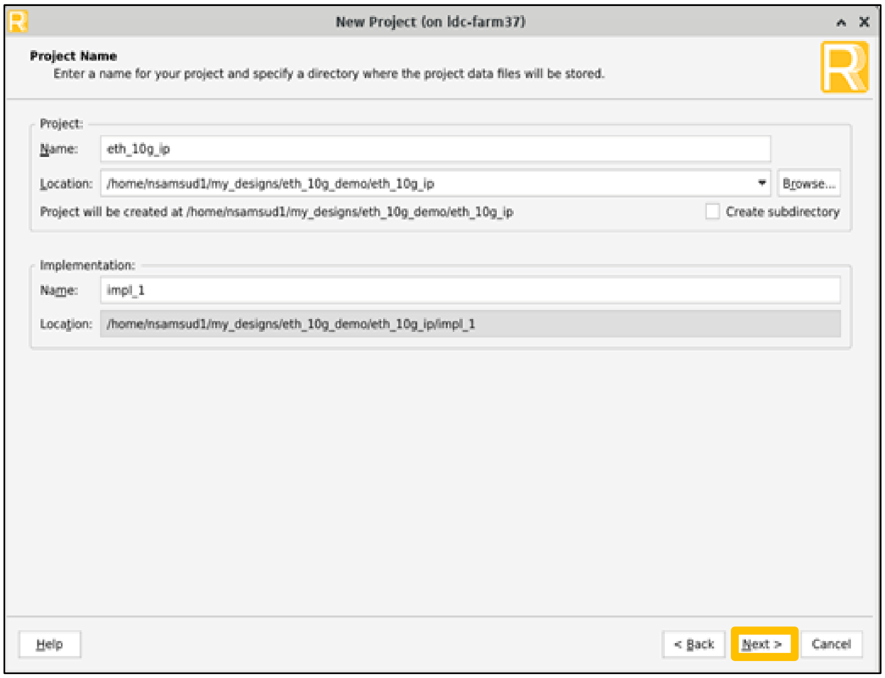


Figure 6.3. Start Page of the Radiant Software

2. Enter the project **Name** and **Location**, and click **Next**.

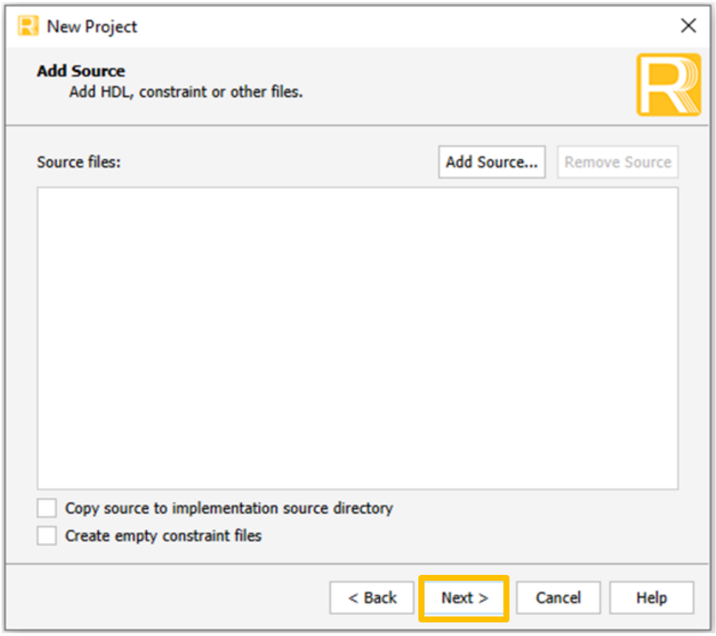


Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

Project:
Name: eth_10g_ip
Location: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip
Project will be created at /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip
□ Create subdirectory

Implementation:
Name: impl_1
Location: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/impl_1

Figure 6.4. Create a New Project



New Project

Add Source
Add HDL, constraint or other files.

Source files:

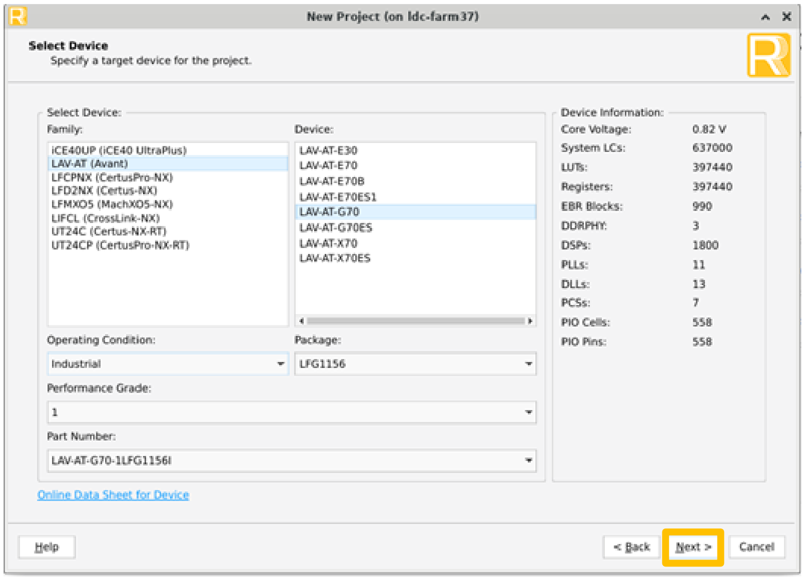
□ Copy source to implementation source directory

□ Create empty constraint files

Figure 6.5. Add Source to Project

3. In the **Select Device** window, follow these steps:

   a. Select **LAV-AT (Avant)** family **→LAV-AT-X70 or G70** device, then click **Next.**



New Project (on ldc-farm37)

Select Device
Specify a target device for the project.

Select Device:

Family:
iCE40UP (iCE40 UltraPlus)
LAV-AT (Avant)
LFCPNX (CertusPro-NX)
LFD2NX (Certus-NX)
LFMXO5 (MachXO5-NX)
LIFCL (CrossLink-NX)
UT24C (Certus-NX-RT)
UT24CP (CertusPro-NX-RT)

Device:
LAV-AT-E30
LAV-AT-E70
LAV-AT-E70B
LAV-AT-E70ES1
LAV-AT-G70
LAV-AT-G70ES
LAV-AT-X70
LAV-AT-X70ES

Operating Condition: Industrial

Package: LFG1156

Performance Grade: 1

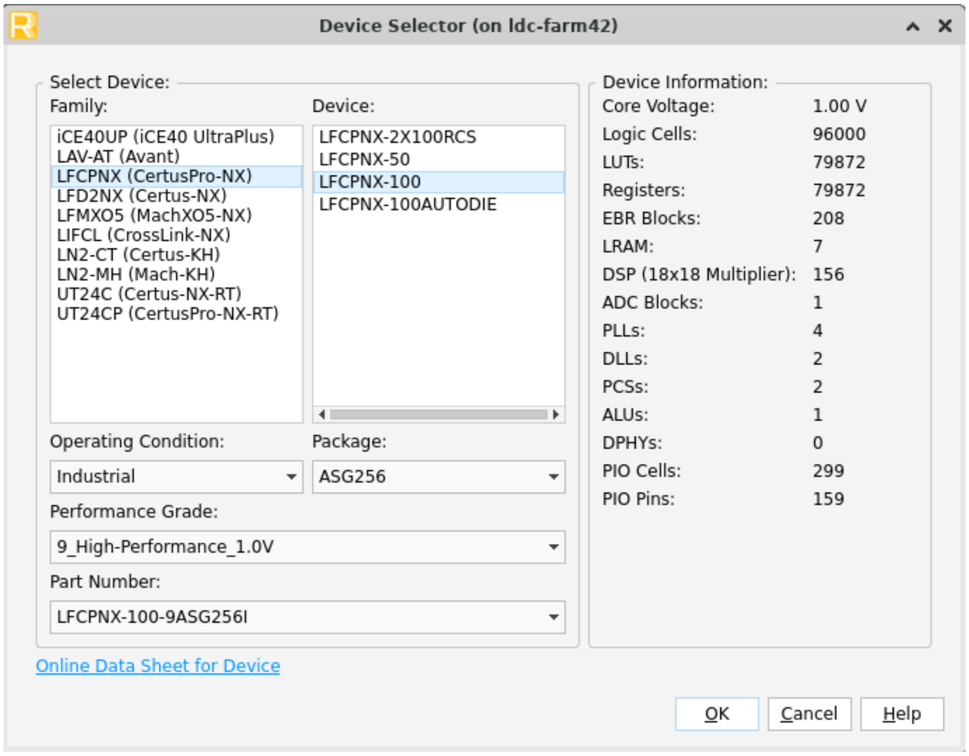
Part Number: LAV-AT-G70-1LFG1156I

Device Information:

Core Voltage:	0.82 V
System LCs:	637000
LUTs:	397440
Registers:	397440
EBR Blocks:	990
DDRPHY:	3
DSPs:	1800
PLLs:	11
DLLs:	13
PCSs:	7
PIO Cells:	558
PIO Pins:	558

Figure 6.6. Select an Avant Device for the Project

b. Select **LFCPNX (CertusPro-NX)** family **→LFCPNX-100**, then click **Next.**



Device Selector (on ldc-farm42)

Select Device:

Family:
iCE40UP (iCE40 UltraPlus)
LAV-AT (Avant)
LFCPNX (CertusPro-NX)
LFD2NX (Certus-NX)
LFMXO5 (MachXO5-NX)
LIFCL (CrossLink-NX)
LN2-CT (Certus-KH)
LN2-MH (Mach-KH)
UT24C (Certus-NX-RT)
UT24CP (CertusPro-NX-RT)

Device:
LFCPNX-2X100RCS
LFCPNX-50
LFCPNX-100
LFCPNX-100AUTODIE

Operating Condition: Industrial

Package: ASG256

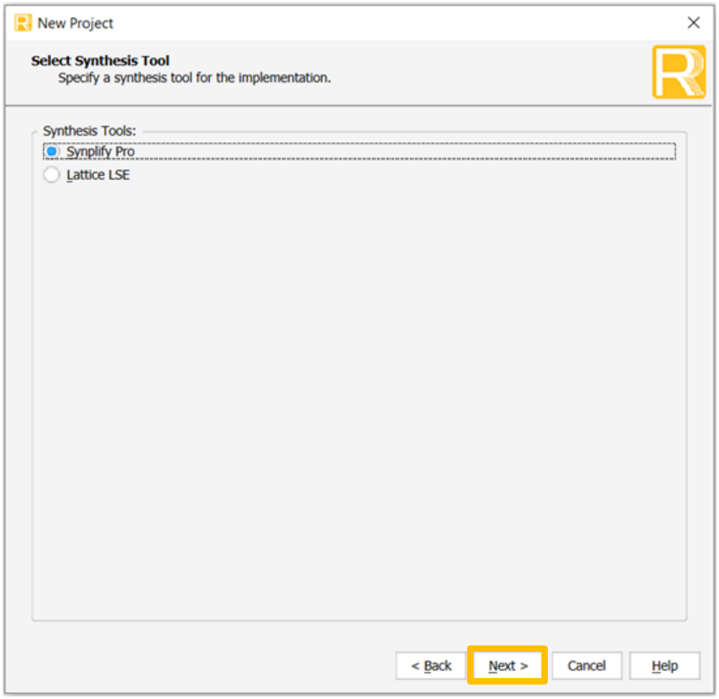
Performance Grade: 9_High-Performance_1.0V

Part Number: LFCPNX-100-9ASG256I

Device Information:	
Core Voltage:	1.00 V
Logic Cells:	96000
LUTs:	79872
Registers:	79872
EBR Blocks:	208
LRAM:	7
DSP (18x18 Multiplier):	156
ADC Blocks:	1
PLLs:	4
DLLs:	2
PCSs:	2
ALUs:	1
DPHYs:	0
PIO Cells:	299
PIO Pins:	159

Figure 6.7. Select an CPNX Device for the Project

4. In the **Select Synthesis Tool** window, click **Next**.



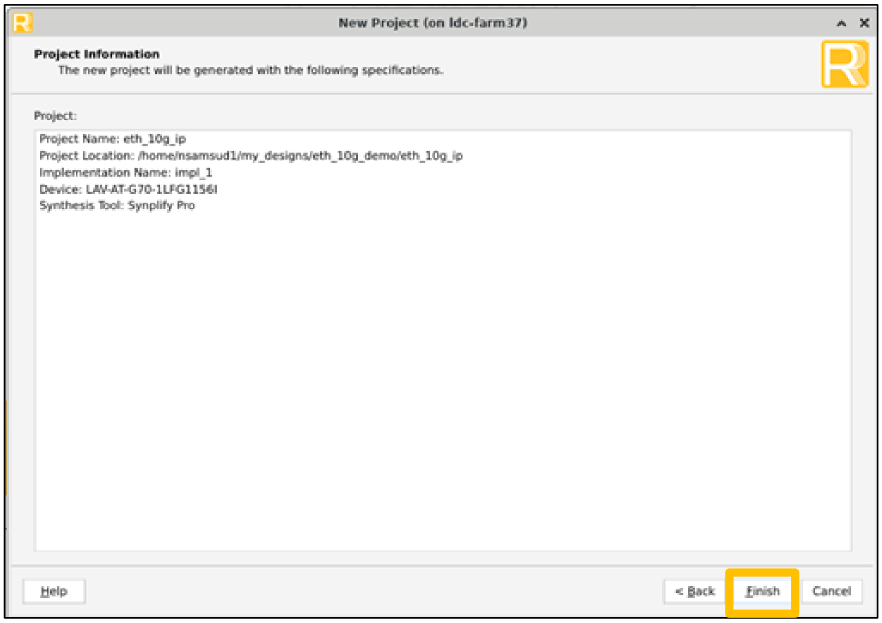
New Project

Select Synthesis Tool
Specify a synthesis tool for the implementation.

Synthesis Tools:
● Synplify Pro
○ Lattice LSE

Figure 6.8. Select a Synthesis Tool

5. The Project Information window lists the specifications of the new project. Click **Finish**.



New Project (on ldc-farm37)

Project Information

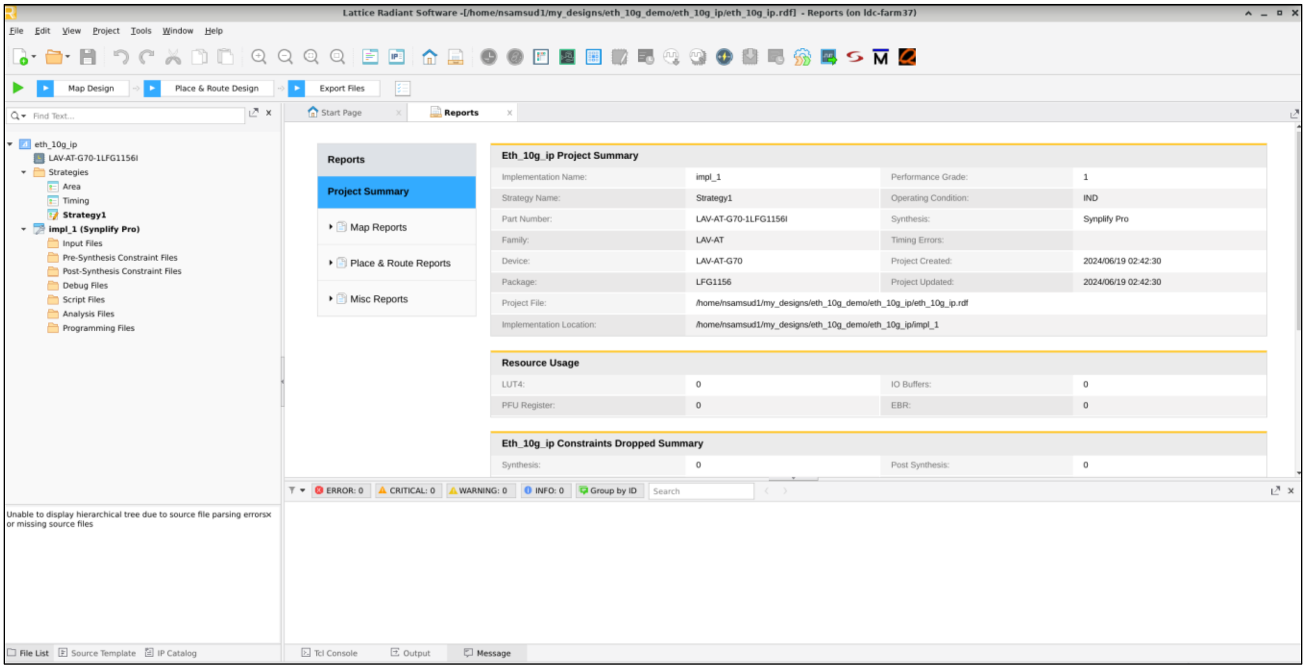
The new project will be generated with the following specifications.

Project:

```
Project Name: eth_10g_ip
Project Location: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip
Implementation Name: impl_1
Device: LAV-AT-G70-1LFG1156I
Synthesis Tool: Synplify Pro
```

Figure 6.9. Project Information

6. The new Radiant project is generated. View the project summary under the **Project Summary** tab.



Lattice Radiant Software -[/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip.rdf] - Reports (on ldc-farm37)

Unable to display hierarchical tree due to source file parsing errorsx or missing source files

Eth_10g_ip Project Summary

Implementation Name:	impl_1	Performance Grade:	1
Strategy Name:	Strategy1	Operating Condition:	IND
Part Number:	LAV-AT-G70-1LFG1156I	Synthesis:	Synplify Pro
Family:	LAV-AT	Timing Errors:	
Device:	LAV-AT-G70	Project Created:	2024/06/19 02:42:30
Package:	LFG1156	Project Updated:	2024/06/19 02:42:30
Project File:	/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip.rdf		
Implementation Location:	/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/impl_1		

Resource Usage

LUT4:	0	IO Buffers:	0
PFU Register:	0	EBR:	0

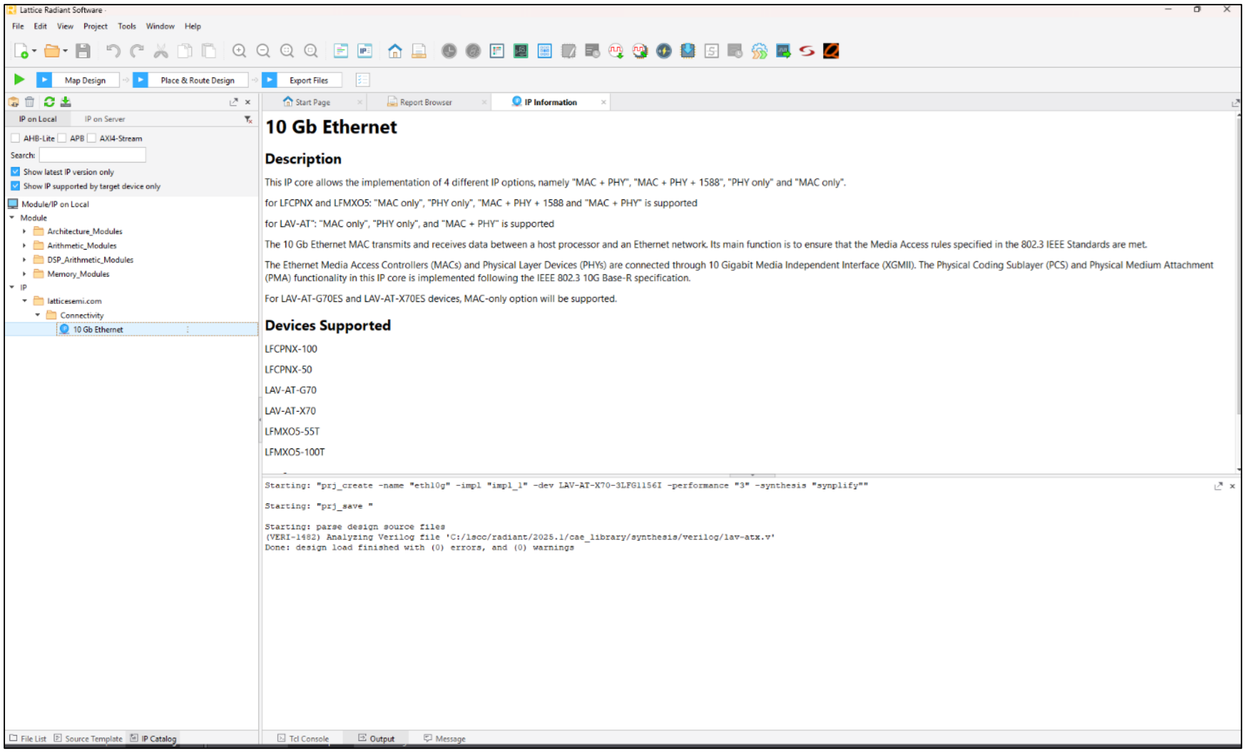
Eth_10g_ip Constraints Dropped Summary

Synthesis:	0	Post Synthesis:	0

Figure 6.10. A Radiant Software Project is Created

6.4.2. IP Installation and Generation

1. Go to **IP Catalog → IP on Server** tab. Under the **Connectivity** category, right-click on the **10 Gb Ethernet IP** and select **Download** to install the IP. Alternatively, you can click on the ⬇ icon to start the IP installation. If you already have the 10G Ethernet IP installed, proceed to step 2.



□ AHB-Lite □ APB □ AXI4-Stream

Search:

☑ Show latest IP version only

☑ Show IP supported by target device only

- Module/IP on Local
  - Module
    - Architecture_Modules
    - Arithmetic_Modules
    - DSP_Arithmetic_Modules
    - Memory_Modules
  - IP
    - latticesemi.com
      - Connectivity
        - 10 Gb Ethernet

10 Gb Ethernet

Description

This IP core allows the implementation of 4 different IP options, namely "MAC + PHY", "MAC + PHY + 1588", "PHY only" and "MAC only".

for LFCPNX and LFMXO5: "MAC only", "PHY only", "MAC + PHY + 1588 and "MAC + PHY" is supported

for LAV-AT": "MAC only", "PHY only", and "MAC + PHY" is supported

The 10 Gb Ethernet MAC transmits and receives data between a host processor and an Ethernet network. Its main function is to ensure that the Media Access rules specified in the 802.3 IEEE Standards are met.

The Ethernet Media Access Controllers (MACs) and Physical Layer Devices (PHYs) are connected through 10 Gigabit Media Independent Interface (XGMII). The Physical Coding Sublayer (PCS) and Physical Medium Attachment (PMA) functionality in this IP core is implemented following the IEEE 802.3 10G Base-R specification.

For LAV-AT-G70ES and LAV-AT-X70ES devices, MAC-only option will be supported.

Devices Supported

LFCPNX-100

LFCPNX-50

LAV-AT-G70

LAV-AT-X70

LFMXO5-55T

LFMXO5-100T

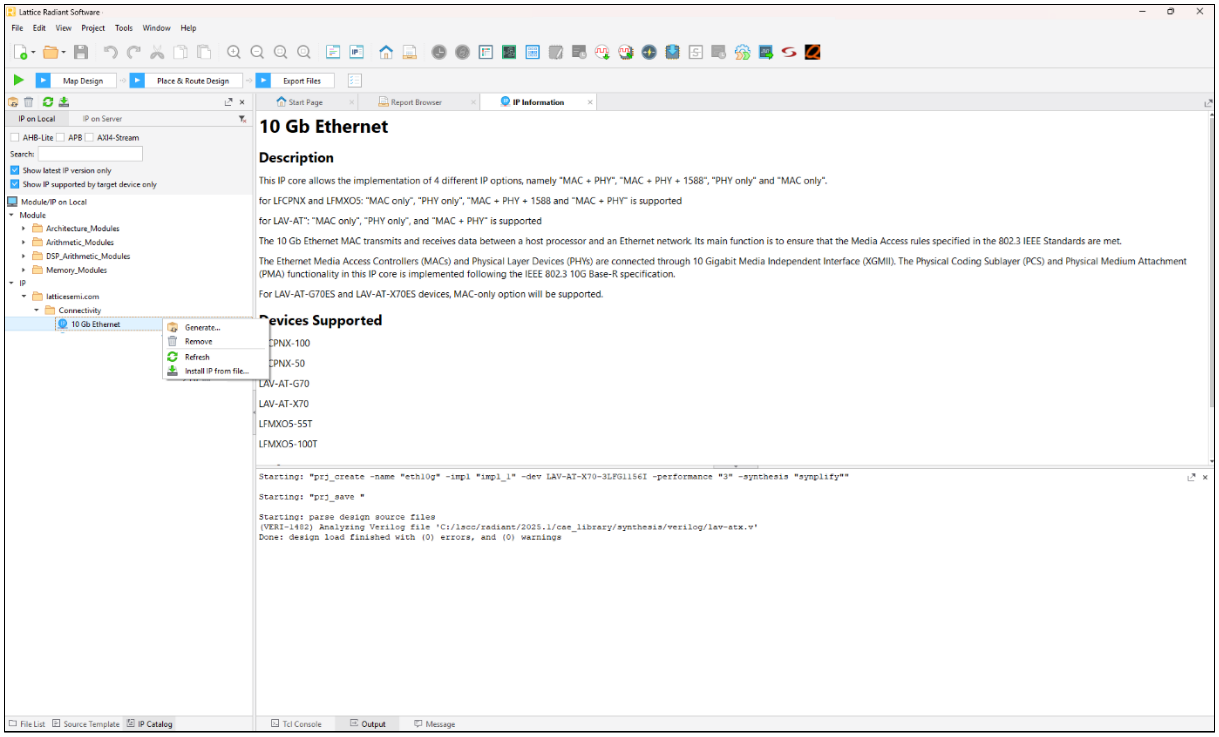
```
Starting: "prj_create -name "eth10g" -impl "impl_1" -dev LAV-AT-X70-3LFG1156I -performance "3" -synthesis "synplify""

Starting: "prj_save "

Starting: parse design source files
(VERI-1482) Analyzing Verilog file 'C:/lscc/radiant/2025.1/cae_library/synthesis/verilog/lav-atx.v'
Done: design load finished with (0) errors, and (0) warnings
```

Figure 6.11. IP Catalog

2. Go to the **IP on Local** tab. Right-click on the 10 Gb Ethernet IP and select **Generate…** to launch the IP GUI.



10 Gb Ethernet

Description

This IP core allows the implementation of 4 different IP options, namely "MAC + PHY", "MAC + PHY + 1588", "PHY only" and "MAC only".

for LFCPNX and LFMXO5: "MAC only", "PHY only", "MAC + PHY + 1588 and "MAC + PHY" is supported

for LAV-AT": "MAC only", "PHY only", and "MAC + PHY" is supported

The 10 Gb Ethernet MAC transmits and receives data between a host processor and an Ethernet network. Its main function is to ensure that the Media Access rules specified in the 802.3 IEEE Standards are met.

The Ethernet Media Access Controllers (MACs) and Physical Layer Devices (PHYs) are connected through 10 Gigabit Media Independent Interface (XGMII). The Physical Coding Sublayer (PCS) and Physical Medium Attachment (PMA) functionality in this IP core is implemented following the IEEE 802.3 10G Base-R specification.

For LAV-AT-G70ES and LAV-AT-X70ES devices, MAC-only option will be supported.

Devices Supported

CPNX-100

CPNX-50

LAV-AT-G70

LAV-AT-X70

LFMXO5-55T

LFMXO5-100T

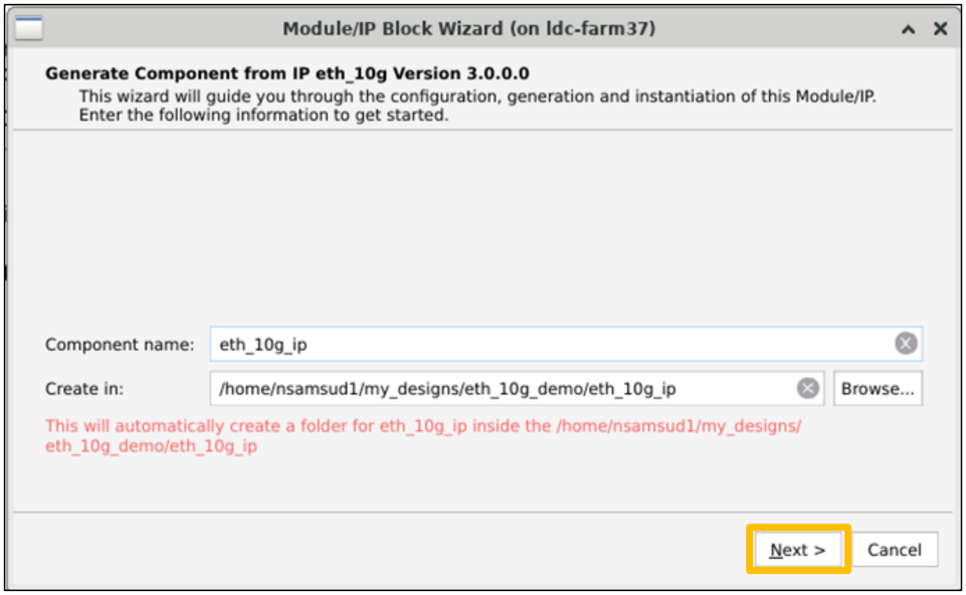
```
Starting: "prj_create -name "eth10g" -impl "impl_1" -dev LAV-AT-X70-3LFG1156I -performance "3" -synthesis "synplify""

Starting: "prj_save "

Starting: parse design source files
(VERI-1482) Analyzing Verilog file 'C:/lscc/radiant/2025.1/cae_library/synthesis/verilog/lav-atx.v'
Done: design load finished with (0) errors, and (0) warnings
```

Figure 6.12. IP Generation

3. Enter **Component Name** and click **Next**.



Generate Component from IP eth_10g Version 3.0.0.0

This wizard will guide you through the configuration, generation and instantiation of this Module/IP.
Enter the following information to get started.

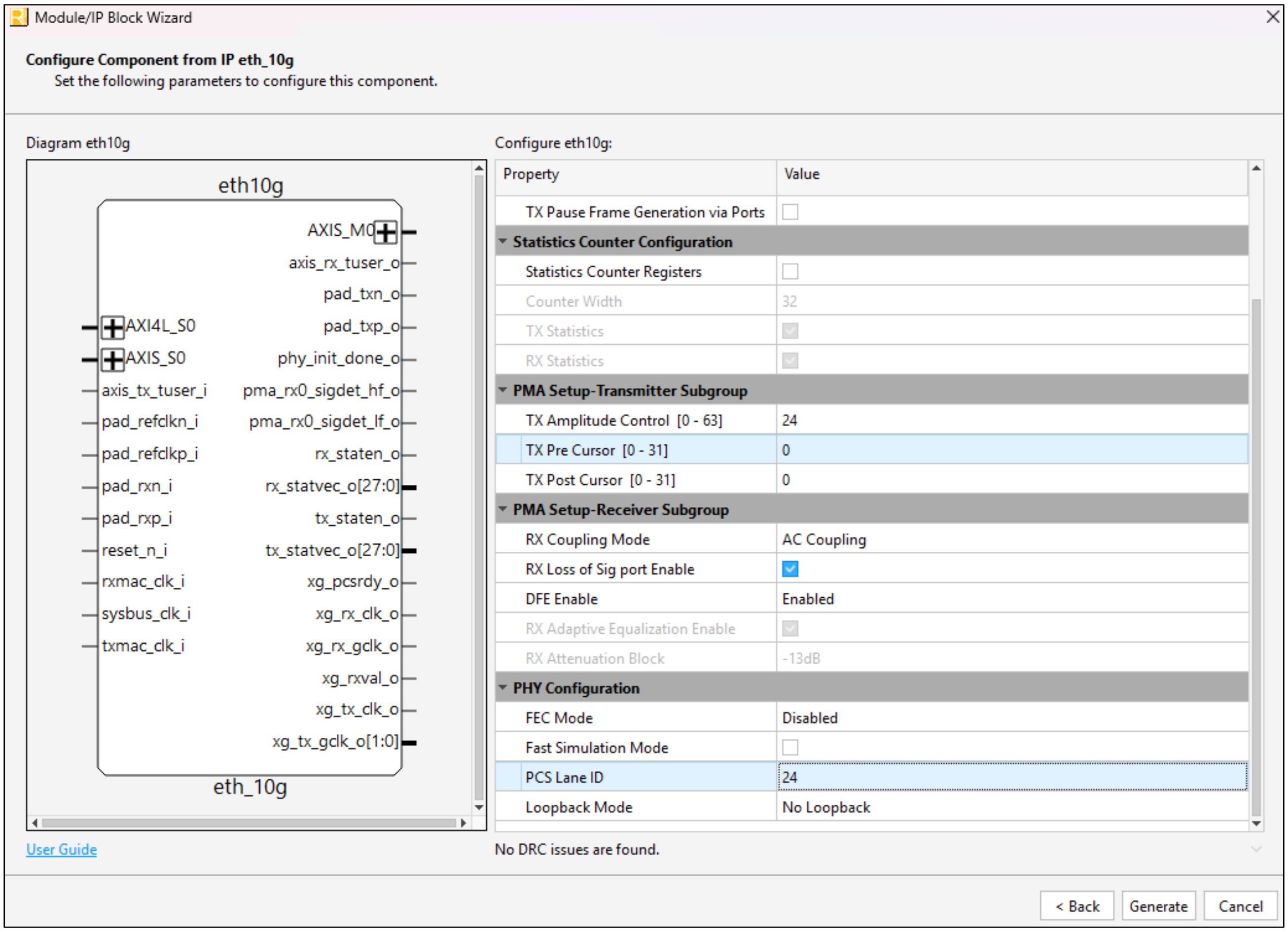
Component name: eth_10g_ip

Create in: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

This will automatically create a folder for eth_10g_ip inside the /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

Figure 6.13. Generate Component

4. The following window allows you to change the IP configuration. Set the **PCS Lane ID** to 24 for Avant G/X devices and set the **PCS Lane ID** to 6 for CertusPro-NX devices. Apply the default settings for the remaining IP configurations, and click **Generate**.



Module/IP Block Wizard

Configure Component from IP eth_10g

Set the following parameters to configure this component.

Diagram eth10g

Configure eth10g:

Property	Value
TX Pause Frame Generation via Ports	☐
Statistics Counter Configuration	
Statistics Counter Registers	☐
Counter Width	32
TX Statistics	☑
RX Statistics	☑
PMA Setup-Transmitter Subgroup	
TX Amplitude Control [0 - 63]	24
TX Pre Cursor [0 - 31]	0
TX Post Cursor [0 - 31]	0
PMA Setup-Receiver Subgroup	
RX Coupling Mode	AC Coupling
RX Loss of Sig port Enable	☑
DFE Enable	Enabled
RX Adaptive Equalization Enable	☑
RX Attenuation Block	-13dB
PHY Configuration	
FEC Mode	Disabled
Fast Simulation Mode	☐
PCS Lane ID	24
Loopback Mode	No Loopback

User Guide

No DRC issues are found.

Figure 6.14. Configure Component

Note:

* Applicable for LAV-AT-G70 and LAV-AT-X70.

For PHY configuration, verify that the selected PCS Lane ID corresponds to the actual serial connection on board. For onboard connectivity details, refer to the CertusPro-NX Versa Board User Guide or Avant-G/X Versa Board User Guide.

6.4.3. Importing Example Design Files to a Project

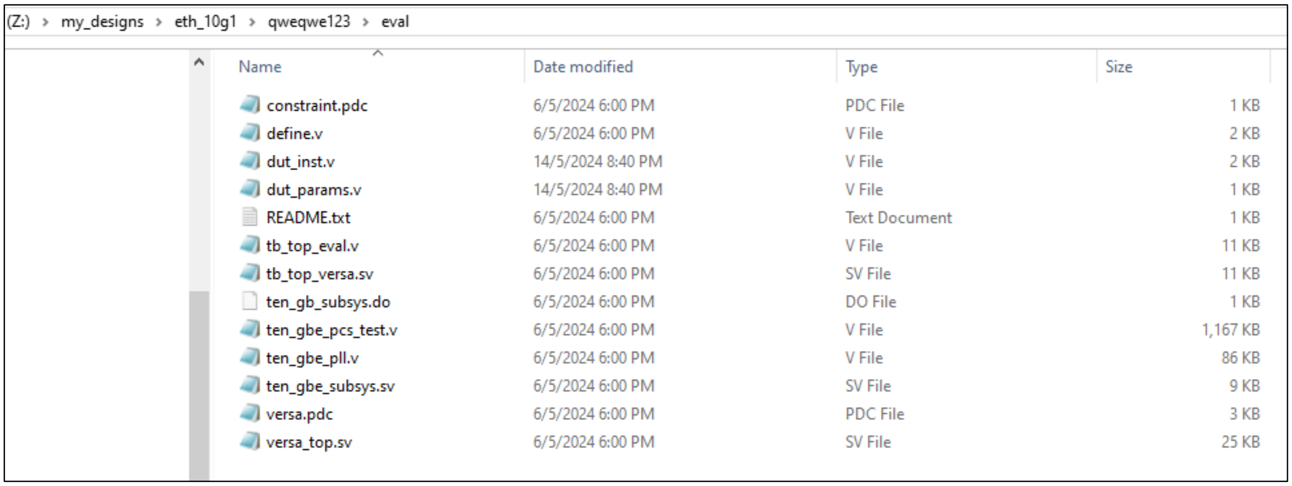
The 10G Ethernet IP generates two sets of example design files in the eval directory of the IP.

Table 6.4. Comparison of Eval and Versa Example Design Files

Feature Settings	**Example Design**	
	Eval (CertusPro-NX Devices)	**Versa (Avant Devices)**
Supported variants	MAC only PHY only MAC + PHY*	MAC + PHY
Simulation	Yes	Yes
Run hardware in development board	No	Yes

*Note: This MAC + PHY is supported through instantiation of MAC only + PHY only.

The following figure shows the files generated under the Eval directory.



(Z:) › my_designs › eth_10g1 › qweqwe123 › eval

Name	Date modified	Type	Size
constraint.pdc	6/5/2024 6:00 PM	PDC File	1 KB
define.v	6/5/2024 6:00 PM	V File	2 KB
dut_inst.v	14/5/2024 8:40 PM	V File	2 KB
dut_params.v	14/5/2024 8:40 PM	V File	1 KB
README.txt	6/5/2024 6:00 PM	Text Document	1 KB
tb_top_eval.v	6/5/2024 6:00 PM	V File	11 KB
tb_top_versa.sv	6/5/2024 6:00 PM	SV File	11 KB
ten_gb_subsys.do	6/5/2024 6:00 PM	DO File	1 KB
ten_gbe_pcs_test.v	6/5/2024 6:00 PM	V File	1,167 KB
ten_gbe_pll.v	6/5/2024 6:00 PM	V File	86 KB
ten_gbe_subsys.sv	6/5/2024 6:00 PM	SV File	9 KB
versa.pdc	6/5/2024 6:00 PM	PDC File	3 KB
versa_top.sv	6/5/2024 6:00 PM	SV File	25 KB

Figure 6.15. File List in the Eval Directory

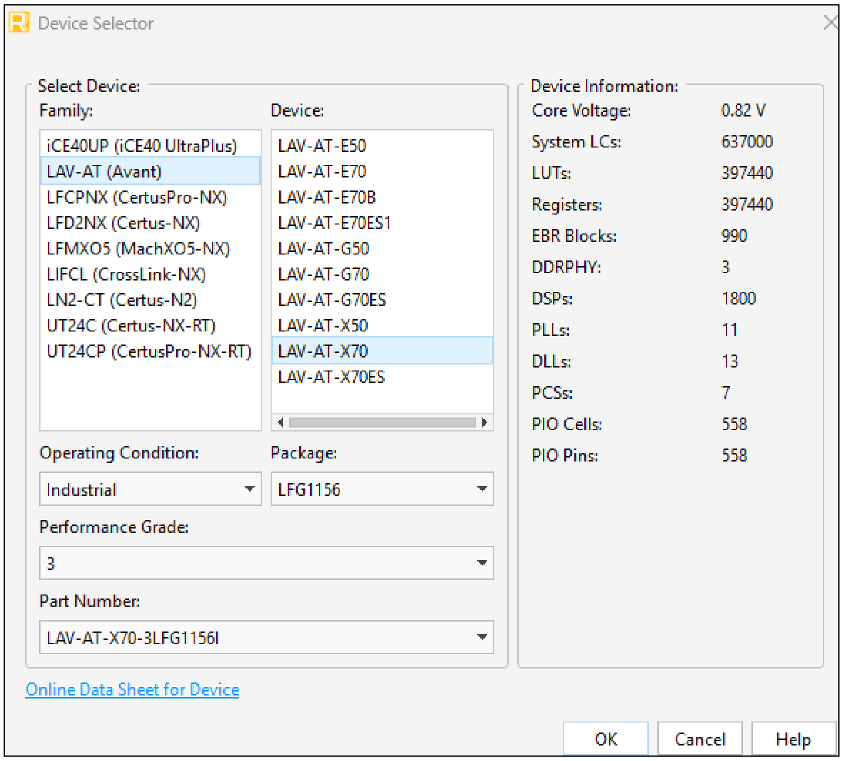
Table 6.5. Description of Generated Files in the Eval Folder

Files	Description
README.txt	Contain instructions to use Eval and Versa example design files of this directory.
constraint.pdc	Post-synthesis constraints for Eval example design.
dut_inst.v	Contain instantiation of IP based on selected configuration of the 10G Ethernet IP. Included by *eval_top.sv* and *versa_top.sv*.
dut_params.sv	Contain IP parameters based on selected configuration of the 10G Ethernet IP. Included by *tb_top_eval.sv* and *tb_top_versa.sv*.
Files supported by CertusPro-NX devices: LFCPNX-100 only	
ten_gb_subsys.do	QuestaSim script to perform simulation for the Eval testbench. This *.do* file is used to execute sub-system (Generated *<Instance Name>.v* with 10 Gb PCS IP sample design) simulation. Only applicable to the *MAC Only* option.
ten_gbe_subsys.sv	Top-level design file for the Eval example design.
ten_gbe_pcs_test.v	10GBASE-R PCS testbench for CertusPro-NX devices.
ten_gbe_pll.v	10GBASE-R PCS PLL file for CertusPro-NX devices.
tb_top_eval.sv	Testbench for the Eval example design.
Files supported by CertusPro-NX devices only (eval/versa_top/lfcpnx_10g/)	
top.v	Top-level design file for hardware example design.
tb_top.v	Top-level simulation file for hardware example design.
pll_1.v	PLL module from foundation IP.
OSCC.v	OSC module from foundation IP.
pcs_apb_w.v; *pcs_apb_w_r.v*	APB module for register configuration.
traffic_genchk.v	Traffic generator module to generate random traffic for transmission and perform checking in loopback.
debounce.v	Debounce module for hardware mechanical input.
cpnx_versa_10g.pdc	Post-synthesis constraint file for CertusPro-NX Versa board.
tb_top.do	Script to group simulation signals.
Files supported by Avant devices only: LAV-AT-G70, LAV-AT-X70, (eval/versa_top/avant_x70_10g)	
versa.pdc	Post-synthesis constraints for the Versa example design.
versa_top.sv	Top-level design file for the Versa example design.
tb_top_versa.sv	Testbench for the Versa example design.

Files	Description
debounce.v	Debounce module for hardware mechanical input.
osc_ip.v	OSC module from foundation IP.
psc_apb_w_r.v *pcs_apb_w.v*	APB module for register configuration.
traffic_genchk.v	Traffic generator module to generate random traffic for transmission and perform checking in loopback.
tb_top_versa.do	Script to group simulation signals.

To import Versa files to the Radiant software project, follow these steps:

1. For Avant devices, select part number **LAV-AT-X70-3LFG1156I** and set the **Package** option to **LFG1156**.
   Note that the current Versa example design is tested on LAV-AT-X70.



Device Selector

Select Device:

Family:
iCE40UP (iCE40 UltraPlus)
LAV-AT (Avant)
LFCPNX (CertusPro-NX)
LFD2NX (Certus-NX)
LFMXO5 (MachXO5-NX)
LIFCL (CrossLink-NX)
LN2-CT (Certus-N2)
UT24C (Certus-NX-RT)
UT24CP (CertusPro-NX-RT)

Device:
LAV-AT-E50
LAV-AT-E70
LAV-AT-E70B
LAV-AT-E70ES1
LAV-AT-G50
LAV-AT-G70
LAV-AT-G70ES
LAV-AT-X50
LAV-AT-X70
LAV-AT-X70ES

Operating Condition: Industrial

Package: LFG1156

Performance Grade: 3

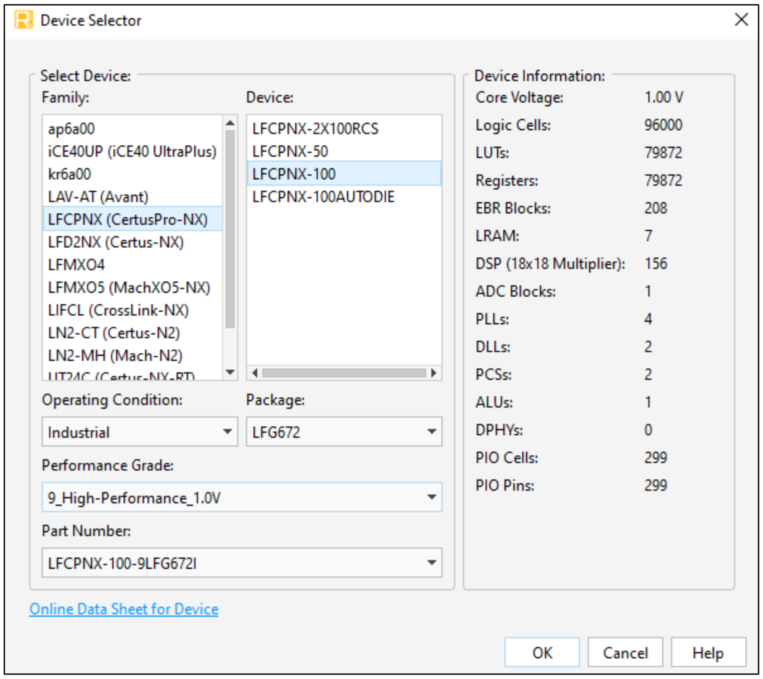
Part Number: LAV-AT-X70-3LFG1156I

Device Information:

Core Voltage:	0.82 V
System LCs:	637000
LUTs:	397440
Registers:	397440
EBR Blocks:	990
DDRPHY:	3
DSPs:	1800
PLLs:	11
DLLs:	13
PCSs:	7
PIO Cells:	558
PIO Pins:	558

Figure 6.16. Device Setup for Avant Devices

For CertusPro-NX devices, select part number **LFCPNX-100-9LFG672I** and set the **Package** option to **LFG672**.



Operating Condition: Industrial

Package: LFG672

Performance Grade: 9_High-Performance_1.0V

Part Number: LFCPNX-100-9LFG672I

Device Information:

Core Voltage:	1.00 V
Logic Cells:	96000
LUTs:	79872
Registers:	79872
EBR Blocks:	208
LRAM:	7
DSP (18x18 Multiplier):	156
ADC Blocks:	1
PLLs:	4
DLLs:	2
PCSs:	2
ALUs:	1
DPHYs:	0
PIO Cells:	299
PIO Pins:	299

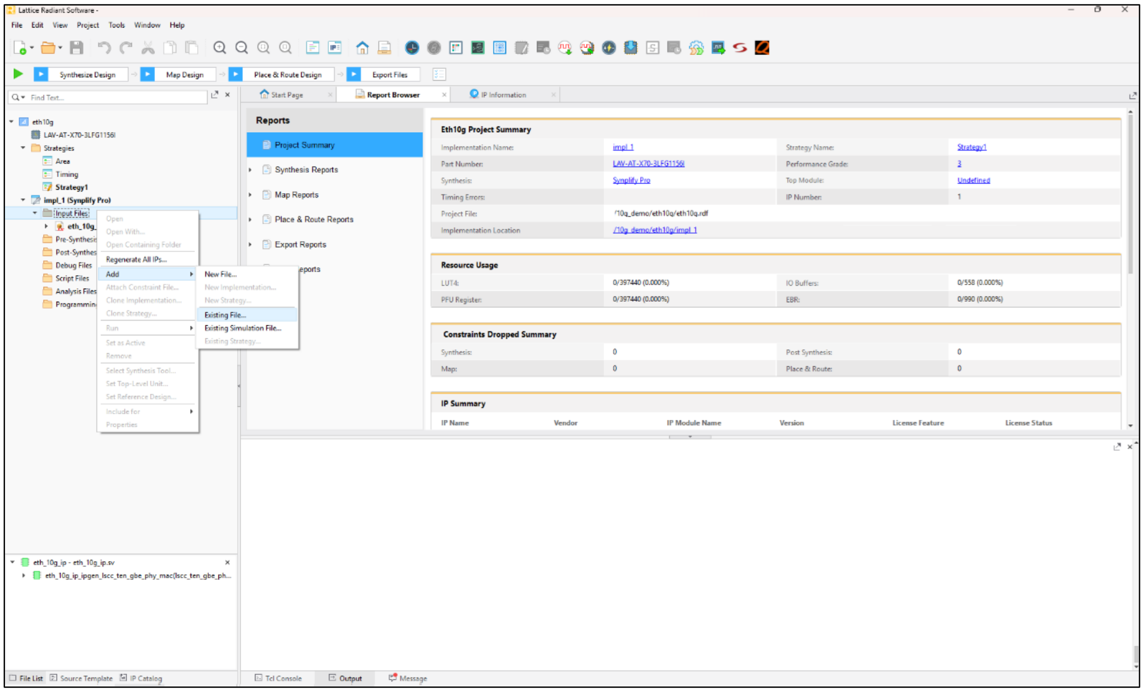
Figure 6.17. Device Setup for CertusPro-NX Devices

2. Right-click on **Input Files**, select **Add → Existing File…,** then add the *versa_top.sv* file to the Radiant software project.

   The top-level files can be found in *IP_NAME/eval/versa_top/DEVICE_INTERFACE.*

Table 6.6. Top-Level Files

File Description	Avant X70 Devices (10G Ethernet)	CertusPro-NX Devices (10G Ethernet)
Top-level file to include in software project	*versa_top.sv*	*top.v*
Simulation for top-level file to include.	*tb_top_versa.sv*	*tb_top.sv*
Post-synthesis constraint file.	*versa.pdc*	*cpnx_versa.pdc*



Eth10g Project Summary

Implementation Name:	impl_1	Strategy Name:	Strategy1
Part Number:	LAV-AT-X70-3LFG1156I	Performance Grade:	3
Synthesis:	Synplify Pro	Top Module:	Undefined
Timing Errors:		IP Number:	1
Project File:	/10g_demo/eth10g/eth10g.rdf		
Implementation Location	/10g_demo/eth10g/impl_1		

Resource Usage

LUT4:	0/397440 (0.000%)	IO Buffers:	0/558 (0.000%)
PFU Register:	0/397440 (0.000%)	EBR:	0/990 (0.000%)

Constraints Dropped Summary

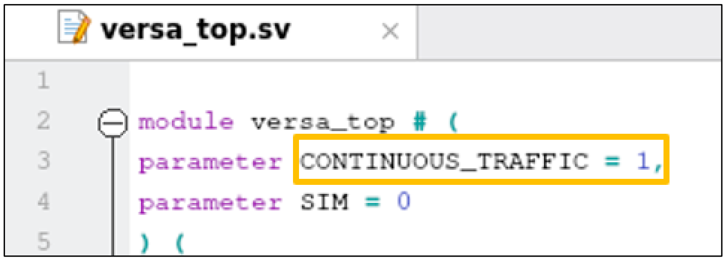
Synthesis:	0	Post Synthesis:	0
Map:	0	Place & Route:	0

IP Summary

IP Name	Vendor	IP Module Name	Version	License Feature	License Status

Figure 6.18. Add Related Files into the Radiant Software Project

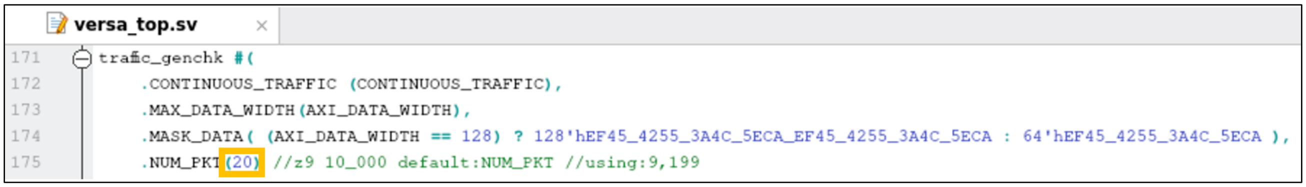
3. Versa design file: *versa_top.sv* or *top.v* file. The SIM parameter remains unchanged for both modes.
   - Continuous mode:
     a. CONTINUOUS_TRAFFIC parameter is default to 1.
     b. Allows you to pause and resume the packet transmission via push button in hardware. For more information on testing the IP with the evaluation board, refer to the Hardware Testing section.



```
module versa_top # (
parameter CONTINUOUS_TRAFFIC = 1,
parameter SIM = 0
) (
```

Figure 6.19. CONTINUOUS_TRAFFIC parameter for Continuous Mode in versa_top.sv File

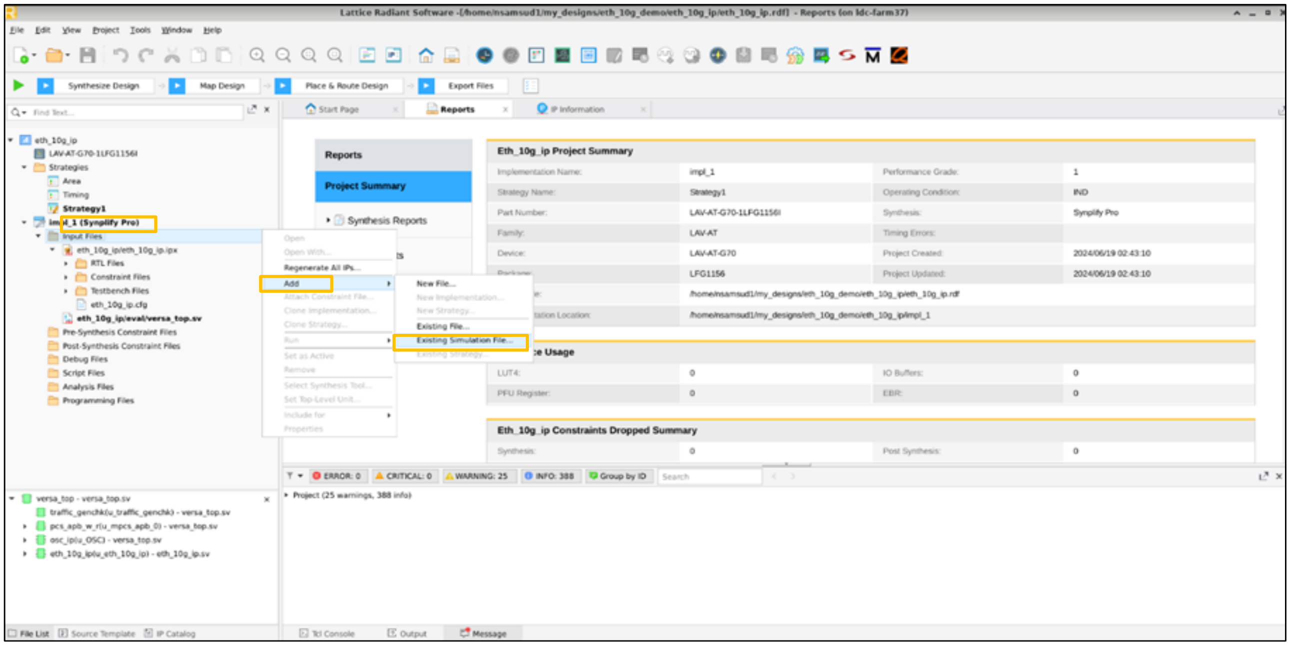
   - Non-continuous mode
     a. Allows you to change the CONTINUOUS_TRAFFIC parameter to 0.
     b. Allows you to define the number of packets transmitted through NUM_PKT parameter.



```
trafic_genchk #(
    .CONTINUOUS_TRAFFIC (CONTINUOUS_TRAFFIC),
    .MAX_DATA_WIDTH(AXI_DATA_WIDTH),
    .MASK_DATA( (AXI_DATA_WIDTH == 128) ? 128'hEF45_4255_3A4C_5ECA_EF45_4255_3A4C_5ECA : 64'hEF45_4255_3A4C_5ECA ),
    .NUM_PKT(20) //z9 10_000 default:NUM_PKT //using:9,199
```

Figure 6.20. NUM_PKT for Non-Continuous Mode

4. Right-click on **Input Files**, select **Add → Existing Simulation File**..., then add the *tb_top_versa.sv* file to the Radiant software project.



Eth_10g_ip Project Summary

Implementation Name:	impl_1	Performance Grade:	1
Strategy Name:	Strategy1	Operating Condition:	IND
Part Number:	LAV-AT-G70-1LFG1156I	Synthesis:	Synplify Pro
Family:	LAV-AT	Timing Errors:	
Device:	LAV-AT-G70	Project Created:	2024/06/19 02:43:10
Package:	LFG1156	Project Updated:	2024/06/19 02:43:10
e:	/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip.rdf		
tation Location:	/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/impl_1		

e Usage

LUT4:	0	IO Buffers:	0
PFU Register:	0	EBR:	0

Eth_10g_ip Constraints Dropped Summary

Synthesis:	0	Post Synthesis:	0

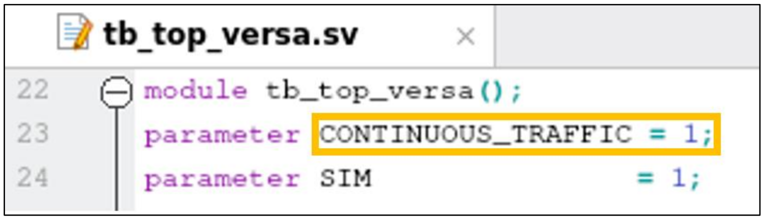
Project (25 warnings, 388 info)

versa_top - versa_top.sv
traffic_genchk(u_traffic_genchk) - versa_top.sv
pcs_apb_w_r(u_mpcs_apb_0) - versa_top.sv
osc_ip(u_OSC) - versa_top.sv
eth_10g_ip(u_eth_10g_ip) - eth_10g_ip.sv

Figure 6.21. Add the tb_top_versa.sv File into the Radiant Software Project

5. Versa testbench: *tb_versa_top.sv* or *tb_top.v*. The SIM parameter remains unchanged for both modes.

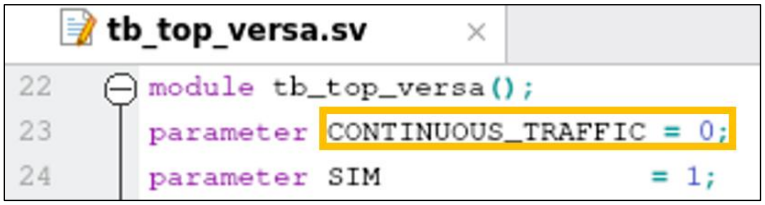
    a. CONTINUOUS_TRAFFIC parameter is default to 1 for continuous mode packet transmission.



```
module tb_top_versa();
parameter CONTINUOUS_TRAFFIC = 1;
parameter SIM                = 1;
```

Figure 6.22. CONTINUOUS_TRAFFIC Parameter for Continuous Mode in the tb_versa_top.sv File

    b. Allows you to change the CONTINUOUS_TRAFFIC parameter to 0 for non-continuous mode.



```
module tb_top_versa();
parameter CONTINUOUS_TRAFFIC = 0;
parameter SIM                = 1;
```

Figure 6.23. CONTINUOUS_TRAFFIC Parameter for Non-Continuous Mode in the tb_versa_top.sv File

6. The *tb_top_versa.sv* file must be included for simulation only so that the *versa_top.sv* file is automatically set as the top-level file for the Versa project.

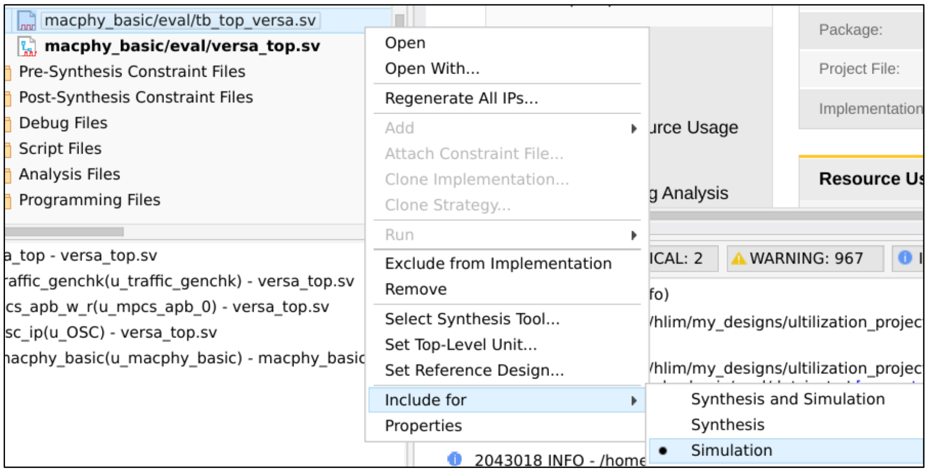


Figure 6.24. Set the tb_top_versa.sv File to be Included for Simulation Only

7. Right-click on **Post-Synthesis Constraint Files**, select **Add → Existing File...,** then add the *.pdc* file to the Radiant software project.

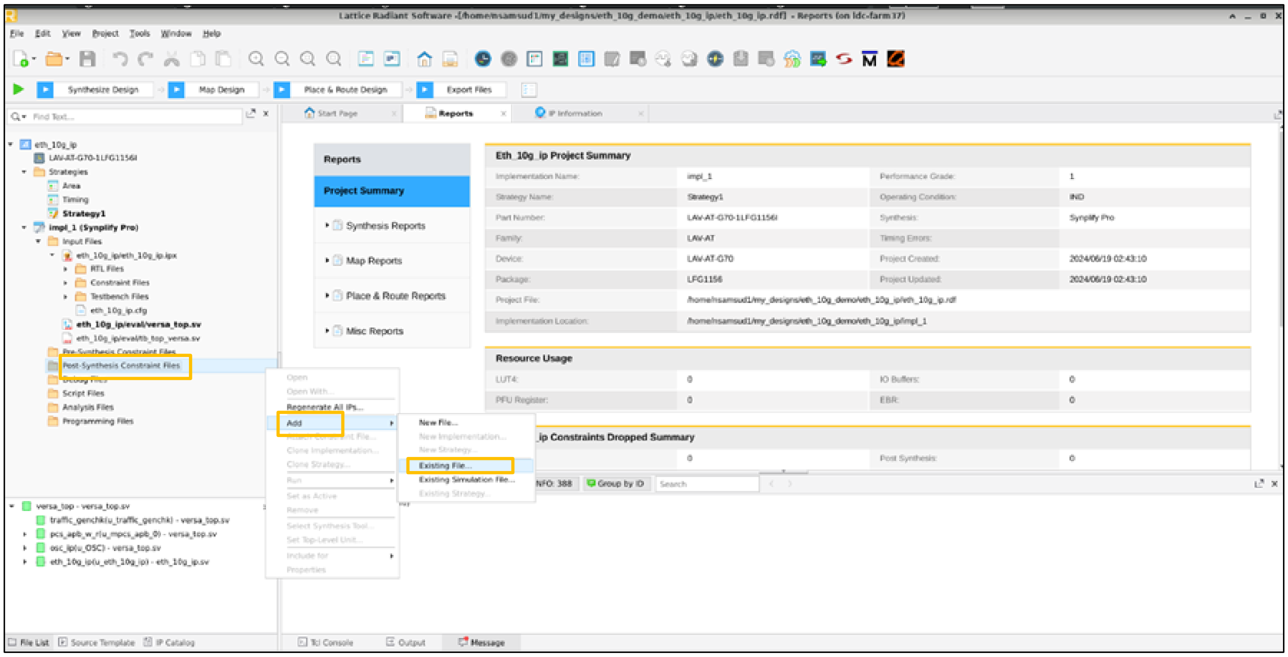
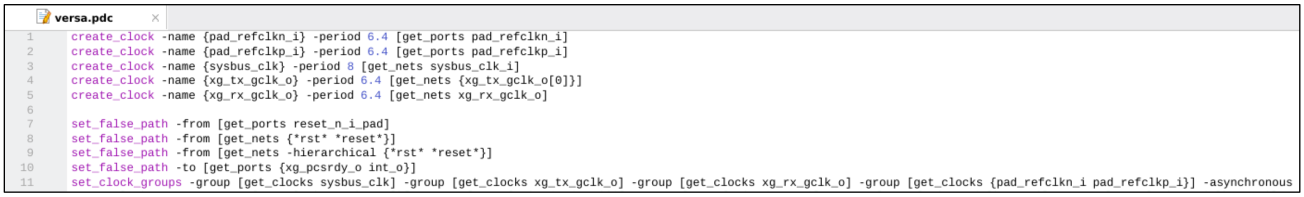


Figure 6.25. Add the .pdc File into the Radiant Software Project

8. You can add and edit the constraints using the Device Constraint Editor or by manually creating a PDC file. Post-Synthesis constraint files (*.pdc*) contain both timing and non-timing constraint *.pdc* source files for storing logical timing or physical constraints.



```
create_clock -name {pad_refclkn_i} -period 6.4 [get_ports pad_refclkn_i]
create_clock -name {pad_refclkp_i} -period 6.4 [get_ports pad_refclkp_i]
create_clock -name {sysbus_clk} -period 8 [get_nets sysbus_clk_i]
create_clock -name {xg_tx_gclk_o} -period 6.4 [get_nets {xg_tx_gclk_o[0]}]
create_clock -name {xg_rx_gclk_o} -period 6.4 [get_nets xg_rx_gclk_o]

set_false_path -from [get_ports reset_n_i_pad]
set_false_path -from [get_nets {*rst* *reset*}]
set_false_path -from [get_nets -hierarchical {*rst* *reset*}]
set_false_path -to [get_ports {xg_pcsrdy_o int_o}]
set_clock_groups -group [get_clocks sysbus_clk] -group [get_clocks xg_tx_gclk_o] -group [get_clocks xg_rx_gclk_o] -group [get_clocks {pad_refclkn_i pad_refclkp_i}] -asynchronous
```

Figure 6.26. Timing Constraint File (versa.pdc Example from an Avant Device)

9. The hardware example design preparation is completed.

6.5. Example Design Simulation

This section describes the example design testbench simulation flow for Avant devices and CertusPro-NX devices, how to create a simulation project, and how to run the simulation.

6.5.1. 10 Gb Ethernet MAC Example Design Testbench Flow

The following diagram shows the testbench simulation flow. After the simulation starts, the testbench waits for 10G Ethernet PCS link up. The APB driver will configure the 10G Ethernet MAC core to enable transmit and receive data paths. After that, packets are generated by the AXI-Stream driver and loopback at serial interface. Based on the result of the content comparison between the transmitted and loopback packets, the simulation PASSED or FAILED status is displayed.

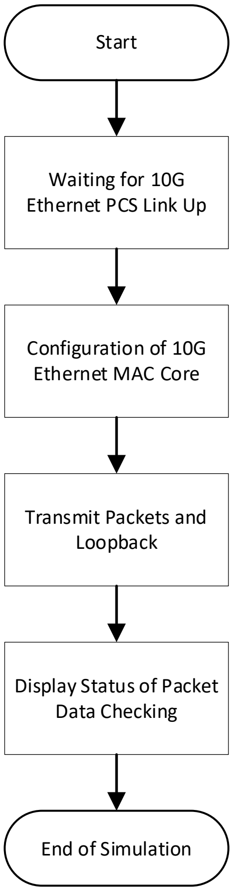
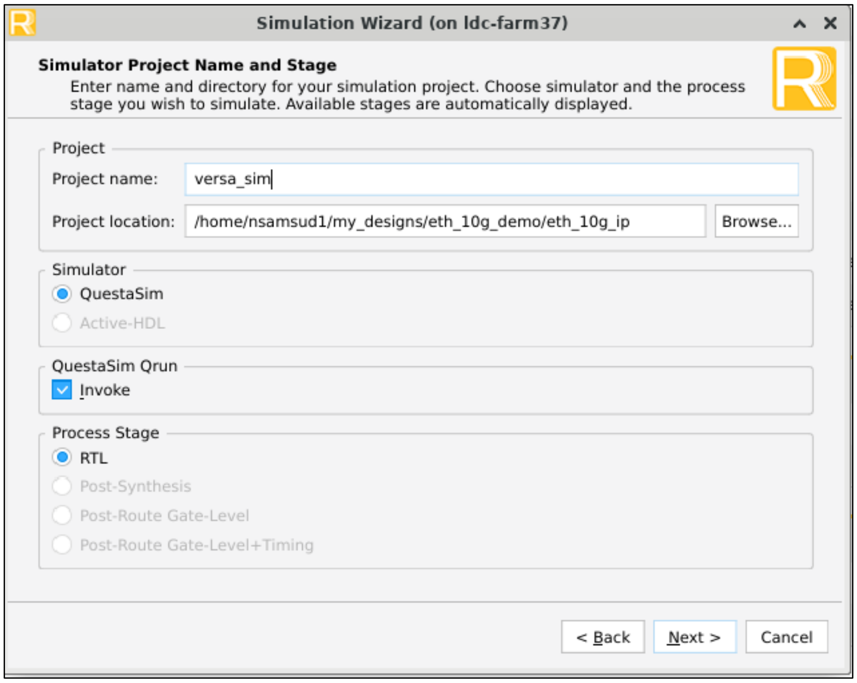

Start
Waiting for 10G Ethernet PCS Link Up
Configuration of 10G Ethernet MAC Core
Transmit Packets and Loopback
Display Status of Packet Data Checking
End of Simulation

Figure 6.27. 10G Ethernet MAC Example Design Testbench Flowchart

6.5.2. Create a Simulation Project

1. Go to menu **Tools → Simulation Wizard** or click on icon to launch the Simulation Wizard GUI.
2. Enter **Project name**, and click **Next**.



Simulation Wizard (on ldc-farm37)

Simulator Project Name and Stage

Enter name and directory for your simulation project. Choose simulator and the process stage you wish to simulate. Available stages are automatically displayed.

Project

Project name: versa_sim

Project location: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

Simulator

● QuestaSim

○ Active-HDL

QuestaSim Qrun

☑ Invoke

Process Stage

● RTL

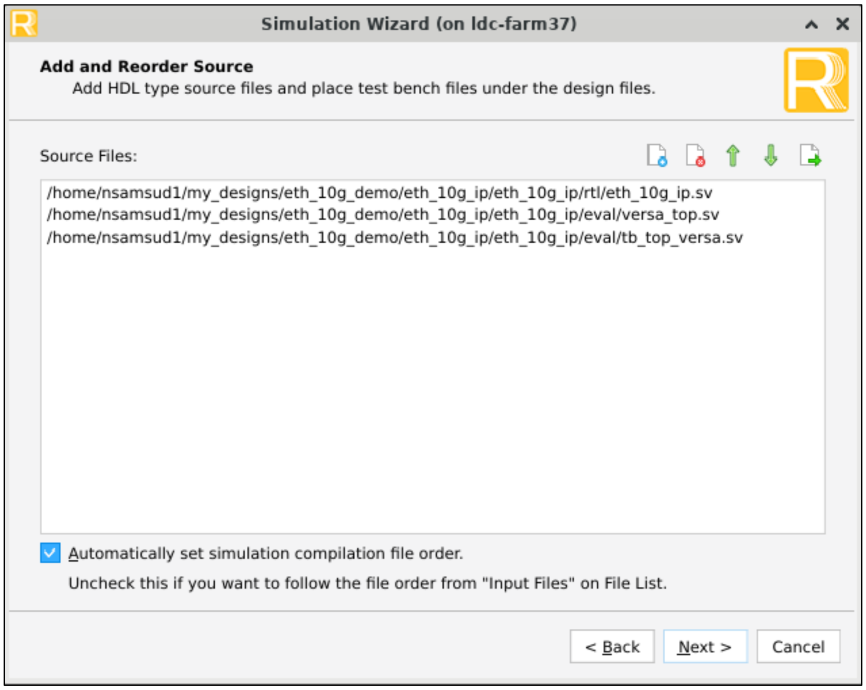
○ Post-Synthesis

○ Post-Route Gate-Level

○ Post-Route Gate-Level+Timing

Figure 6.28. Create a Simulation Project

3. Click **Next.**



Simulation Wizard (on ldc-farm37)

Add and Reorder Source

Add HDL type source files and place test bench files under the design files.

Source Files:

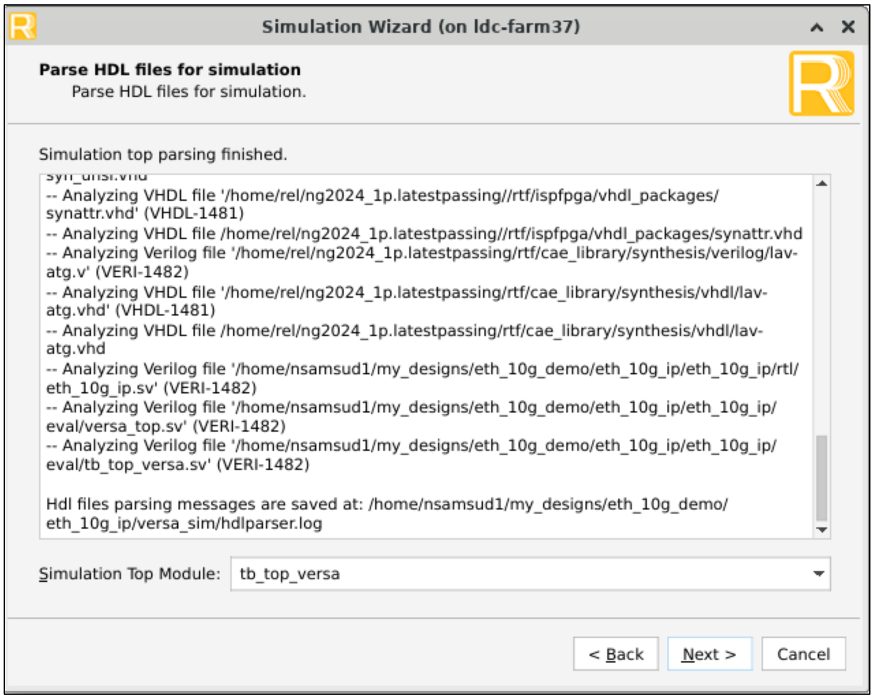
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/rtl/eth_10g_ip.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/eval/versa_top.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/eval/tb_top_versa.sv

☑ Automatically set simulation compilation file order.

Uncheck this if you want to follow the file order from "Input Files" on File List.

Figure 6.29. Include Source Files

4. Click **Next**.



Parse HDL files for simulation
Parse HDL files for simulation.

Simulation top parsing finished.

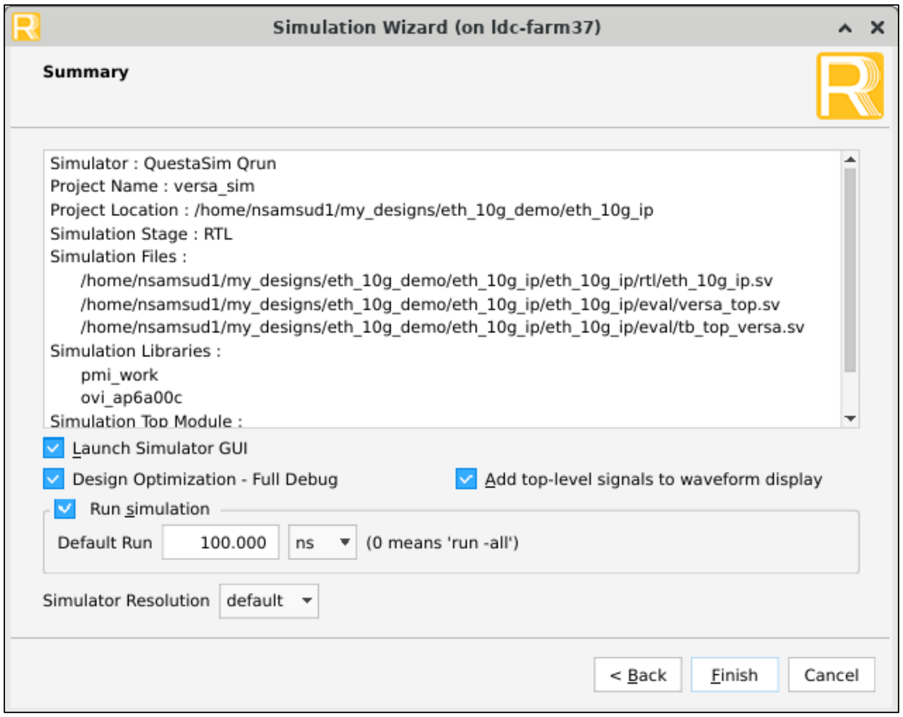
```
-- Analyzing VHDL file '/home/rel/ng2024_1p.latestpassing//rtf/ispfpga/vhdl_packages/
synattr.vhd' (VHDL-1481)
-- Analyzing VHDL file /home/rel/ng2024_1p.latestpassing//rtf/ispfpga/vhdl_packages/synattr.vhd
-- Analyzing Verilog file '/home/rel/ng2024_1p.latestpassing/rtf/cae_library/synthesis/verilog/lav-
atg.v' (VERI-1482)
-- Analyzing VHDL file '/home/rel/ng2024_1p.latestpassing/rtf/cae_library/synthesis/vhdl/lav-
atg.vhd' (VHDL-1481)
-- Analyzing VHDL file /home/rel/ng2024_1p.latestpassing/rtf/cae_library/synthesis/vhdl/lav-
atg.vhd
-- Analyzing Verilog file '/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/rtl/
eth_10g_ip.sv' (VERI-1482)
-- Analyzing Verilog file '/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/
eval/versa_top.sv' (VERI-1482)
-- Analyzing Verilog file '/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/
eval/tb_top_versa.sv' (VERI-1482)

Hdl files parsing messages are saved at: /home/nsamsud1/my_designs/eth_10g_demo/
eth_10g_ip/versa_sim/hdlparser.log
```

Simulation Top Module: tb_top_versa

Figure 6.30. Set tb_top* as Top Module

5. Change the settings according to the following figure and click **Finish** to run the QuestaSim Lattice-Edition software.



Summary

Simulator : QuestaSim Qrun
Project Name : versa_sim
Project Location : /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip
Simulation Stage : RTL
Simulation Files :
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/rtl/eth_10g_ip.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/eval/versa_top.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/eval/tb_top_versa.sv
Simulation Libraries :
pmi_work
ovi_ap6a00c
Simulation Top Module :

☑ Launch Simulator GUI

☑ Design Optimization - Full Debug ☑ Add top-level signals to waveform display

☑ Run simulation

Default Run 100.000 ns (0 means 'run -all')

Simulator Resolution default

Figure 6.31. Change the Time Setting for Default Run

6.5.2.1. Continuous Mode

1. The following waveform shows an example of a simulation result for continuous mode.

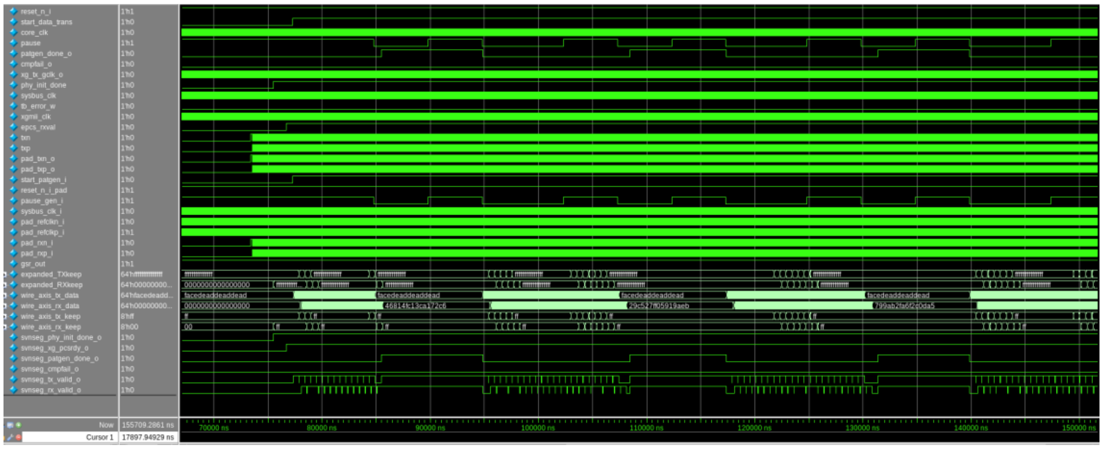
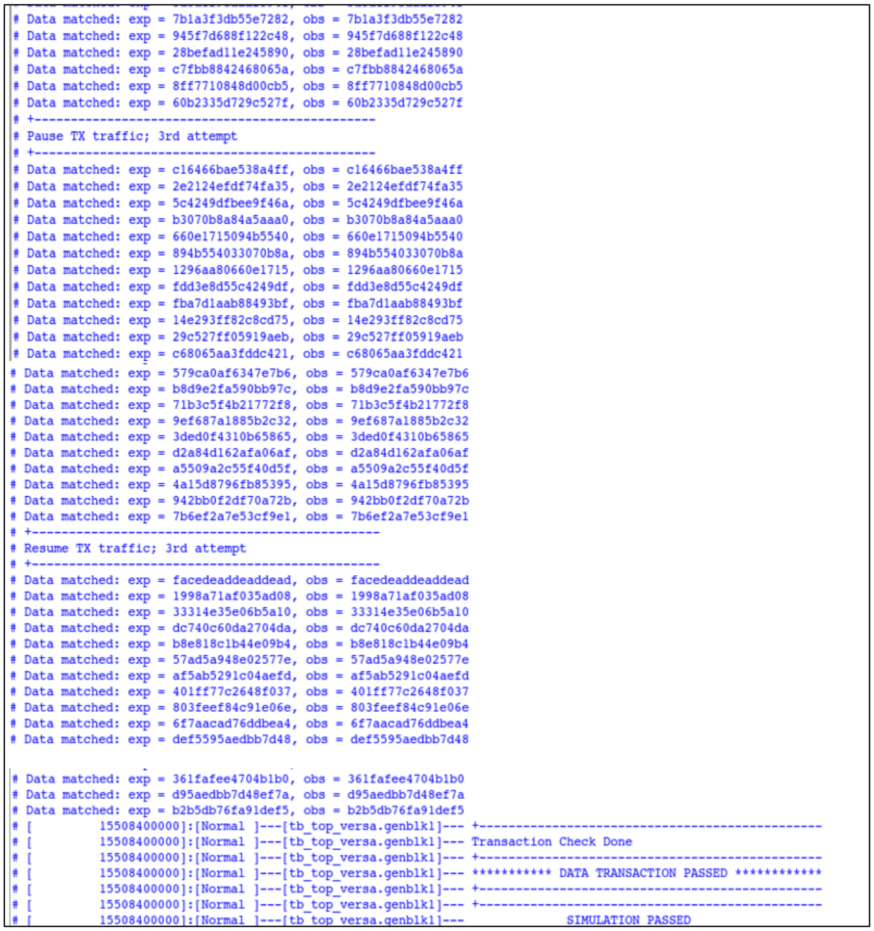


Figure 6.32. Example Design Simulation Waveform for Continuous Mode

2. The following figure shows the simulation output for continuous mode with a PASSED status.



```
# Data matched: exp = 7b1a3f3db55e7282, obs = 7b1a3f3db55e7282
# Data matched: exp = 945f7d688f122c48, obs = 945f7d688f122c48
# Data matched: exp = 28befad11e245890, obs = 28befad11e245890
# Data matched: exp = c7fbb8842468065a, obs = c7fbb8842468065a
# Data matched: exp = 8ff7710848d00cb5, obs = 8ff7710848d00cb5
# Data matched: exp = 60b2335d729c527f, obs = 60b2335d729c527f
# +-----------------------------------------------
# Pause TX traffic; 3rd attempt
# +-----------------------------------------------
# Data matched: exp = c16466bae538a4ff, obs = c16466bae538a4ff
# Data matched: exp = 2e2124efdf74fa35, obs = 2e2124efdf74fa35
# Data matched: exp = 5c4249dfbee9f46a, obs = 5c4249dfbee9f46a
# Data matched: exp = b3070b8a84a5aaa0, obs = b3070b8a84a5aaa0
# Data matched: exp = 660e1715094b5540, obs = 660e1715094b5540
# Data matched: exp = 894b554033070b8a, obs = 894b554033070b8a
# Data matched: exp = 1296aa80660e1715, obs = 1296aa80660e1715
# Data matched: exp = fdd3e8d55c4249df, obs = fdd3e8d55c4249df
# Data matched: exp = fba7d1aab88493bf, obs = fba7d1aab88493bf
# Data matched: exp = 14e293ff82c8cd75, obs = 14e293ff82c8cd75
# Data matched: exp = 29c527ff05919aeb, obs = 29c527ff05919aeb
# Data matched: exp = c68065aa3fddc421, obs = c68065aa3fddc421
# Data matched: exp = 579ca0af6347e7b6, obs = 579ca0af6347e7b6
# Data matched: exp = b8d9e2fa590bb97c, obs = b8d9e2fa590bb97c
# Data matched: exp = 71b3c5f4b21772f8, obs = 71b3c5f4b21772f8
# Data matched: exp = 9ef687a1885b2c32, obs = 9ef687a1885b2c32
# Data matched: exp = 3ded0f4310b65865, obs = 3ded0f4310b65865
# Data matched: exp = d2a84d162afa06af, obs = d2a84d162afa06af
# Data matched: exp = a5509a2c55f40d5f, obs = a5509a2c55f40d5f
# Data matched: exp = 4a15d8796fb85395, obs = 4a15d8796fb85395
# Data matched: exp = 942bb0f2df70a72b, obs = 942bb0f2df70a72b
# Data matched: exp = 7b6ef2a7e53cf9e1, obs = 7b6ef2a7e53cf9e1
# +-----------------------------------------------
# Resume TX traffic; 3rd attempt
# +-----------------------------------------------
# Data matched: exp = facedeaddeaddead, obs = facedeaddeaddead
# Data matched: exp = 1998a71af035ad08, obs = 1998a71af035ad08
# Data matched: exp = 33314e35e06b5a10, obs = 33314e35e06b5a10
# Data matched: exp = dc740c60da2704da, obs = dc740c60da2704da
# Data matched: exp = b8e818c1b44e09b4, obs = b8e818c1b44e09b4
# Data matched: exp = 57ad5a948e02577e, obs = 57ad5a948e02577e
# Data matched: exp = af5ab5291c04aefd, obs = af5ab5291c04aefd
# Data matched: exp = 401ff77c2648f037, obs = 401ff77c2648f037
# Data matched: exp = 803feef84c91e06e, obs = 803feef84c91e06e
# Data matched: exp = 6f7aacad76ddbea4, obs = 6f7aacad76ddbea4
# Data matched: exp = def5595aedbb7d48, obs = def5595aedbb7d48

# Data matched: exp = 361fafee4704b1b0, obs = 361fafee4704b1b0
# Data matched: exp = d95aedbb7d48ef7a, obs = d95aedbb7d48ef7a
# Data matched: exp = b2b5db76fa91def5, obs = b2b5db76fa91def5
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- +-----------------------------------------------
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- Transaction Check Done
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- +-----------------------------------------------
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- *********** DATA TRANSACTION PASSED ************
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- +-----------------------------------------------
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]--- +-----------------------------------------------
# [         15508400000]:[Normal ]---[tb_top_versa.genblk1]---            SIMULATION PASSED
```

Figure 6.33. Example Design Simulation Output for Continuous Mode

6.5.2.2. Non-Continuous Mode

1. The following waveform shows an example of a simulation result for non-continuous mode.

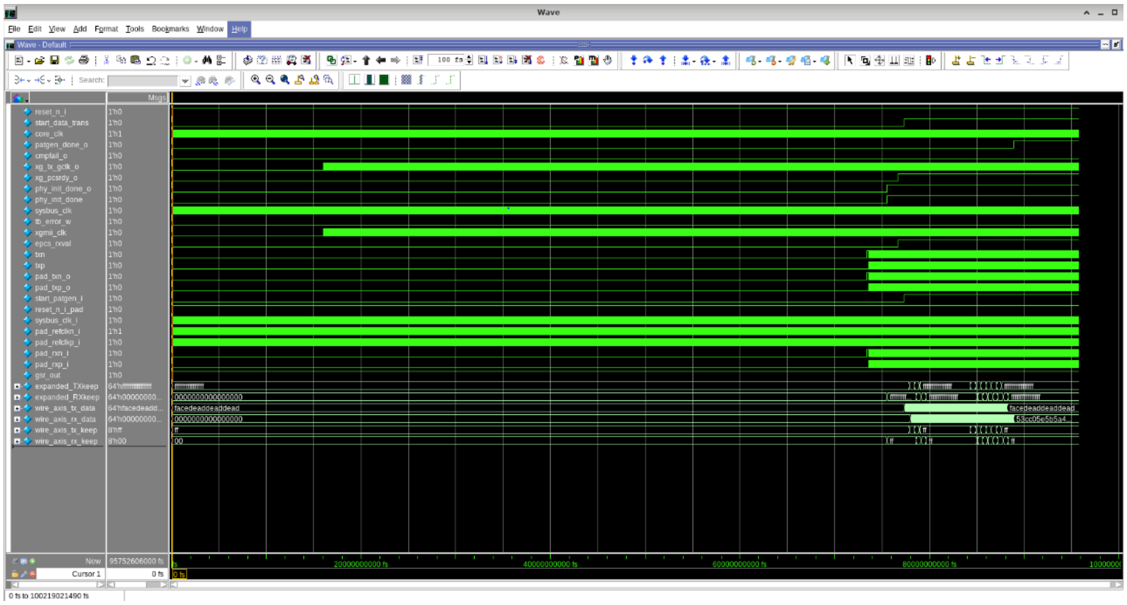
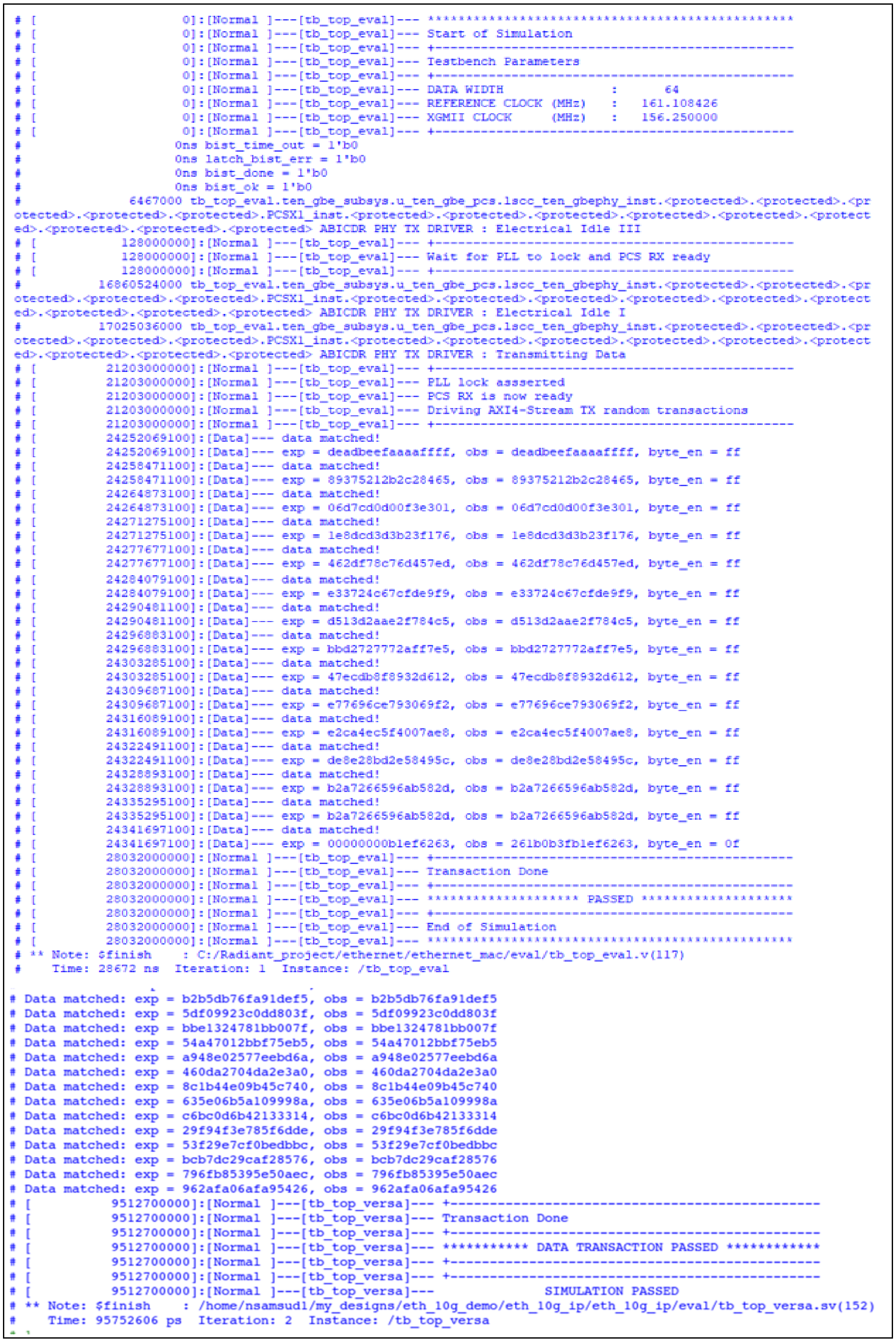


Figure 6.34. Example Design Simulation Waveform for Non-Continuous Mode

2. The following figure shows the simulation output for non-continuous mode with a PASSED status.



```
# [                   0]:[Normal ]---[tb_top_eval]--- ************************************************
# [                   0]:[Normal ]---[tb_top_eval]--- Start of Simulation
# [                   0]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [                   0]:[Normal ]---[tb_top_eval]--- Testbench Parameters
# [                   0]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [                   0]:[Normal ]---[tb_top_eval]--- DATA WIDTH               :       64
# [                   0]:[Normal ]---[tb_top_eval]--- REFERENCE CLOCK (MHz)    :    161.108426
# [                   0]:[Normal ]---[tb_top_eval]--- XGMII CLOCK     (MHz)    :    156.250000
# [                   0]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
#                     0ns bist_time_out = 1'b0
#                     0ns latch_bist_err = 1'b0
#                     0ns bist_done = 1'b0
#                     0ns bist_ok = 1'b0
#              6467000 tb_top_eval.ten_gbe_subsys.u_ten_gbe_pcs.lscc_ten_gbephy_inst.<protected>.<protected>.<pr
otected>.<protected>.<protected>.PCSX1_inst.<protected>.<protected>.<protected>.<protected>.<protected>.<protect
ed>.<protected>.<protected>.<protected> ABICDR PHY TX DRIVER : Electrical Idle III
# [           128000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [           128000000]:[Normal ]---[tb_top_eval]--- Wait for PLL to lock and PCS RX ready
# [           128000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
#          16860524000 tb_top_eval.ten_gbe_subsys.u_ten_gbe_pcs.lscc_ten_gbephy_inst.<protected>.<protected>.<pr
otected>.<protected>.<protected>.PCSX1_inst.<protected>.<protected>.<protected>.<protected>.<protected>.<protect
ed>.<protected>.<protected>.<protected> ABICDR PHY TX DRIVER : Electrical Idle I
#          17025036000 tb_top_eval.ten_gbe_subsys.u_ten_gbe_pcs.lscc_ten_gbephy_inst.<protected>.<protected>.<pr
otected>.<protected>.<protected>.PCSX1_inst.<protected>.<protected>.<protected>.<protected>.<protected>.<protect
ed>.<protected>.<protected>.<protected> ABICDR PHY TX DRIVER : Transmitting Data
# [         21203000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [         21203000000]:[Normal ]---[tb_top_eval]--- PLL lock asserted
# [         21203000000]:[Normal ]---[tb_top_eval]--- PCS RX is now ready
# [         21203000000]:[Normal ]---[tb_top_eval]--- Driving AXI4-Stream TX random transactions
# [         21203000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [         24252069100]:[Data]--- data matched!
# [         24252069100]:[Data]--- exp = deadbeefaaaaffff, obs = deadbeefaaaaffff, byte_en = ff
# [         24258471100]:[Data]--- data matched!
# [         24258471100]:[Data]--- exp = 89375212b2c28465, obs = 89375212b2c28465, byte_en = ff
# [         24264873100]:[Data]--- data matched!
# [         24264873100]:[Data]--- exp = 06d7cd0d00f3e301, obs = 06d7cd0d00f3e301, byte_en = ff
# [         24271275100]:[Data]--- data matched!
# [         24271275100]:[Data]--- exp = 1e8dcd3d3b23f176, obs = 1e8dcd3d3b23f176, byte_en = ff
# [         24277677100]:[Data]--- data matched!
# [         24277677100]:[Data]--- exp = 462df78c76d457ed, obs = 462df78c76d457ed, byte_en = ff
# [         24284079100]:[Data]--- data matched!
# [         24284079100]:[Data]--- exp = e33724c67cfde9f9, obs = e33724c67cfde9f9, byte_en = ff
# [         24290481100]:[Data]--- data matched!
# [         24290481100]:[Data]--- exp = d513d2aae2f784c5, obs = d513d2aae2f784c5, byte_en = ff
# [         24296883100]:[Data]--- data matched!
# [         24296883100]:[Data]--- exp = bbd2727772aff7e5, obs = bbd2727772aff7e5, byte_en = ff
# [         24303285100]:[Data]--- data matched!
# [         24303285100]:[Data]--- exp = 47ecdb8f8932d612, obs = 47ecdb8f8932d612, byte_en = ff
# [         24309687100]:[Data]--- data matched!
# [         24309687100]:[Data]--- exp = e77696ce793069f2, obs = e77696ce793069f2, byte_en = ff
# [         24316089100]:[Data]--- data matched!
# [         24316089100]:[Data]--- exp = e2ca4ec5f4007ae8, obs = e2ca4ec5f4007ae8, byte_en = ff
# [         24322491100]:[Data]--- data matched!
# [         24322491100]:[Data]--- exp = de8e28bd2e58495c, obs = de8e28bd2e58495c, byte_en = ff
# [         24328893100]:[Data]--- data matched!
# [         24328893100]:[Data]--- exp = b2a7266596ab582d, obs = b2a7266596ab582d, byte_en = ff
# [         24335295100]:[Data]--- data matched!
# [         24335295100]:[Data]--- exp = b2a7266596ab582d, obs = b2a7266596ab582d, byte_en = ff
# [         24341697100]:[Data]--- data matched!
# [         24341697100]:[Data]--- exp = 00000000b1ef6263, obs = 261b0b3fb1ef6263, byte_en = 0f
# [         28032000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [         28032000000]:[Normal ]---[tb_top_eval]--- Transaction Done
# [         28032000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [         28032000000]:[Normal ]---[tb_top_eval]--- ******************** PASSED ********************
# [         28032000000]:[Normal ]---[tb_top_eval]--- +-----------------------------------------------
# [         28032000000]:[Normal ]---[tb_top_eval]--- End of Simulation
# [         28032000000]:[Normal ]---[tb_top_eval]--- ************************************************
# ** Note: $finish    : C:/Radiant_project/ethernet/ethernet_mac/eval/tb_top_eval.v(117)
#    Time: 28672 ns  Iteration: 1  Instance: /tb_top_eval

# Data matched: exp = b2b5db76fa91def5, obs = b2b5db76fa91def5
# Data matched: exp = 5df09923c0dd803f, obs = 5df09923c0dd803f
# Data matched: exp = bbe1324781bb007f, obs = bbe1324781bb007f
# Data matched: exp = 54a47012bbf75eb5, obs = 54a47012bbf75eb5
# Data matched: exp = a948e02577eebd6a, obs = a948e02577eebd6a
# Data matched: exp = 460da2704da2e3a0, obs = 460da2704da2e3a0
# Data matched: exp = 8c1b44e09b45c740, obs = 8c1b44e09b45c740
# Data matched: exp = 635e06b5a109998a, obs = 635e06b5a109998a
# Data matched: exp = c6bc0d6b42133314, obs = c6bc0d6b42133314
# Data matched: exp = 29f94f3e785f6dde, obs = 29f94f3e785f6dde
# Data matched: exp = 53f29e7cf0bedbbc, obs = 53f29e7cf0bedbbc
# Data matched: exp = bcb7dc29caf28576, obs = bcb7dc29caf28576
# Data matched: exp = 796fb85395e50aec, obs = 796fb85395e50aec
# Data matched: exp = 962afa06afa95426, obs = 962afa06afa95426
# [          9512700000]:[Normal ]---[tb_top_versa]--- +-----------------------------------------------
# [          9512700000]:[Normal ]---[tb_top_versa]--- Transaction Done
# [          9512700000]:[Normal ]---[tb_top_versa]--- +-----------------------------------------------
# [          9512700000]:[Normal ]---[tb_top_versa]--- *********** DATA TRANSACTION PASSED ************
# [          9512700000]:[Normal ]---[tb_top_versa]--- +-----------------------------------------------
# [          9512700000]:[Normal ]---[tb_top_versa]--- +-----------------------------------------------
# [          9512700000]:[Normal ]---[tb_top_versa]---            SIMULATION PASSED
# ** Note: $finish    : /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/eval/tb_top_versa.sv(152)
#    Time: 95752606 ps  Iteration: 2  Instance: /tb_top_versa
```

Figure 6.35. Example Design Simulation Output for Non-Continuous Mode

6.6. Hardware Testing

The 10G Ethernet IP is hardware tested on Avant G/X Versa boards and CertusPro-NX Versa board. The following setup demonstrates the steps for hardware testing on an Avant G/X Versa board:

1. Set up the board as shown in the following figure.
2. Ensure the input power is 12 V.
3. It is optional that JP17 jumper is set.to enable the board to operate in 10 Gb. This is applicable for Avant-G/X Versa board design only.

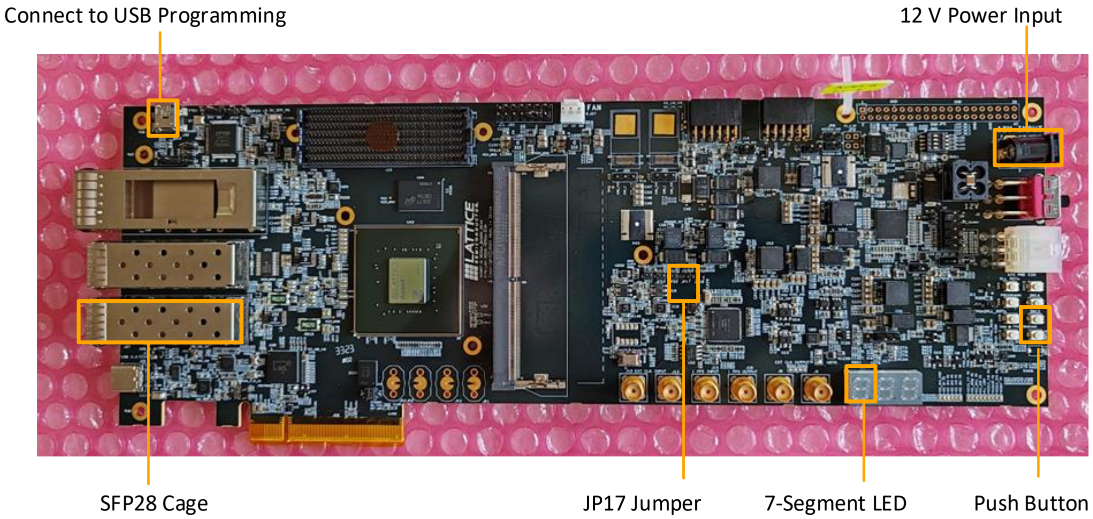


Figure 6.36. Avant-X Versa Board Setup

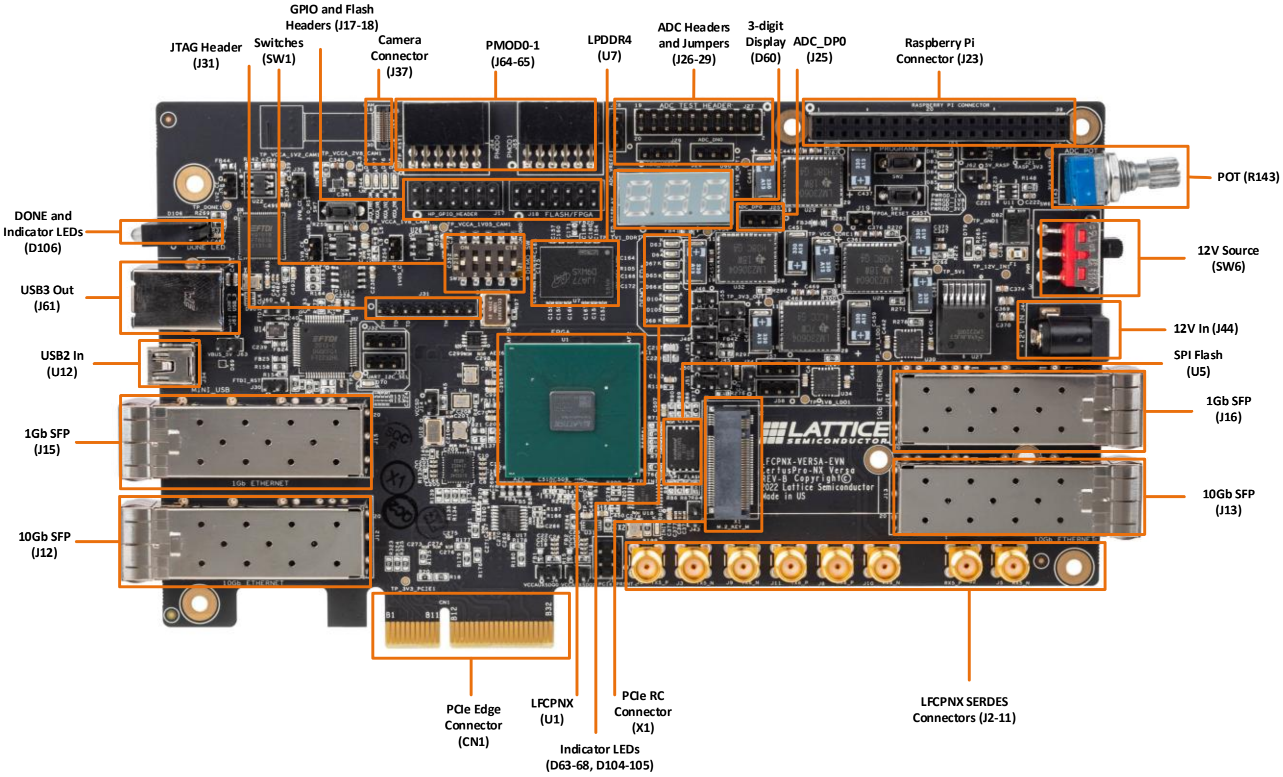


Figure 6.37. CertusPro-NX Versa Board Setup

4. The following figure shows the 10 Gb SFP28 optical module and optical loopback module as well as the connection to the Avant-X Versa board SFP1. For CertusPro-NX Versa board, use J12 10Gb SFP.

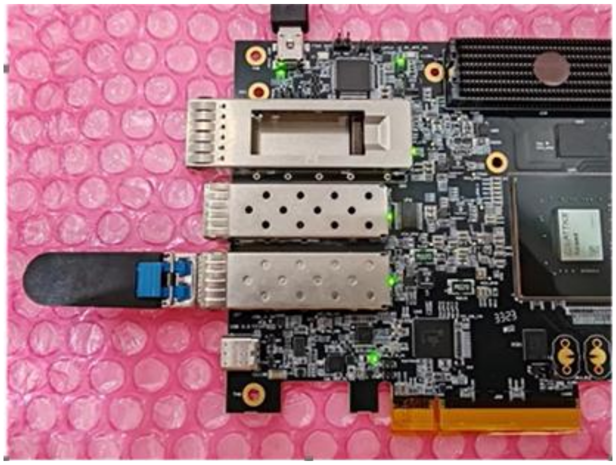


Figure 6.38. Loopback Module Setup

5. Click on the Run button shown in the following figure to compile the design until it generates the bitstream file.

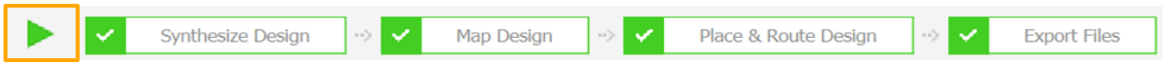
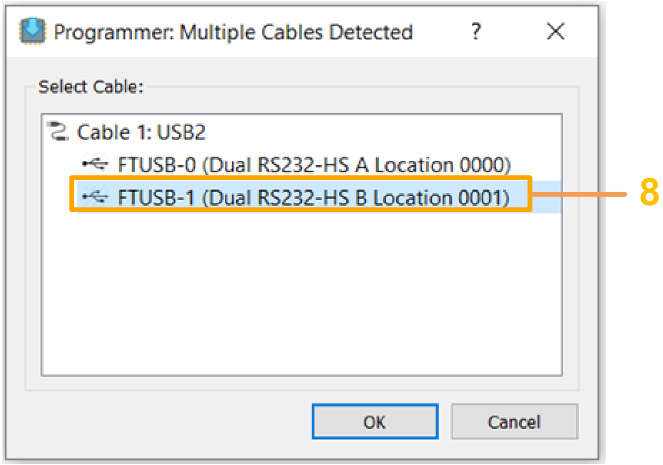


Figure 6.39. Successful Design Compilation

6. Launch the Radiant Programmer via menu **Tools→ Programmer** or click on the Radiant Programmer icon .
7. Click on the **Detect Cable** button.
8. Select the correct cable associated with the Avant-X Versa board. The number of cables shown in your environment setup may be different. Select FTUSB-1 for Avant-X Versa board and FTUSB-0 for CertusPro NX Versa board.
9. Select the correct bitstream file that is generated from the 10G Ethernet project in step 5.
10. Click on the programming icon . The software displays the programming status.

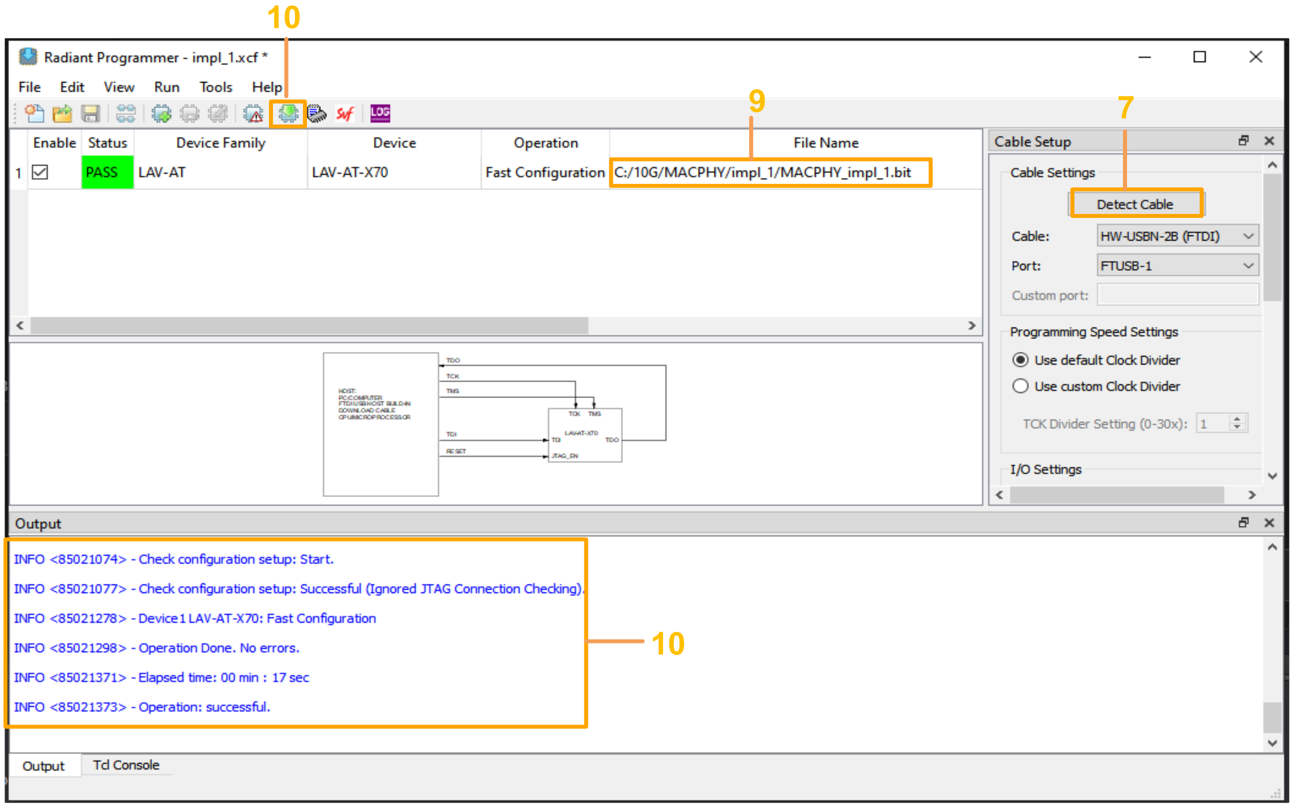


Programmer: Multiple Cables Detected

Select Cable:

Cable 1: USB2
- FTUSB-0 (Dual RS232-HS A Location 0000)
- FTUSB-1 (Dual RS232-HS B Location 0001)

Figure 6.40. Cable Selection for Avant-AT-X Device



	Enable	Status	Device Family	Device	Operation	File Name
1	☑	PASS	LAV-AT	LAV-AT-X70	Fast Configuration	C:/10G/MACPHY/impl_1/MACPHY_impl_1.bit

Cable: HW-USBN-2B (FTDI)
Port: FTUSB-1
Custom port:

Programming Speed Settings
● Use default Clock Divider
○ Use custom Clock Divider
TCK Divider Setting (0-30x): 1

I/O Settings

```
INFO <85021074> - Check configuration setup: Start.
INFO <85021077> - Check configuration setup: Successful (Ignored JTAG Connection Checking).
INFO <85021278> - Device1 LAV-AT-X70: Fast Configuration
INFO <85021298> - Operation Done. No errors.
INFO <85021371> - Elapsed time: 00 min : 17 sec
INFO <85021373> - Operation: successful.
```

Figure 6.41. Programmer GUI

11. After the programming stage, press the pushbutton SW12 for Avant-X/G Versa board, or toggle DIP switch DIP_SW1 to HIGH for CertusPro-NX Versa board to trigger a forced reset even though the example design will automatically trigger a reset. Set the DIP switch for pattern generator to LOW before triggering a reset.
12. After the design has linked up, toggle DIP_SW_1 for Avant-X/G Versa board or DIP_SW2 to HIGH for CertusPro-NX Versa board to start Ethernet frames generation and transmission. The Ethernet frames loopback via the connected SFP with the optical loopback module and the traffic generator receives and compares the frames.
13. You can pause and resume the Ethernet frames transmission by pressing the push button SW13* for Avant-X/G Versa board or DIP_SW2 to LOW for CertusPro-NX Versa board. The 7-segment LED glows according to the following diagram, which indicates that the hardware has been successfully set up and received a passing status.

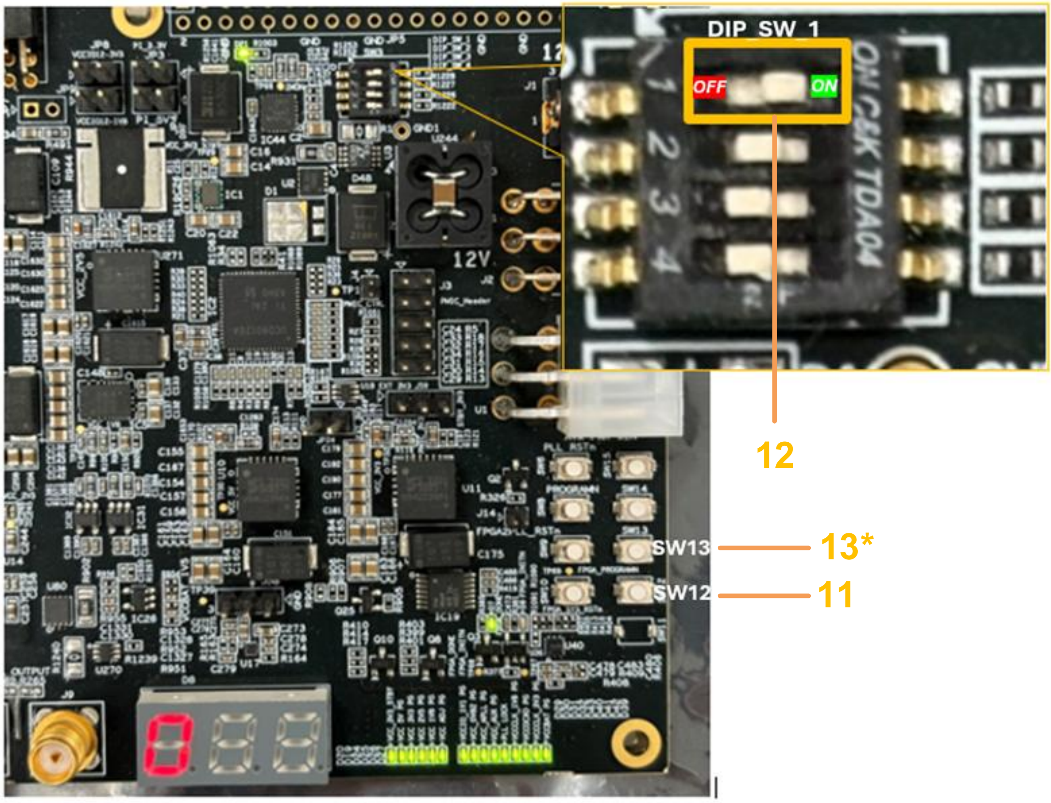


Figure 6.42. Passing Scenario on Avant-AT-X Device

* **Note:** Applicable if continuous mode is enabled (CONTINUOUS_TRAFFIC=1).

14. In the failing scenario due to various reasons (for example, link is not stable, loopback module is not connected, and so on), the 7-segment LED glows according to the following diagram. Check the connectivity, re-flash the bitstream, or power cycle to resolve it.

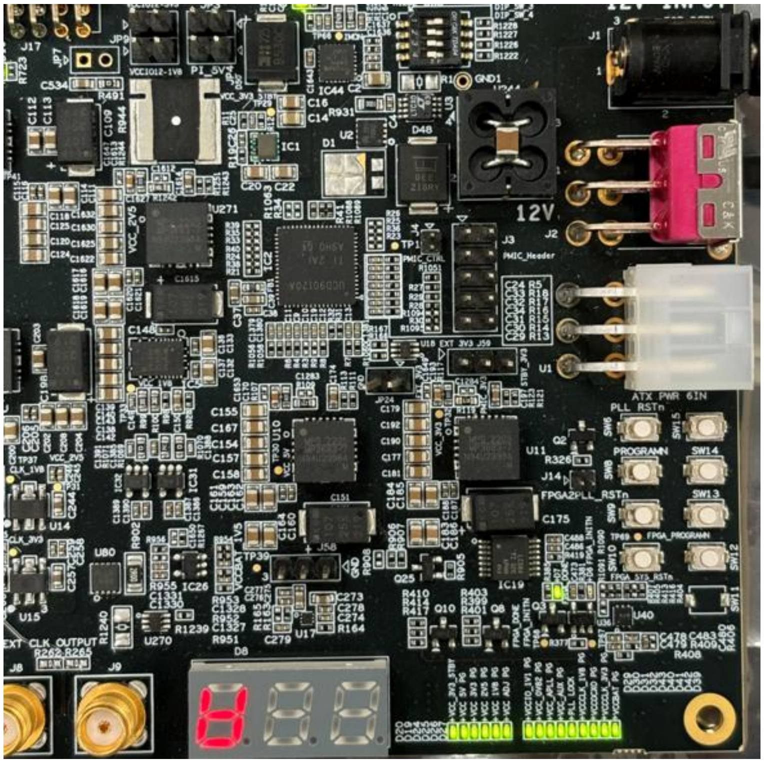


Figure 6.43. Failing Scenario on Avant-AT-X Device

15. The following diagram shows the various LED segments for Avant-X/G Versa board and the description for each segment is described in the following table.

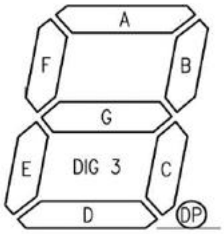


Figure 6.44. LED 7-Segment

Table 6.7. LED 7-Segment Description for Avant-X/G Versa Board

Segment	Description
A	Traffic generator ended, or frames transmission paused*.
B	Reset (push button) deasserted.
C	10G Ethernet IP initialization is done.
D	10G Ethernet IP PCS is ready.
E	Frames transmitting or traffic generator ended.
F	Frames receiving or traffic generator ended.
G	Loopback patterns comparison fails.

*** Note**: Default continuous mode and if the SW13 is pressed, Segment A of the 7-segment LED indicates that the frames transmission is paused. For non-continuous mode, Segment A of the 7-segment LED indicates that the traffic generator has ended.

Table 6.8. General-Purpose LED (Green Color LED) Description for CertusPro-NX Versa Board

Signal Name	Description
LED_0	PLL locked.
LED_1	PHY transmitter ready.
LED_2	PHY receiver ready.
LED_3	TX transmission is live.
LED_4	RX transmission is live.
LED_5	Transmission completed or paused.
LED_6	Loopback patterns comparison fails.
LED_7	PHY bit error rate is high.

7. Designing with the IP

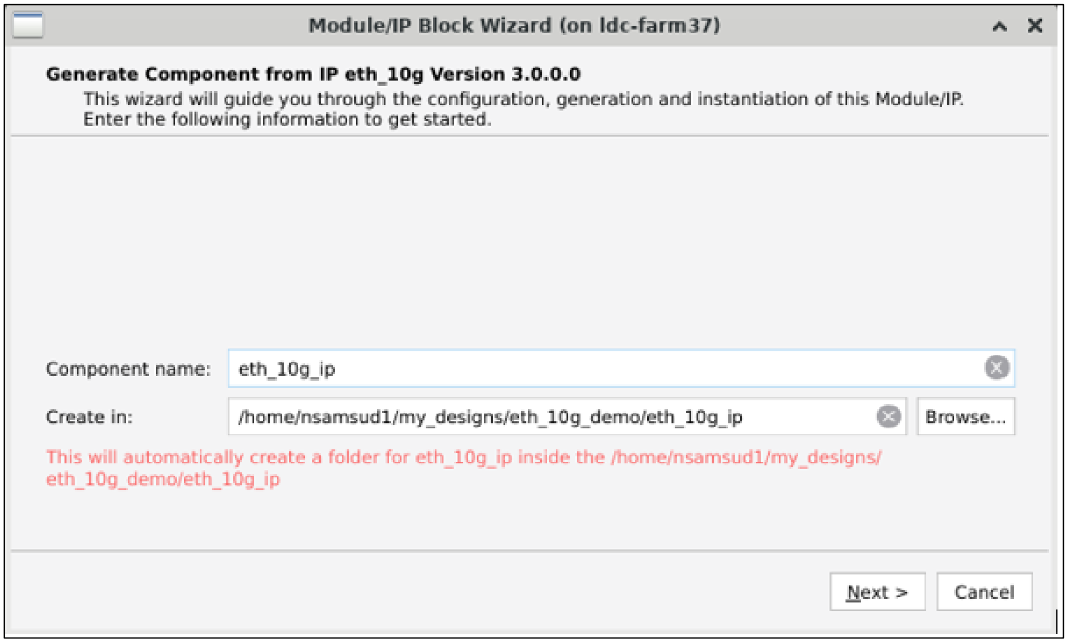
This section provides information on how to generate the IP core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, refer to the Lattice Radiant Software User Guide and relevant Lattice tutorials.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

7.1. Generating and Instantiating the IP

You can use the Lattice Radiant software to generate IP modules and integrate them into the device's architecture. The following steps describe how to generate the 10G Ethernet IP core in the Lattice Radiant software:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **10 Gb Ethernet** under **IP, Connectivity** category. The **Module/IP Block Wizard** opens. Enter values in the **Component name** and the **Create in** fields and click **Next**.



Module/IP Block Wizard (on ldc-farm37)

Generate Component from IP eth_10g Version 3.0.0.0

This wizard will guide you through the configuration, generation and instantiation of this Module/IP.
Enter the following information to get started.

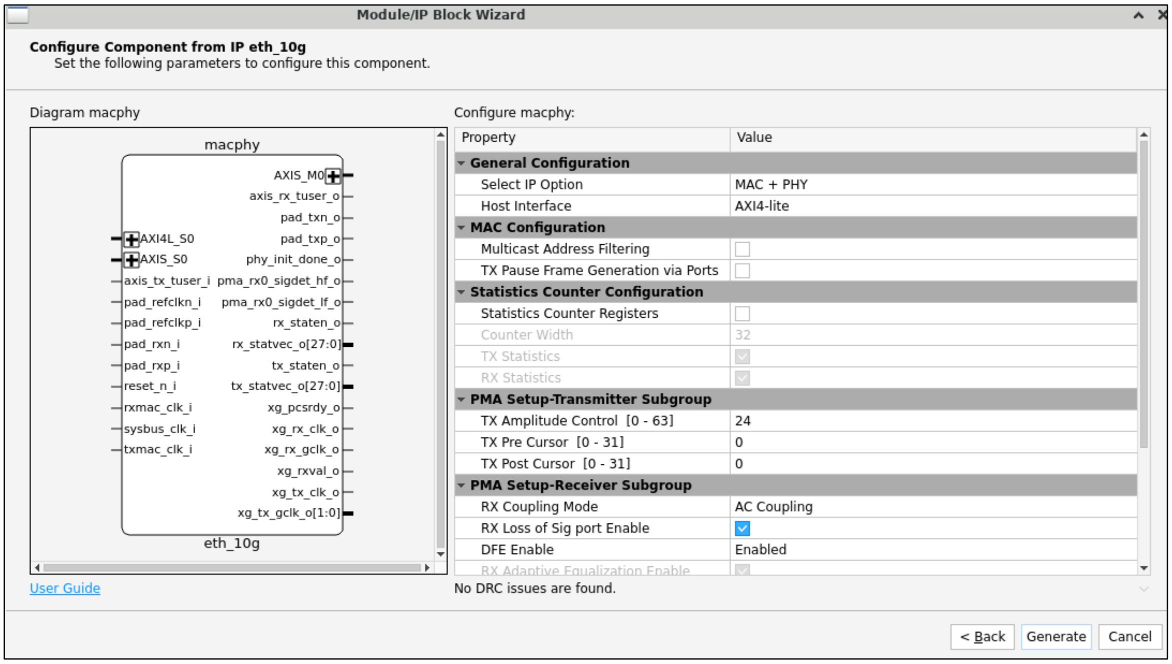
Component name: eth_10g_ip

Create in: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

This will automatically create a folder for eth_10g_ip inside the /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

Figure 7.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected 10G Ethernet IP core using the drop-down lists and checkboxes. The following figure shows a configuration of the 10G Ethernet IP core. For details on the configuration options, refer to the relevant *IP Parameter Description* section.



Module/IP Block Wizard

Configure Component from IP eth_10g
Set the following parameters to configure this component.

Diagram macphy

Configure macphy:

Property	Value
General Configuration	
Select IP Option	MAC + PHY
Host Interface	AXI4-lite
MAC Configuration	
Multicast Address Filtering	☐
TX Pause Frame Generation via Ports	☐
Statistics Counter Configuration	
Statistics Counter Registers	☐
Counter Width	32
TX Statistics	☑
RX Statistics	☑
PMA Setup-Transmitter Subgroup	
TX Amplitude Control [0 - 63]	24
TX Pre Cursor [0 - 31]	0
TX Post Cursor [0 - 31]	0
PMA Setup-Receiver Subgroup	
RX Coupling Mode	AC Coupling
RX Loss of Sig port Enable	☑
DFE Enable	Enabled
RX Adaptive Equalization Enable	☑

User Guide

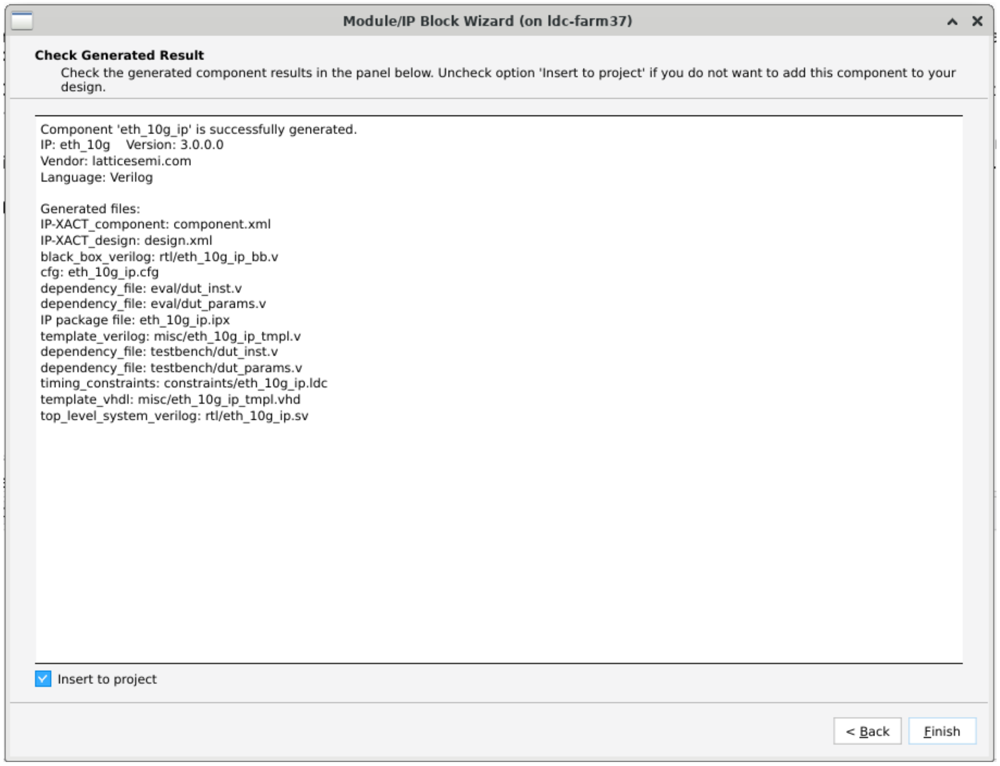
No DRC issues are found.

Figure 7.2. Configure the User Interface of 10G Ethernet IP Core

Note:

* Applicable for LAV-AT-G70 and LAV-AT-X70.

4. Click **Generate**. The **Check Generated Result** dialog box opens with design block messages and results as shown in the following figure.



Check Generated Result

Check the generated component results in the panel below. Uncheck option 'Insert to project' if you do not want to add this component to your design.

```
Component 'eth_10g_ip' is successfully generated.
IP: eth_10g    Version: 3.0.0.0
Vendor: latticesemi.com
Language: Verilog

Generated files:
IP-XACT_component: component.xml
IP-XACT_design: design.xml
black_box_verilog: rtl/eth_10g_ip_bb.v
cfg: eth_10g_ip.cfg
dependency_file: eval/dut_inst.v
dependency_file: eval/dut_params.v
IP package file: eth_10g_ip.ipx
template_verilog: misc/eth_10g_ip_tmpl.v
dependency_file: testbench/dut_inst.v
dependency_file: testbench/dut_params.v
timing_constraints: constraints/eth_10g_ip.ldc
template_vhdl: misc/eth_10g_ip_tmpl.vhd
top_level_system_verilog: rtl/eth_10g_ip.sv
```

☑ Insert to project

Figure 7.3. Check Generated Result

5. Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields.
6. You can synthesize your generated design by clicking on **Synthesize Design** located in the top-left corner of the screen.

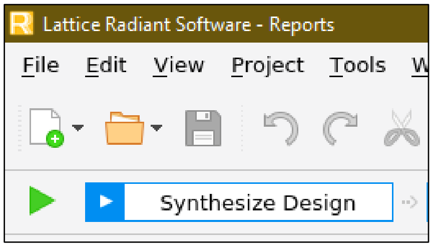


Figure 7.4. Synthesize Design

7.1.1. Generated Files and File Structure

The generated 10G Ethernet IP core package includes the black box (*<Instance Name>_bb.v*) and instance templates (*<Instance Name>_tmpl.v/vhd*) that can be used to instantiate the core in a top-level design. An RTL example of the top-level reference source file (*<Instance Name>.v*) that can be used as an instantiation template for the IP core is also provided. You may also use this example as the starting template for your top-level design.

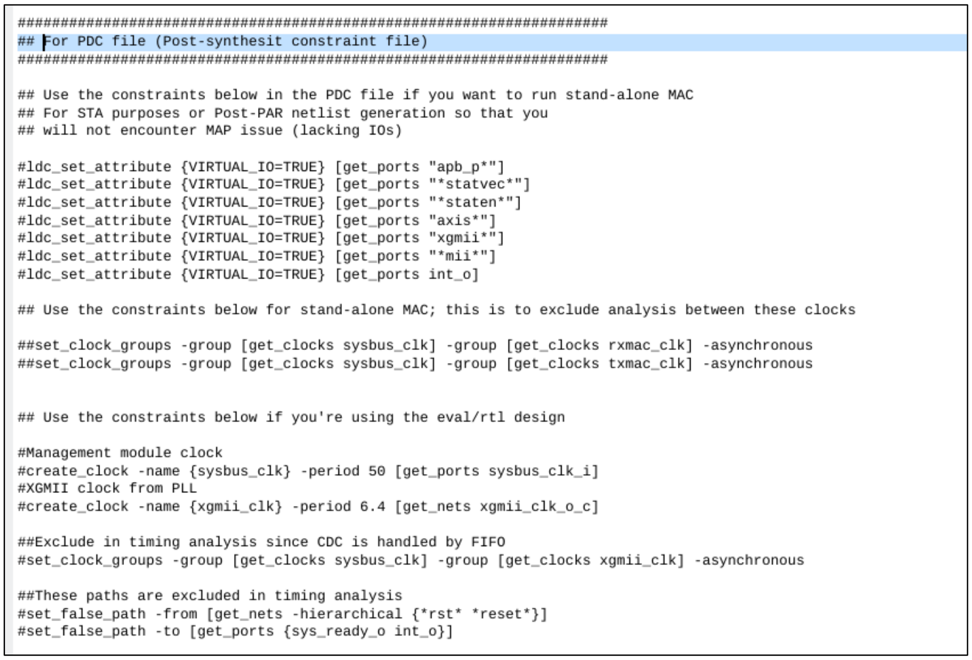
Table 7.1. Generated File List

Generated Files	Description
<Instance Name>.ipx	This file contains the information on the files associated with the generated IP.
<Instance Name>.cfg	This file contains the parameter values used in IP configuration.
component.xml	This file contains the ipxact:component information of the IP.
design.xml	This file documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Instance Name>.v	This file provides an RTL example of the top-level source file that instantiates the IP core.
rtl/<Instance Name>_bb.v	This file provides the synthesis closed-box.
misc/<Instance Name>_tmpl.v *misc /<Instance Name>_tmpl.vhd*	These files provide instance templates for the IP core.
eval/ ten_gb_subsys.do	This .do file is used to execute sub-system (*Generated <Instance Name>.v* with 10 Gb PCS IP sample design) simulation. Only applicable to the *MAC Only* option. Note: For other generated files in the Eval folder, refer to Table 6.5. Description of Generated Files in the Eval Folder.
testbench/tb_top.v	Top testbench to run loopback test of the generated *<Instance Name>.v* file.

7.2. Design Implementation

Completing your design includes additional steps to specify analog properties, pin assignments, and timing and physical constraints. You can add and edit the constraints using the Device Constraint Editor or by manually creating a PDC file.

Post-Synthesis constraint files (*.pdc*) contain both timing and non-timing constraint *.pdc* source files for storing logical timing or physical constraints. Refer to *.ldc* and include them in the (*.pdc*) files based on your usage.



```
#####################################################################
## For PDC file (Post-synthesit constraint file)
#####################################################################

## Use the constraints below in the PDC file if you want to run stand-alone MAC
## For STA purposes or Post-PAR netlist generation so that you
## will not encounter MAP issue (lacking IOs)

#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "apb_p*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "*statvec*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "*staten*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "axis*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "xgmii*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports "*mii*"]
#ldc_set_attribute {VIRTUAL_IO=TRUE} [get_ports int_o]

## Use the constraints below for stand-alone MAC; this is to exclude analysis between these clocks

##set_clock_groups -group [get_clocks sysbus_clk] -group [get_clocks rxmac_clk] -asynchronous
##set_clock_groups -group [get_clocks sysbus_clk] -group [get_clocks txmac_clk] -asynchronous


## Use the constraints below if you're using the eval/rtl design

#Management module clock
#create_clock -name {sysbus_clk} -period 50 [get_ports sysbus_clk_i]
#XGMII clock from PLL
#create_clock -name {xgmii_clk} -period 6.4 [get_nets xgmii_clk_o_c]

##Exclude in timing analysis since CDC is handled by FIFO
#set_clock_groups -group [get_clocks sysbus_clk] -group [get_clocks xgmii_clk] -asynchronous

##These paths are excluded in timing analysis
#set_false_path -from [get_nets -hierarchical {*rst* *reset*}]
#set_false_path -to [get_ports {sys_ready_o int_o}]
```

Figure 7.5. Timing Constraint File (.pdc) for the 10G MAC IP

Constraints that are added using the Device Constraint Editor are saved to the active *.pdc* file. The active post-synthesis design constraint file is then used as input for post-synthesis processes.

For more information on how to create or edit constraints and how to use the Device Constraint Editor, refer to the relevant sections in the Lattice Radiant Software User Guide.

7.3. Timing Constraints

Timing constraints (*.sdc*) is provided to ensure that the IP core meets the design timing requirement in Lattice FPGA device. The constraint file contains the necessary design constraints to ensure the proper timing closure. The timing constraints file will be generated automatically during IP generation. However, clock constraints must be performed at user level design.

You must include the following clock constraints in your design per IP configuration. Note that you might need to modify the port name according to your RTL.

For more information on timing constraints, refer to the Lattice Radiant Timing Constraints Methodology Application Note (FPGA-AN-02059).

7.3.1. Create Clock Constraints for CertusPro-NX and Avant G/X MAC Only Option

```
#create 156.25MHz clock for CertusPro-NX and Avant G/X txmac and rxmac clocks
create_clock -name {rxmac_clk_i} -period 6.4 [get_ports rxmac_clk_i]
create_clock -name {txmac_clk_i} -period 6.4 [get_ports txmac_clk_i]

#create 100MHz clock for CertusPro-NX and Avant G/X sysbus_clk_i, supported maximum
frequency up to 100MHz
create_clock -name {sysbus_clk_i} -period 10 [get_ports sysbus_clk_i]
```

7.3.2. Create Clock Constraints for CertusPro-NX and Avant G/X PHY Only Option

```
#create 161.132812MHz clock for CertusPro-NX SerDes refclk pin
create_clock -name {pad_refclkn_i} -period 6.2 [get_ports pad_refclkn_i]
create_clock -name {pad_refclkp_i} -period 6.2 [get_ports pad_refclkp_i]

#create 156.25MHz clock for Avant G/X SerDes refclk pin
create_clock -name {pad_refclkn_i} -period 6.4 [get_ports pad_refclkn_i]
create_clock -name {pad_refclkp_i} -period 6.4 [get_ports pad_refclkp_i]

#create 156.25MHz for Avant G/X PHY
create_clock -name {xg_rx_clk_i} -period 6.4 [get_ports xg_rx_clk_i]
create_clock -name {xg_tx_clk_i} -period 6.4 [get_ports xg_tx_clk_i]
create_clock -name {xg_rx_clk_o} -period 6.4 [get_ports xg_rx_clk_o]
create_clock -name {xg_tx_clk_o} -period 6.4 [get_ports xg_tx_clk_o]
create_clock –name {xg_tx_gclk_o0} -period 6.4 [get_ports xg_tx_gclk_clk_o[0]]
create_clock –name {xg_tx_gclk_o1} -period 6.4 [get_ports xg_tx_gclk_clk_o[1]]
create_clock -name {xg_rx_gclk_o} -period 6.4 [get_ports xg_rx_gclk_clk_o]

#create 156.25MHz and 322.265625MHz clock for CertusPro-NX PHY
create_clock -name {xg_rx_clk_i} -period 6.4 [get_ports xg_rx_clk_i]
create_clock -name {xg_tx_clk_i} -period 6.4 [get_ports xg_tx_clk_i]
create_clock -name {xg_rx_clk_o} -period 3.1 [get_ports xg_rx_clk_o]
create_clock -name {xg_tx_clk_o} -period 3.1 [get_ports xg_tx_clk_o]

#create 100MHz clock for CertusPro-NX PMA logic. This clock is not needed for Avant G/X.
create_clock –name {xg_pcs_clkin_i} -period 10 [get_ports xg_pcs_clkin_i]
```

```
#create 100Mhz clock for CertusPro-NX and Avant G/X sysbus_clk_i, supported maximum
frequency up to 100MHz
create_clock -name {sysbus_clk} -period 10 [get_ports sysbus_clk_i]
```

7.3.3. Create Clock Constraints for CertusPro-NX and Avant G/X MAC + PHY Option

```
#create 161.132812MHz clock for CertusPro-NX SerDes refclk pin
create_clock -name {pad_refclkn_i} -period 6.2 [get_ports pad_refclkn_i]
create_clock -name {pad_refclkp_i} -period 6.2 [get_ports pad_refclkp_i]

#create 156.25MHz clock for Avant G/X SerDes refclk pin
create_clock -name {pad_refclkn_i} -period 6.4 [get_ports pad_refclkn_i]
create_clock -name {pad_refclkp_i} -period 6.4 [get_ports pad_refclkp_i]

# create 156.25MHz and 322.265625MHz clock for CertusPro-NX
create_clock -name {mac_clk} -period 6.4 [get_ports mac_clk]
create_clock -name {xg_rx_clk_i} -period 6.4 [get_ports xg_rx_clk_i]
create_clock -name {xg_tx_clk_i} -period 6.4 [get_ports xg_tx_clk_i]
create_clock -name {xg_rx_clk_o} -period 3.1 [get_ports xg_rx_clk_o]
create_clock -name {xg_tx_clk_o} -period 3.1 [get_ports xg_tx_clk_o]

#create 100MHz clock for CertusPro-NX logic. This clock is not needed for Avant G/X.
create_clock –name {xg_pcs_clkin_i} -period 10 [get_ports xg_pcs_clkin_i]

#create 156.25MHz clock for Avant G/X.
create_clock –name {txmac_clk_i} -period 6.4 [get_ports txmac_clk_i]
create_clock -name {rxmac_clk_i} -period 6.4 [get_ports rxmac_clk_i]
create_clock -name {xg_rx_clk_i} -period 6.4 [get_ports xg_rx_clk_i]
create_clock -name {xg_tx_clk_i} -period 6.4 [get_ports xg_tx_clk_i]
create_clock –name {xg_tx_clk_o} -period 6.4 [get_ports xg_tx_clk_o]
create_clock -name {xg_rx_clk_o} -period 6.4 [get_ports xg_rx_clk_o]
create_clock –name {xg_tx_gclk_o0} -period 6.4 [get_ports xg_tx_gclk_o[0]]
create_clock –name {xg_tx_gclk_o1} -period 6.4 [get_ports xg_tx_gclk_o[1]]
create_clock -name {xg_rx_gclk_o} -period 6.4 [get_ports xg_rx_gclk_o]

#create 100Mhz clock for CertusPro-NX and Avant G/X  sysbus_clk_i, supported maximum
frequency up to 100MHz
create_clock -name {sysbus_clk_i} -period 10 [get_ports sysbus_clk_i]
```

7.4. Specifying the Strategy

The Radiant software provides two predefined strategies—Area and Timing. It also enables you to create customized strategies. For details on how to create a new strategy, refer to the *Strategies* section of the Lattice Radiant Software User Guide.

7.5. Running Functional Simulation

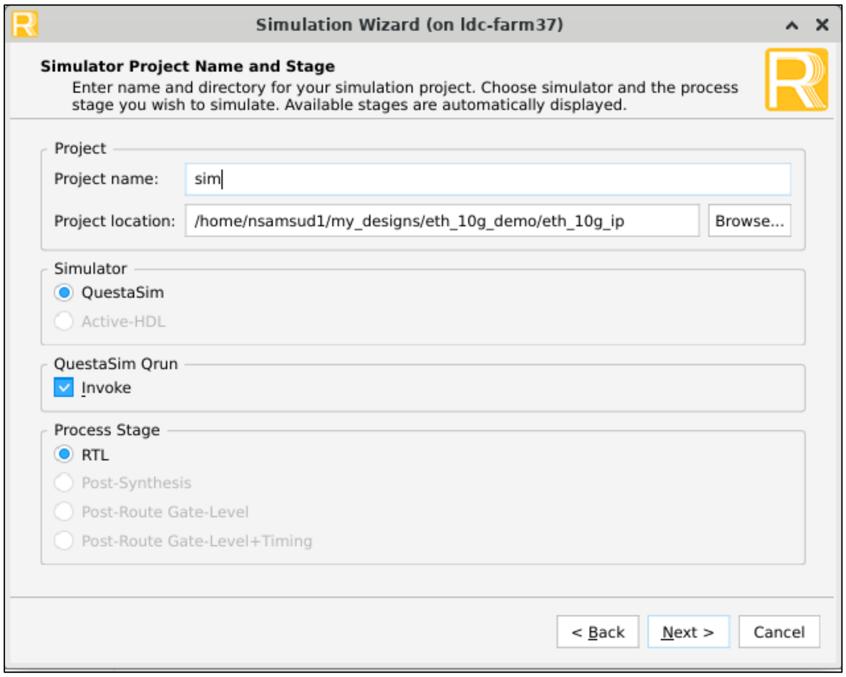
You can run functional simulation after the IP is generated.

To run the standalone QuestaSim software, the FOUNDRY environment variable must be set as follow:

set ::env(FOUNDRY) <Radiant installation path>/ispfpga

To run functional simulation, follow these steps:

1. Click the button located on the **Toolbar** to initiate the **Simulation Wizard**, as shown in Figure 7.6.



Simulator Project Name and Stage

Enter name and directory for your simulation project. Choose simulator and the process stage you wish to simulate. Available stages are automatically displayed.

Project

Project name: sim

Project location: /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip

Simulator

● QuestaSim

○ Active-HDL

QuestaSim Qrun

☑ Invoke

Process Stage

● RTL

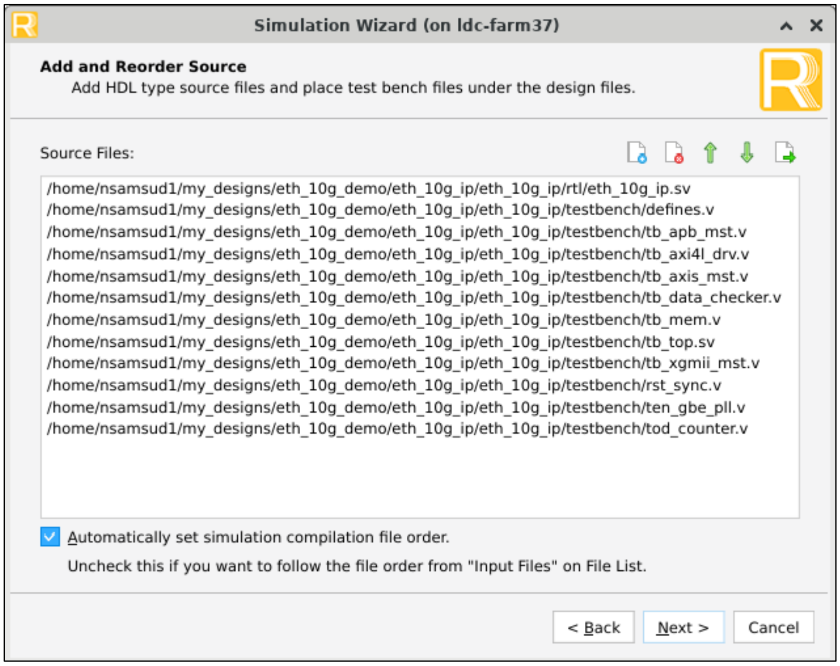
○ Post-Synthesis

○ Post-Route Gate-Level

○ Post-Route Gate-Level+Timing

Figure 7.6. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window, as shown in the following figure.



Add and Reorder Source

Add HDL type source files and place test bench files under the design files.

Source Files:

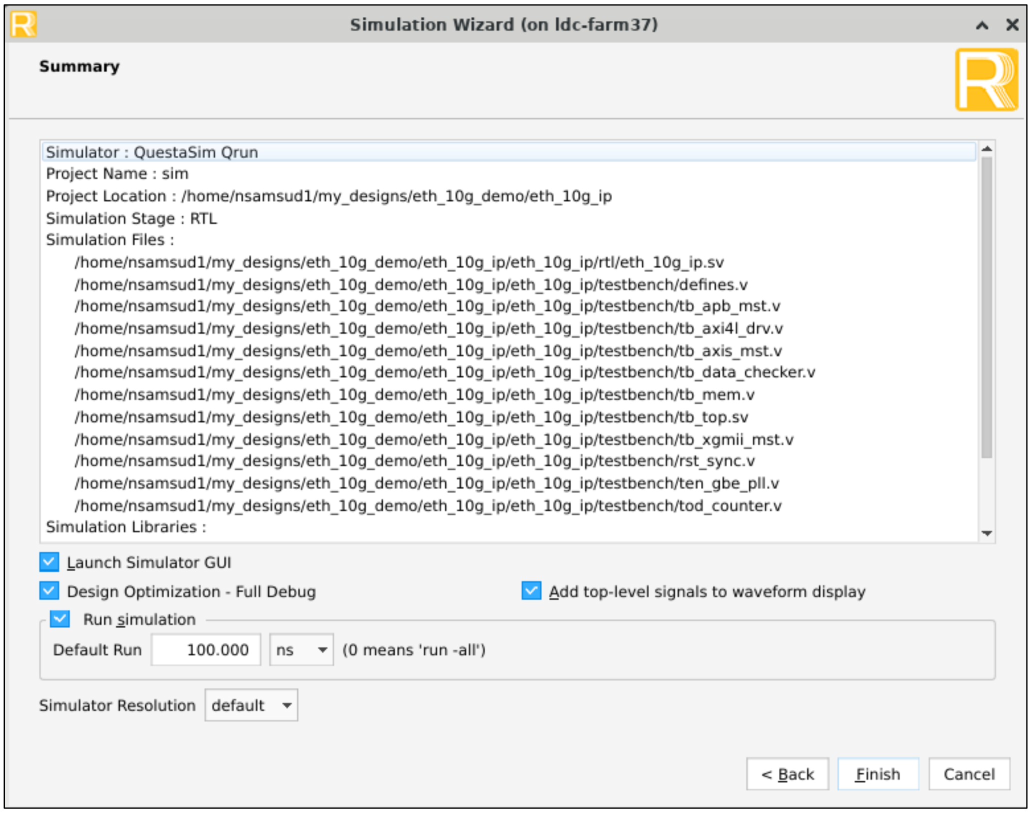
```
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/rtl/eth_10g_ip.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/defines.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_apb_mst.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_axi4l_drv.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_axis_mst.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_data_checker.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_mem.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_top.sv
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_xgmii_mst.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/rst_sync.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/ten_gbe_pll.v
/home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tod_counter.v
```

☑ Automatically set simulation compilation file order.

Uncheck this if you want to follow the file order from "Input Files" on File List.

Figure 7.7. Add and Reorder Source

3. Click **Next**. The **Summary** window is shown.



Simulator : QuestaSim Qrun
Project Name : sim
Project Location : /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip
Simulation Stage : RTL
Simulation Files :
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/rtl/eth_10g_ip.sv
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/defines.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_apb_mst.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_axi4l_drv.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_axis_mst.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_data_checker.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_mem.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_top.sv
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tb_xgmii_mst.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/rst_sync.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/ten_gbe_pll.v
    /home/nsamsud1/my_designs/eth_10g_demo/eth_10g_ip/eth_10g_ip/testbench/tod_counter.v
Simulation Libraries :

☑ Launch Simulator GUI
☑ Design Optimization - Full Debug
☑ Add top-level signals to waveform display
☑ Run simulation
Default Run: 100.000 ns (0 means 'run -all')
Simulator Resolution: default

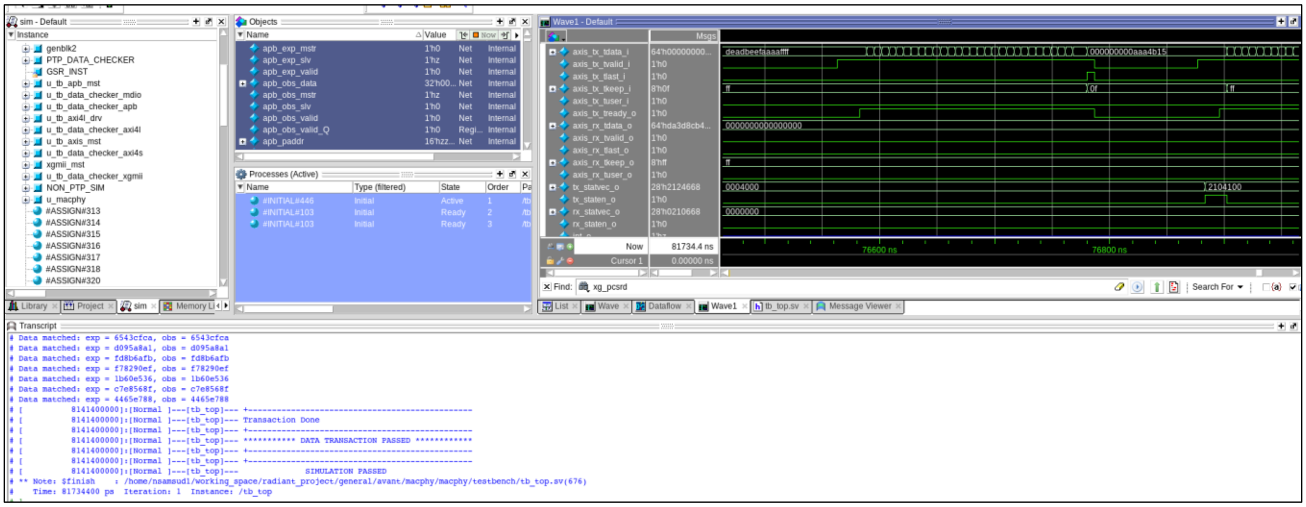
Figure 7.8. Summary Window

4. Click **Finish** to run the simulation.

   Note: It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant Software Suite.

7.5.1. Simulation Results

The following waveform shows an example of the simulation result.



```
# Data matched: exp = 6543cfca, obs = 6543cfca
# Data matched: exp = d095a8a1, obs = d095a8a1
# Data matched: exp = fd8b6afb, obs = fd8b6afb
# Data matched: exp = f78290ef, obs = f78290ef
# Data matched: exp = 1b60e536, obs = 1b60e536
# Data matched: exp = c7e8568f, obs = c7e8568f
# Data matched: exp = 4465e788, obs = 4465e788
# [          8141400000]:[Normal ]---[tb_top]--- +------------------------------------------
# [          8141400000]:[Normal ]---[tb_top]--- Transaction Done
# [          8141400000]:[Normal ]---[tb_top]--- +------------------------------------------
# [          8141400000]:[Normal ]---[tb_top]--- *********** DATA TRANSACTION PASSED ***********
# [          8141400000]:[Normal ]---[tb_top]--- +------------------------------------------
# [          8141400000]:[Normal ]---[tb_top]--- +------------------------------------------
# [          8141400000]:[Normal ]---[tb_top]---              SIMULATION PASSED
# ** Note: $finish    : /home/nsamsud1/working_space/radiant_project/general/avant/macphy/macphy/testbench/tb_top.sv(676)
#    Time: 81734400 ps  Iteration: 1  Instance: /tb_top
```

Figure 7.9. Simulation Results

8. Known Issues

8.1. MAC Pause Frame Receipt Function with a Non-XGMII Interface

8.1.1. Devices Affected

All devices.

8.1.2. Designs Affected

This issue affects only designs using the *MAC only* IP option with a non-XGMII (16-bit GMII, 8-bit GMII, or 4-bit MII) interface.

The following design options are not affected:

- *MAC only* IP option with XGMII interface
- *MAC + PHY* IP option
- *PHY only* IP option

8.1.3. Pause Frame Reception with Non-XGMII Interface

Upon the receipt of a valid pause frame, the Ethernet MAC sublayer must parse the pause frame and inhibit the transmission of data frame for the pause quanta or duration defined in the pause frame.

It is found that when the Ethernet MAC is configured with non-XGMII PHY interface, a valid pause frame received during an existing AXI4-Stream transaction from higher applications will not pause the transmit MAC from transmitting when the pause quanta counter has already started.

The transmission of pause frame through non-XGMII interface is not impacted.

This receipt of pause frame and subsequent pause on the Transmit MAC is also not impacted on the XGMII interface.

8.1.4. Planned Fix

Lattice intends to resolve this issue in the next revision of the 10G Ethernet IP core.

8.2. TX Custom Preamble Mode Function with Non-XGMII Interface

8.2.1. Devices Affected

CertusPro-NX devices (LFCPNX-100) and Avant Devices.

8.2.2. Designs Affected

This issue affects only designs using the *MAC only* IP option with a non-XGMII (16-bit GMII, 8-bit GMII, or 4-bit MII) interface on CertusPro-NX devices (LFCPNX-100) and Avant devices.

The following design options are not affected:

- *MAC only* IP option with XGMII interface
- *MAC + PHY* IP option
- *PHY only* IP option
- *MAC + PHY + 1588* IP option

8.2.3. TX Custom Preamble Mode with Non-XGMII Interface

When tx_pass_pream =1, you are allowed to supply the customize preamble and SFD value in the packet. The TX MAC must preserve the preamble field and passed it directly to the physical layer.
However, when the Ethernet MAC is configured with non-XGMII interface, the 1st byte of the data is missing.
It works fine in the XGMII interface.

8.2.4. Planned Fix

Lattice intends to resolve this issue in the next revision of the 10G Ethernet IP core.

8.3. Short Frame as Last Frame Limitation

8.3.1. Devices Affected

CertusPro-NX devices (LFCPNX-100) and Avant devices.

8.3.2. Designs Affected

This issue affects designs using the *MAC only* and *MAC + PHY* IP option on CertusPro-NX devices (LFCPNX-100) and Avant devices. The *PHY only* IP option is no affected.

8.3.3. Short Frame as Last Frame Limitation

When TX AXI interface sends long groups of packets back-to-back, and it ends with a short frame (less than 60 bytes), there is a possibility that the short frame remains in MAC and does not push to the PHY. Any subsequent transaction after that will push out the short frame.

8.3.4. Planned Fix

Lattice intends to resolve this issue in the next revision of the 10G Ethernet IP core.

Appendix A. Resource Utilization

Table A.1. Resource Utilization for CertusPro-NX and Avant Devices

Configuration	Registers	LUTs	EBRs	Target Device	Synthesis Tools
PHY only Host Interface == APB	36 (<1%)	103 (<1%)	0 (0%)	LFCPNX-100	Synplify Pro
MAC + PHY + 1588 Multicast Address Filtering == Enabled Statistics Counter Registers == Enabled Counter Width == 32 Host Interface == APB	15,234 (19%)	11,438 (14%)	16 (8%)	LFCPNX-100	Synplify Pro
PHY only Host Interface == AXI4-Lite	126 (<1%)	181 (<1%)	0 (0%)	LAV-AT-G70	Synplify Pro
PHY only Host Interface == APB	99 (<1%)	82 (<1%)	0 (0%)	LAV-AT-G70	Synplify Pro
MAC + PHY Host Interface == AXI4-Lite	3,966 (1%)	4,815 (1%)	3 (<1%)	LAV-AT-G70	Synplify Pro
MAC + PHY Multicast Address Filtering == Enabled Statistics Counter Registers == Enabled Counter Width == 32 Host Interface == AXI4-Lite	7,857 (2%)	8,548 (2%)	3 (<1%)	LAV-AT-G70	Synplify Pro
MAC + PHY Multicast Address Filtering == Enabled Statistics Counter Registers == Enabled Counter Width == 64 Host Interface == AXI4-Lite	11,377 (3%)	11,241 (3%)	3 (<1%)	LAV-AT-G70	Synplify Pro
MAC only Multicast Address Filtering == Enabled	3,876 (1%)	4,576 (1%)	3 (<1%)	LAV-AT-G70	Synplify Pro
MAC only Multicast Address Filtering == Enabled Statistics Counter Registers == Enabled Counter Width == 64	11,242 (3%)	10,962 (3%)	3 (<1%)	LAV-AT-G70	Synplify Pro

Appendix B. Clock and Reset Requirements (CertusPro-NX)

The following figure shows the suggested connection of xg_tx/rx_clk_i and xg_tx_clk_o. The GPLL module with 64/66 clock ratio is used to generate the 156.25 MHz clock input to xg_tx/rx_clk_i.

The GPLL module output lock signal is also recommended to be used as one of the PHY ready indicators before sending valid signals to PHY.

Refer to the following sample code:

```
assign sys_ready_o  = pll_lock_w & xg_rxval_o & xg_txrdy_o & xg_rx_blk_lock_o;
```

Note: For MAC reset, you do not need to use the PHY RX status signal.
For example:

```
assign mac_rst_n  = (pll_lock_w & xg_txrdy_o);
```

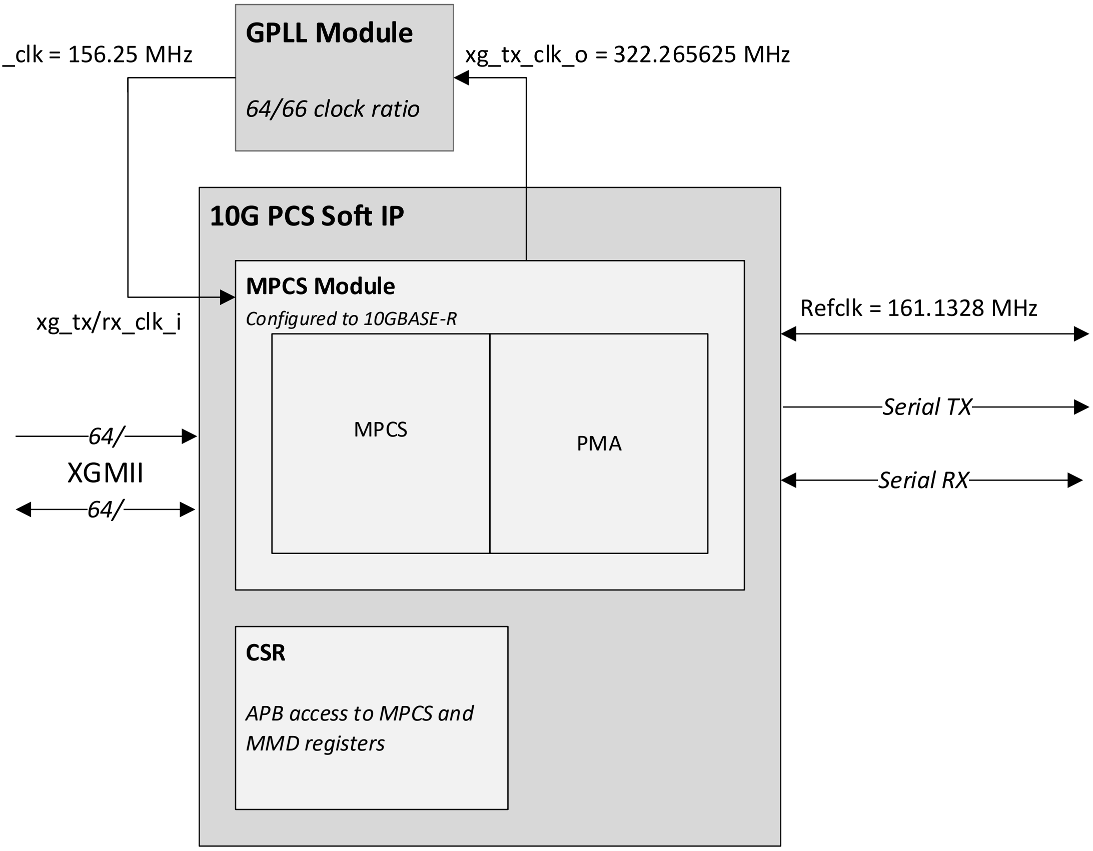


Figure B.1. 10G PCS IP Core Clock Setup

This setup is also available in the *eval_top.v* file under the eval folder.

References

- 10G Ethernet IP Release Notes (FPGA-RN-02031)
- 2.5G, 10G, and 25G Ethernet Driver API Reference (FPGA-TN-02375)
- CertusPro-NX 10G Ethernet PHY IP Core User Guide (FPGA-IPUG-02163)
- Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide (FPGA-UG-02039)
- Multi-Protocol PCS Module User Guide (FPGA-IPUG-02118)
- Lattice Radiant Timing Constraints Methodology Application Note (FPGA-AN-02059)
- Lattice Radiant Software web page
- 10Gb Ethernet MAC + PHY IP Core web page
- Avant-G web page
- Avant-X web page
- CertusPro-NX web page
- IP Cores and Reference Designs for Avant Devices web page
- Kits, Boards, and Demonstrations for Avant Devices web page
- IP Cores and Reference Designs for CertusPro-NX Devices web page
- Development Kits and Boards for CertusPro-NX Devices web page
- Lattice Insights web page for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 1.8, IP v.3.4.0, December 2025

Section	Change Summary
All	• Updated the IP version on the cover page. • Added LAV-AT-G30 and LAV-AT-X30 device support.
Introduction	• Updated Table 1.1. Summary of the 10G Ethernet IP. • Updated the Licensing and Ordering Information section. • Updated the Minimum Device Requirements section to include LAV-AT-G30 and LAV-AT-X30 devices.
Functional Description	• Updated Figure 2.2. 10GbE MAC + PHY IP Core SyncE Block Diagram. • Updated the Lane Merging (Avant Devices) section. • Updated the Clocking section.
Signal Description	• Updated Table 4.1. Signal Description—MAC + PHY. • Updated the sysbus_clk_i port name description in Table 4.3. Signal Description—PHY Only (Avant Devices). • Updated the xg_pcs_clkin_i port description in Table 4.5. Signal Description—MAC + PHY + 1588 (CertusPro-NX Devices). • Updated the xg_pcs_clkin_i port description in Table 4.7. Signal Description—MAC + PHY (CertusPro-NX Devices).
Register Description	Updated the description for the tx_pausreq register in Table 5.15. MAC_CTL Register.
Example Design	• Updated Step 4 in the IP Installation and Generation section. • Updated Figure 6.14. Configure Component.
Designing with the IP	• Updated Figure 7.2. Configure the User Interface of 10G Ethernet IP Core. • Updated the Timing Constraints section.
Known Issues	Added the Short Frame as Last Frame Limitation section.

Revision 1.7, IP v.3.3.1, September 2025

Section	Change Summary
All	• Updated the IP version on the cover page. • Renamed *10G Ethernet MAC + PHY IP* to *10G Ethernet IP*. • Removed MachXO5-NX device support.
Introduction	• Updated Table 1.1. Summary of the 10G Ethernet IP. • Updated Table 1.2. 10G Ethernet IP Support Readiness. • Updated the Features section.
Functional Description	Updated the following sections: • PHY (Avant Devices) section. • PHY (CertusPro-NX Devices) section.
IP Parameter Description	Updated the following tables: • Table 3.1. MAC + PHY Attributes (Avant Devices). • Table 3.3. PHY Only Attributes (Avant Devices). • Table 3.4. PHY Only Attributes (CertusPro-NX Devices Devices). • Table 3.5. MAC + PHY + 1588 Attributes (CertusPro-NX Devices). • Table 3.6. MAC + PHY Attributes (CertusPro-NX Devices).
Signal Description	Updated the following tables: • Table 4.3. Signal Description—PHY Only (Avant Devices). • Table 4.4. Signal Description—PHY Only (CertusPro-NX Devices).

Section	Change Summary
Register Description	Updated the following tables: • Table 5.23. Register Address Map for PHY. • Table 5.24. Register Address Map for PHY (CertusPro-NX Devices).
Example Design	• Updated the Generating Example Design section. • Updated the Hardware Testing section.
Designing with the IP	Updated Step 3 in the Generating and Instantiating the IP section.
Appendix A. Resource Utilization	Updated Table A.1. Resource Utilization for CertusPro-NX and Avant Devices.
References	Updated the document title from *10G Ethernet MAC + PHY IP Release Notes* to *10G Ethernet IP Release Notes*.

Revision 1.6, IP v.3.3.0, June 2025

Section	Change Summary
All	Updated the IP version on the cover page.
Introduction	• Updated Table 1.1. Summary of the 10G Ethernet IP. • Updated Table 1.2. 10G Ethernet IP Support Readiness. • Updated the Features Features section. • Updated Table 1.3. Ordering Part Number. • Updated Table 1.4. Minimum Device Requirements for 10G Ethernet IP Core.
Functional Description	• Updated the IP Architecture Overview sub section of MAC + PHY (Avant Devices). • Updated the reference document for details on MPPHY foundation IP to *the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).* • Added the following sentence to the Custom Preamble Passing section: *Note: Custom preamble mode is only supported by XGMII interface.* • Added the Lane Merging section. • Updated the PHY (CertusPro-NX and MachXO5-NX Devices) section. • Updated the MAC + PHY + 1588 (CertusPro-NX and MachXO5-NX Devices) section. • Added the MAC + PHY (CertusPro-NX and MachXO5-NX Devices) section. • Added the following note to the Clocking Overview section: *Note: Make sure that the GPLL being used does not have Fractional-N Divider mode enabled.* • Updated the title of the PHY + MAC + 1588 Reset Sequence (CertusPro-NX Devices) section. • Added the Latency section.
IP Parameter Description	• Updated the following tables: • Table 3.1. MAC + PHY Attributes (Avant Devices) • Table 3.3. PHY Only Attributes (Avant Devices). • Updated the PHY Only (CertusPro-NX and MachXO5-NX Devices) section. • Updated the MAC + PHY + 1588 (CertusPro-NX and MachXO5-NX Devices) section. • Added the MAC + PHY (CertusPro-NX and MachXO5-NX Devices) section.
Signal Description	• Added the MAC + PHY Signals (CertusPro-NX and MachXO5-NX Devices) section. • Updated the title of the PHY Only Signals (CertusPro-NX and MachXO5-NX Devices) section. • Updated the title of the MAC + PHY + 1588 Signals (CertusPro-NX and MachXO5-NX Devices) section.
Register Description	• Updated the following tables: • Table 5.4. TX_CTL Register • Table 5.5. RX_CTL Register • Updated the title of the PHY Registers (CertusPro-NX and MachXO5-NX Devices) section. • Updated the title of the MAC + PHY+ 1588 Registers (CertusPro-NX and MachXO5-NX Devices) section.
Example Design	• Updated the list of example designs in this section.

Section	Change Summary
	- Updated the GPLL section. - Updated the Versa Example Design Components (Avant Devices and CertusPro-NX Devices) section. - Updated the IP Installation and Generation section. - Updated the Importing Example Design Files to a Project section. - Updated the introductory sentence in the Example Design Simulation section. - Updated the figure title for Figure 6.30. Set tb_top* as Top Module - Updated the Hardware Testing section.
Designing with the IP	Updated Figure 7.9. Simulation Results.
Known Issues	- Updated the MAC Pause Frame Receipt Function with a Non-XGMII Interface section. - Added the TX Custom Preamble Mode Function with Non-XGMII Interface section.
Appendix A. Resource Utilization	Updated the following tables: - Table A.1. Resource Utilization for Avant Devices. - Table A.2. Resource Utilization for CertusPro-NX and Avant Devices.
References	Updated the title of the following reference in this section: *2.5G, 20G, and 2.5GEthernet MAC + PHY IP Release Notes*.

Revision 1.5, IP v.3.2.0 (LAV-AT-G/X70, LFCPNX-100), v2.1.2 (LAV-AT-G/X70ES), February 2025

Section	Change Summary
Functional Description	- Updated the IP Architecture Overview section. - Updated the Design Architecture section.
Signal Description	Updated the following tables: - Table 4.1. Signal Description—MAC + PHY. - Table 4.2. Signal Description—MAC Only. - Table 4.3. Signal Description—PHY Only (Avant Devices). - Table 4.4. Signal Description—PHY Only (CertusPro-NX Devices). - Table 4.5. Signal Description—MAC + PHY + 1588 (CertusPro-NX Devices).
Register Description	Updated Table 5.21. Summary of Statistics Counters.
Designing with the IP	Updated the Running Functional Simulation section.

Revision 1.4, IP v.3.2.0 (LAV-AT-G/X70, LFCPNX-100), v2.1.2 (LAV-AT-G/X70ES), December 2024

Section	Change Summary
All	Added IP version on the cover page.
Introduction	- Updated the IP version in Table 1.1. Summary of the 10G Ethernet IP. - Added IP Changes in Table 1.1. Summary of the 10G Ethernet IP. - Added the IP Support Summary section. - Renamed the section title IP Validation Summary to Hardware Support and updated the content in the Hardware Support section. - Updated the following feature for MAC in the Features section: *Supports VLAN and Jumbo frames of 9,600 bytes*
Functional Description	- Updated the PHY Delay Value section. - Added Figure 2.2. 10GbE MAC + PHY IP Core SyncE Block Diagram. - Updated the following figures with notes in the Receive AXI4-Stream Interface section: - Figure 2.7. Normal Frame Reception. - Figure 2.8. Back-to-back Frames Reception. - Figure 2.9. Frame Reception with In-Band FCS Passing. - Figure 2.10. Reception with Custom Preamble. - Updated the following figures with notes in the Transmit AXI4-Stream Interface section: - Figure 2.12. Default Normal Frame Transmission.

Section	Change Summary
	• Figure 2.13. Transmission with In-Band FCS Passing. • Figure 2.14. Transmission with Custom Preamble Passing.
IP Parameter Description	Updated the following tables: • Table 3.1. MAC + PHY Attributes. • Table 3.2. MAC Only Attributes. • Table 3.3. PHY Only Attributes (Avant Devices). • Table 3.5. MAC + PHY + 1588 Attributes (CertusPro-NX Devices)
Signal Description	• Added the following signals in Table 4.2. Signal Description—MAC Only: • xg_pcsrdy_i • rx_pause_cntr_o • rx_pause_req_o • rx_pause_time_o • Updated the description for the following signals in Table 4.3. Signal Description—PHY Only (Avant Devices): • xg_txrdy_o • xg_rxrdy_i
Register Description	• Added the following registers in Table 5.21. Summary of Statistics Counters: • TX_STAT_PKT_LNGTH_ACCU_0 • TX_STAT_PKT_LNGTH_ACCU_1 • RX_STAT_PKT_LNGTH_ACCU_0 • RX_STAT_PKT_LNGTH_ACCU_1 • Minor formatting changes to sub-section headings. • Removed the following registers from Table 5.41. TSU Configuration Registers: • TX PPS Delay Register • RX PPS Delay Register • Added the following note for Table 5.5. RX_CTL Register: *Note: If the L/T value is less than 46 bytes, it detects as short frame, and it will not be dropped. If the L/T value is less than 46 bytes but the payload is more than the defined value, then the extra payload will be treated as the padded byte.* • Removed the following registers from Table 5.41. TSU Configuration Registers: • TX PPS Delay Register • RX PPS Delay Register
Example Design	• Updated the introductory paragraph in this section. • Added the following key feature to the Overview of the Example Design and Features section: *Supports continuous and non-continuous mode packet transmission (Versa example design only)* • Updated the following sections: • Example Design Components. • Generating Example Design. • Example Design Simulation. • Hardware Testing. • Removed the *Running Functional Simulation* section.
Known Issues	Removed the *CertusPro-NX RX PTP VLAN IPv6 Packet Reception* section.
Appendix A. Resource Utilization	Updated Table A.1. Resource Utilization for Avant Devices (ES) and Table A.2. Resource Utilization for CertusPro-NX and Avant Devices (non ES).
References	• Added the following references to this section: • 10G Ethernet MAC + PHY IP Release Notes • 10G Ethernet Driver API Reference • IP Cores and Reference Designs for Avant Devices web page • Kits, Boards, and Demonstrations for Avant Devices web page • Replaced the following references with 10Gb Ethernet MAC + PHY IP Core web page:

Section	Change Summary
	• 10G Ethernet MAC IP Core web page • 10G Ethernet PCS IP Core web page

Revision 1.3, September 2024

Section	Change Summary
Abbreviations in This Document	Added *LINTR* to the abbreviation list.
Introduction	• Updated the IP version in Table 1.1. 10G Ethernet IP Core Quick Facts and Table 1.3. IP Validation Level. • Updated the following sub-title in the Features section: *MAC + PHY + 1588 (CertusPro-NX Devices)*
Functional Description	• Updated the PHY (Avant Devices) IP Architecture Overview section. • Added a reference to the MPCS Module User Guide in the PHY (CertusPro-NX Devices) IP Architecture Overview section. • Updated the PHY (CertusPro-NX Devices) Block Diagram section. • Updated Figure 2.28. 64b66b PCS Loopback Diagram in the PCS Loopback section. • Updated Figure 2.30. 10GbE MAC + PHY + 1588 IP Core Block Diagram. • Updated Figure 2.37. 10G Clock Network Diagram for Avant Devices. • Updated the following note 1 in the Timestamp Unit section: *At the RX side, the MAC AXIS output contains PTP and non-PTP packets. Therefore, muxing logic for PTP and non PTP must be handled by user logic if required. The ToD counter is done in user logic as well.* • Updated the following key features description in the Timestamp Unit section: *Correction field calculation for end-to-end transparent clock (E2E TC) (TC^2)* • Updated the following sentence in the PCS Loopback section. *In this mode, the 16-bit input data of the RX path comes from the TX path.* • Updated step 6 in the PTP Synchronization section. • Updated the PHY Delay Value section. • Added Table 2.9. PPS Delay Values. • Updated the address value to *0xA004* in Table 2.2. AXI4-Lite to PCS Address and Data Conversion.
Signal Description	• Added the following signals to Table 4.1. Signal Description—MAC + PHY: • pma_rx0_sigdet_hf_o • pma_rx0_sigdet_lf_o • Added the following signals to Table 4.3. Signal Description—PHY Only (Avant Devices): • pma_rx0_sigdet_hf_o • pma_rx0_sigdet_lf_o • Updated the clock output value for the following signals in Table 4.4. Signal Description—PHY Only (CertusPro-NX Devices): • xg_tx_clk_o • xg_rx_clk_o • Added the Clock Signal for MAC + PHY + 1588 Signals section. • Updated the description for rx_timestamp_rd_data and tx_timestamp_rd_data in Table 4.5. Signal Description—MAC + PHY + 1588 (CertusPro-NX Devices).
Register Description	Updated the TSU Configuration Registers section.
Example Design	• Updated step 4 in the Importing Example Design Files to a Project section. • Added the following sentence in the Example Design Simulation section. • Updated Figure 6.41. Programmer GUI in the Hardware Testing section.
Known Issues	Added the CertusPro-NX RX PTP VLAN IPv6 Packet Reception section.
Appendix B. Clock and Reset Requirements (CertusPro-NX)	• Updated the following code example: *assign mac_rst_n = (pll_lock_w & xg_txrdy_o);* • Updated the xg_tx_clk_o clock output value in Figure B.1. 10G PCS IP Core Clock Setup.

Revision 1.2, June 2024

Section	Change Summary
Introduction	• Updated Table 1.1. 10G Ethernet IP Core Quick Facts. • Updated Features section. • Added the following new sections: • IP Validation Summary section. • Minimum Device Requirements section.
Functional Description	• Removed the following sentence from the MAC section: *The MAC Only option is only applicable to CertusPro-NX devices (LFCPNX-100).* • Updated the following reference to a user guide in the IP Architecture Overview section: *For descriptions of the 10GbE PCS IP core on CertusPro-NX devices (LFCPNX-100), refer to the* CertusPro-NX 10Gb Ethernet PHY IP Core User Guide (FPGA-IPUG-02163). • Updated the term *STA master* to *STA manager* in the MDIO Transaction section. • Added the following new sections: • PHY (CertusPro-NX Devices) section. • MAC + PHY + 1588 (CertusPro-NX Devices) section. • Clocking section. • Updated and moved the Reset section to this section. • Updated Figure 2.37. 10G Clock Network Diagram for Avant Devices.
IP Parameter Description	• Updated and moved the following sections to this new section: • MAC + PHY (Avant Devices) section. • MAC Only section. • PHY Only (Avant Devices) section. • Added the following new sections: • PHY Only (CertusPro-NX Devices) section. • MAC + PHY + 1588 (CertusPro-NX Devices) section.
Signal Description	• Updated and moved the following sections to this section: • MAC + PHY Signals (Avant Devices) section. • MAC Only Signals section. • PHY Only Signals (Avant Devices) section. • Added the following new sections: • PHY Only Signals (CertusPro-NX Devices) section. • MAC + PHY + 1588 Signals (CertusPro-NX Devices) section. • Updated the term *slave* to *completer* in the following tables: • Table 4.1. Signal Description—MAC + PHY. • Table 4.2. Signal Description—MAC Only. • Table 4.3. Signal Description—PHY Only (Avant Devices). • Table 4.4. Signal Description—PHY Only (CertusPro-NX Devices). • Table 4.5. Signal Description—MAC + PHY + 1588 (CertusPro-NX Devices).
Register Description	• Updated and moved the following sections to this section: • MAC + PHY Registers (Avant Devices) section. • MAC Registers section. • PHY Registers (Avant Devices) section. • Added the following new sections: • PHY Registers (CertusPro-NX Devices) section. • MAC + PHY+ 1588 Registers (CertusPro-NX Devices) section.
Example Design	Added this new section.
Designing with the IP	• Updated the figures in the Generating and Instantiating the IP section. • Added the following note to Table 7.1. Generated File List: *Note: For other generated files in the Eval folder, refer to Table 6.4. Description of Generated Files in the Eval Folder.*

Section	Change Summary
	- Added the following sections: - Design Implementation section. - **Error! Reference source not found.** section. - Specifying the Strategy section. - Updated the figures in the Running Functional Simulation section. - Added the Simulation Results section.
Known Issues	- Removed a paragraph from the Pause Frame Reception with Non-XGMII Interface section. - Updated a sentence in the Planned Fix section.
Appendix A. Resource Utilization	Updated the resource utilization information.
Appendix B. Clock and Reset Requirements (CertusPro-NX)	Updated the title of this section and added a note.

Revision 1.1, March 2024

Section	Change Summary
Introduction	Removed *Lattice Synthesis Engine (LSE)* from Table 1.1. Summary of the 10G Ethernet IP .
Register Description	- Updated the MAC address values in 5.2.1.6 MAC_ADDR_0 and MAC_ADDR_1 Register section to the following: *For example, to set the MAC address to: AC-DE-48-00-00-80 would require writing 0x48_00_00_80 to address 0x014 (MAC_ADDR_0). 0xAC_DE to address 0x018 (MAC_ADDR_1).* - Removed the reference to the Appendix B section in 5.2.1.9 MC_TABLE_0 and MC_TABLE_1 Register section.
Appendix B. Code Listing for Multicast Bit Selection Hash Algorithm in C Language	Removed this section.

Revision 1.0, December 2023

Section	Change Summary
All	Initial release.

LATTICE
SEMICONDUCTOR®

www.latticesemi.com