TSEMAC & SGMII

Reference Design

FPGA-RD-02271-1.3

September 2024

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
ARP	Address Resolution Protocol
AXI	Advanced Extensible Interface
CRC	Cyclic Redundancy Check
FCS	Frame Check Sequence
GMII	Gigabit Media Independent Interface
LMMI	Lattice Memory Mapped Interface
LSE	Lattice Synthesis Engine
MAC	Media Access Controller
OFC	Optical Fiber Cable
PCS	Physical Coding Sublayer
PLL	Phase-Locked Loop
RTL	Register Transfer Level
RX	Receiver
SFD	Start of Frame Delimiter
SFP	Small Form-Factor Pluggable
SGMII	Serial Gigabit Media Independent Interface
TSEMAC	Tri-Speed Ethernet Media Access Control
TX	Transmitter
UDP	User Datagram Protocol

1. Introduction

The Lattice Semiconductor Tri-Speed Ethernet Media Access Control (TSEMAC) IP core supports the ability to transmit and receive data between a host processor and an Ethernet network. The main function of the TSEMAC IP core is to make sure that the media access rules specified in the IEEE 802.3 standards are met while transmitting a frame of data over the Ethernet. On the receive side, the TSEMAC IP core extracts the different components of a frame and transfers them to higher applications through an AXI4-Stream interface.

The SGMII PCS IP core provides GMII interface to the MAC and follows the Ethernet frame standard. It supports 8-bit of data and 8-bit of control signals for both transmit and receive path.

This reference design demonstrates an example of 1G/100M/10M Ethernet application using a TSEMAC IP core with a SGMII PCS IP core in loopback mode. A simple Ethernet packet generator is included to generate Ethernet packets to be transmitted on the transmitter (TX) and compare them with the received packets from the receiver (RX).

2. Features

The following lists the features for the TSEMAC IP core and the SGMII Gb Ethernet PCS IP core:

- Provides an Ethernet packet generator to generate and compare packets
- Supports operation speeds of 1 Gbps, 100 Mbps, and 10 Mbps
- Enables RISC-V to configure SFP PHY register to SGMII mode via I2C controller

For more information on the features for the TSEMAC IP core, refer to the TSEMAC IP User Guide (FPGA-IPUG-02084).
For more information on the features for the SGMII and Gb Ethernet PCS IP core, refer to the SGMII and Gb Ethernet PCS IP User Guide (FPGA-IPUG-02077).

3. Functional Description

This reference design instantiates and connects a TSEMAC IP core to a SGMII PCS IP core through the GMII interface. The SGMII PCS IP clock source of 125 MHz is provided through an external phase-locked loop (PLL) and 625 MHz clock is provided from the PLL when you use more than one SGMII in your design. The SGMII feedback clock *usr_clk_o* provides the TSEMAC, SGMII, pattern generator, checker RX, and TX clock source of 125 MHz. The SGMII PCS IP core will generate TX and RX clock enable signals according to the operation speed configured in the SGMII PCS IP and output to the TSEMAC, pattern generator, and checker. The serial interface is looped back externally.

A packet generator is included in the design to generate and send Ethernet packets to the TSEMAC IP core through the AXI4-Stream interface on the TX path. The same packets will be stored in a FIFO and compared with the Ethernet packets that are received from the TSEMAC IP core RX path. Figure 3.1 shows the basic block diagram and connections of the reference design.

Besides that, the design integrates a RISC-V SoC to configure the external (SFP) PHY register to SGMII mode via an I2C controller, as shown in Figure 3.2. The RISC-V SoC that includes a CPU and various peripherals is instantiated within the FPGA portion of CertusPro-NX and CrossLink-NX devices using the Lattice Propel Builder. The I2C Controller IP is programmed through software executed by the RISC-V processor, utilizing the Lattice Propel software to configure the external PHY register to SGMII mode. This configuration software executes automatically after you program the bitstream.

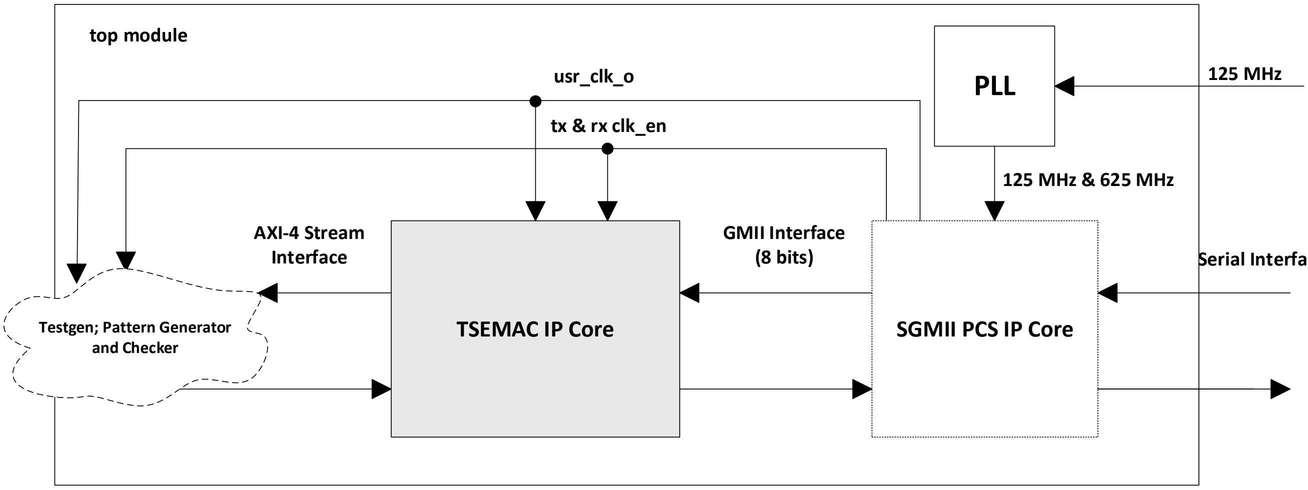


Figure 3.1. Reference Design Basic Diagram

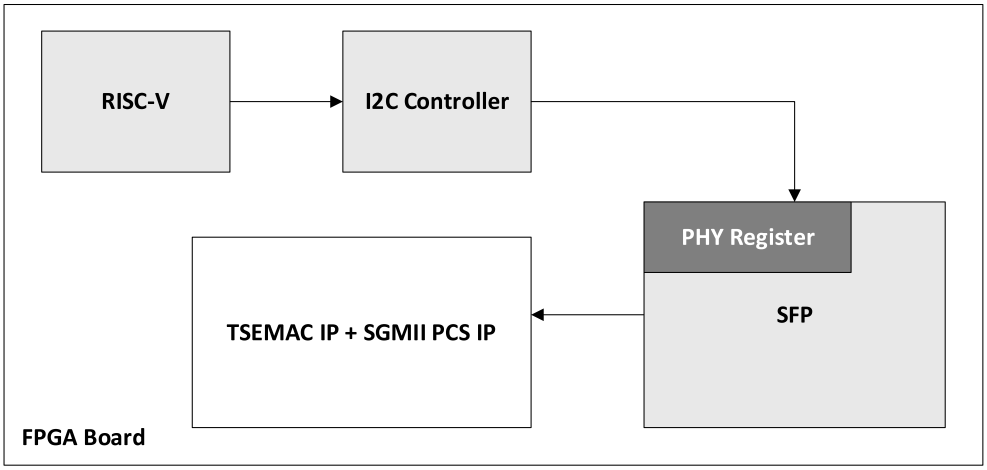


Figure 3.2. Overall Reference Design Block Diagram

The following table lists the top-level signal descriptions.

Table 3.1. Signal Description

<table>
<tr><th>Signal Name</th><th>I/O Type</th><th>I/O Width</th><th>Description</th></tr>
<tr><td>clk_125</td><td>I</td><td>1</td><td>125 MHz clock input for the system.</td></tr>
<tr><td>resetn_i</td><td>I</td><td>1</td><td>Active low reset for the reference design. Activated by pressing SW3 on board.</td></tr>
<tr><td>packet_gen_start</td><td>I</td><td>1</td><td>Initiates packet generator signal. Activated when toggling switch 5 in SW1 for CertusPro-NX devices and toggling switch 1 in SW1 for Crosslink-NX devices to ON position.</td></tr>
<tr><td>ser_rx_i</td><td>I</td><td>1</td><td>Serial receiver.</td></tr>
<tr><td>ser_tx_o</td><td>O</td><td>1</td><td>Serial transceiver.</td></tr>
<tr><td>led</td><td>O</td><td>8</td><td>Refer to Table 8.3 for the LED descriptions.</td></tr>
<tr><td>pll_lock_out</td><td>O</td><td>1</td><td>PLL locked signal.</td></tr>
<tr><td>SFP_DISABLE</td><td>O</td><td>1</td><td>Active high signal to disable the small form-factor pluggable (SFP). This signal is driven to zero on system-level reset de-assertion.</td></tr>
<tr><td>patgen_done_out</td><td>O</td><td>1</td><td>The pattern generator completes the number of packets assigned.</td></tr>
<tr><td>cmpFail_out</td><td>O</td><td>1</td><td>Asserts only when data mismatch is found.</td></tr>
<tr><td>uart_rxd_i</td><td>I</td><td>1</td><td>Receives communication from CertusPro-NX and Crosslink-NX devices and FTDI chip. Make sure jumpers 32 and 33 are closed to establish communication.</td></tr>
<tr><td>uart_txd_o</td><td>O</td><td>1</td><td>Transmits communication from CertusPro-NX and Crosslink-NX devices and FTDI chip. Make sure jumpers 32 and 33 are closed to establish communication.</td></tr>
<tr><td>i2c_c0_inst_scl_io_port</td><td>IO</td><td>1</td><td>Bidirectional data transfer for I2C.</td></tr>
<tr><td>i2c_c0_inst_sda_io_port</td><td>IO</td><td>1</td><td>Bidirectional data transfer for I2C.</td></tr>
<tr><td>rate_sel_i</td><td>I</td><td>2</td><td>Configures the operational speed.
<table>
<tr><th>rate_sel_i[1]</th><th>rate_sel_i[0]</th><th>Operational Rate</th></tr>
<tr><td>0</td><td>0</td><td>10 Mbps</td></tr>
<tr><td>0</td><td>1</td><td>100 Mbps</td></tr>
<tr><td>1</td><td>0</td><td>1 Gbps</td></tr>
</table></td></tr>
</table>

4. Design Architecture

The design architecture is divided into two sections. The first section details the use of a RISC-V SoC to configure the external PHY register via the I2C interface, while the second section covers the top module reference design.

Due to the variability in SFP modules, which may have different register settings for SGMII mode, this section highlights the feature that automatically adjusts the PHY register to SGMII mode via I2C. As an example, this reference design uses the Marvell Alaska 88E1111 PHY chip to demonstrate this feature. Configuring the PHY register to SGMII mode requires using the I2C interface to modify the extended PHY-specific status register at register 27. You may modify the software code in the Lattice Propel software to configure other PHYs. Figure 4.1 illustrates the I2C handling process for accessing the PHY register.

For detailed information on status or control signals and I2C bus, refer to the *Control Buses – I2C, UART, and SPI* section in the CertusPro-NX Versa Board Evaluation Board User Guide (FPGA-EB-02053).

The prerequisite of this design is the external PHY for the SFP module must be the Marvell Alaska 88E1111 to enable I2C configuration. This reference design has been validated with the FS, TechNxT, and Finisar SFP RJ45 transceiver modules. The following lists the details of the modules:

- FS SFP: Generic Compatible SFP SGMII 10/100/1000BASE-T Copper 100m RJ-45 Transceiver Module
- TechNxt SFP: TechNxt 1G copper RJ-45
- Finisar SFP: Finisar FCLF8522P2BTL SFP (mini-GBIC) Module, 1000BASE-T Gigabit Ethernet, hot-pluggable

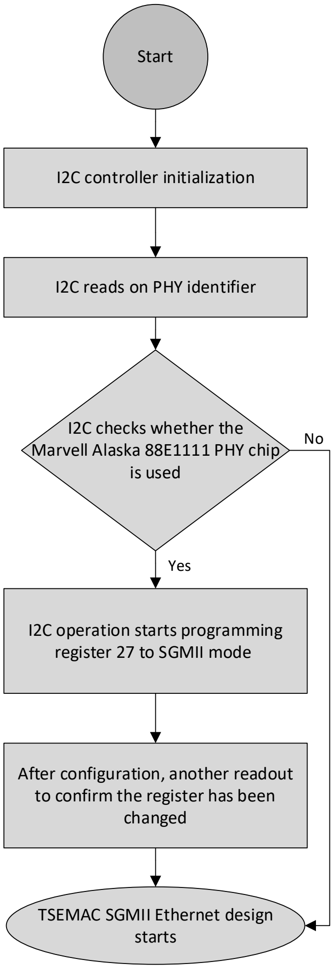

Start
I2C controller initialization
I2C reads on PHY identifier
I2C checks whether the Marvell Alaska 88E1111 PHY chip is used
No
Yes
I2C operation starts programming register 27 to SGMII mode
After configuration, another readout to confirm the register has been changed
TSEMAC SGMII Ethernet design starts

Figure 4.1. I2C Handling Flow

This section describes each module in the top module reference design, as shown in Figure 4.2.

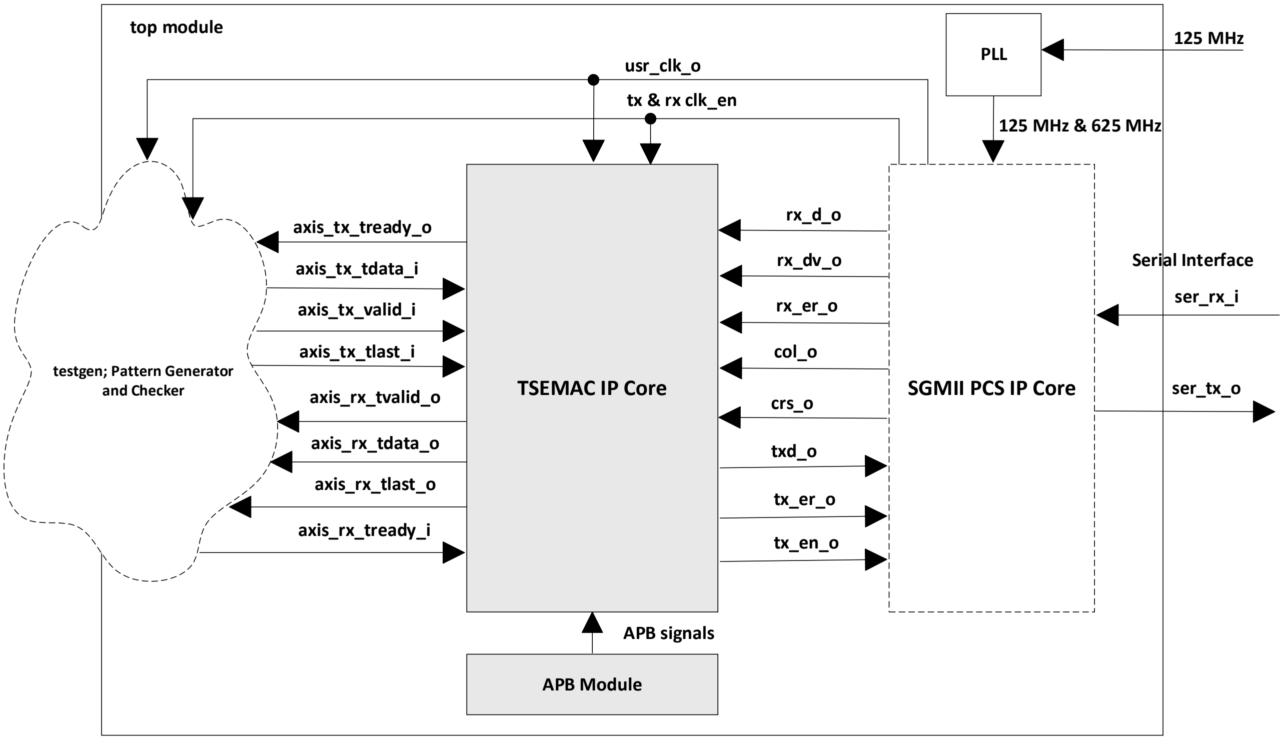


Figure 4.2. Reference Design Architecture

4.1. TSEMAC IP

The TSEMAC IP core can be configured to operate in either the Gigabit mode (1000Mbits/sec data rate) or the Fast Ethernet mode (10/100 Mbits/sec data rate). Operation in either Gigabit mode or Fast Ethernet mode is selected by setting an internal register bit. In this reference design, the operation mode is configurable through the top-level parameter OPERATION_SPEED.

The TSEMAC transmits and receives data between a host processor and an Ethernet network. The main function of the Ethernet MAC is to make sure that the media access rules specified in the IEEE 802.3 standard are met while transmitting a frame of data over the Ethernet. On the receiving side, the Ethernet MAC extracts the different components of a frame and transfers them to higher applications through the FIFO interface.

The data received from the G/MII interface is first buffered until sufficient data is available to be processed by the Receive MAC (RX MAC). The preamble and the start of frame delimiter (SFD) information are then extracted from the incoming frame to determine the start of a valid frame. The Receive MAC checks the address of the received packet and validates whether the frame can be received before transferring it into the FIFO. Only valid frames are transferred into the FIFO. This feature has the following two benefits: the systems need not re-calculate the frame check sequence (FCS) again when the frame is being transmitted, and it also keeps the receive MAC relatively simple. However, the Tri-Speed Ethernet MAC always calculates the cyclic redundancy check (CRC) to verify whether the frame was received error-free.

On the transmit side, the TX MAC is responsible for controlling access to the physical medium. The TX MAC reads data from an external client TX FIFO, formats the data into an Ethernet packet, and passes it to the G/MII module. The TX MAC reads data from the TX Client FIFO when the client indicates a packet is available, and the TX MAC is in its appropriate state. The TX MAC pre-fixes the preamble and the SFD information to the data and appends the FCS at the end of the data. In half-duplex operation, the TX MAC stores the first 64 bytes of data from the external FIFO in an internal buffer, to be used in re-transmitting data on collisions. The SGMII Easy Connect configuration option adds pins and logic for seamless connection to the Lattice's SGMII PCS IP core.

The preamble, SFD, and FCS are added by the TSEMAC. The Ethernet pattern sent by the TSEMAC is shown in the Ethernet frame format in Figure 4.3.

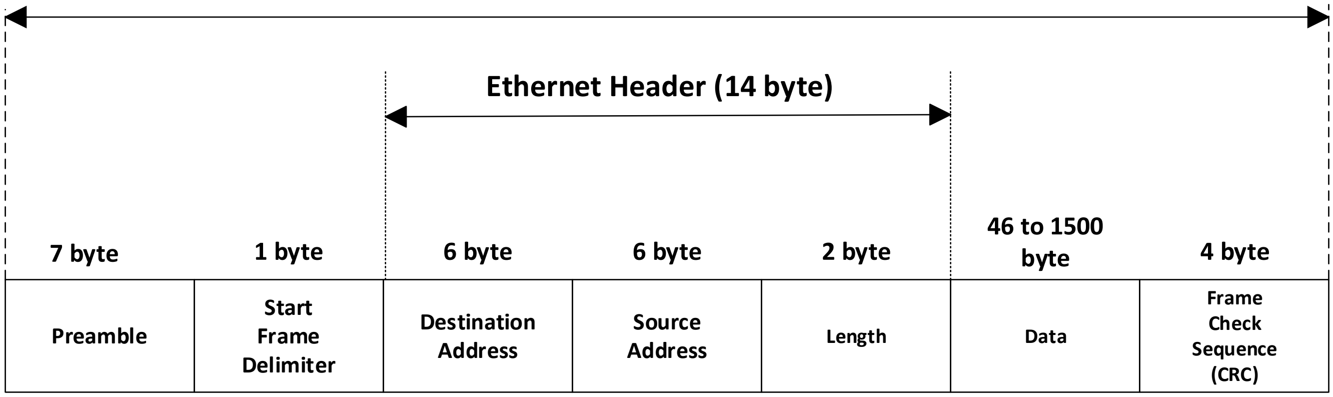


Figure 4.3. Ethernet Frame Structure

In this reference design, the TSEMAC configuration style is configured as SGMII Easy Connection Mode. With this configuration style, you can easily connect the SGMII with the TSEMAC and synchronize the clock of MII using the clk_enable signal that is generated by the SGMII module.

The TX FIFO is read when the packet is received through the AXI interface. The packet that is received from the SGMII is stored in the RX FIFO and read by the pattern generator through the AXI interface. The following table lists the port description of the TSEMAC IP in this reference design.

Table 4.1. TSEMAC IP Signal Description

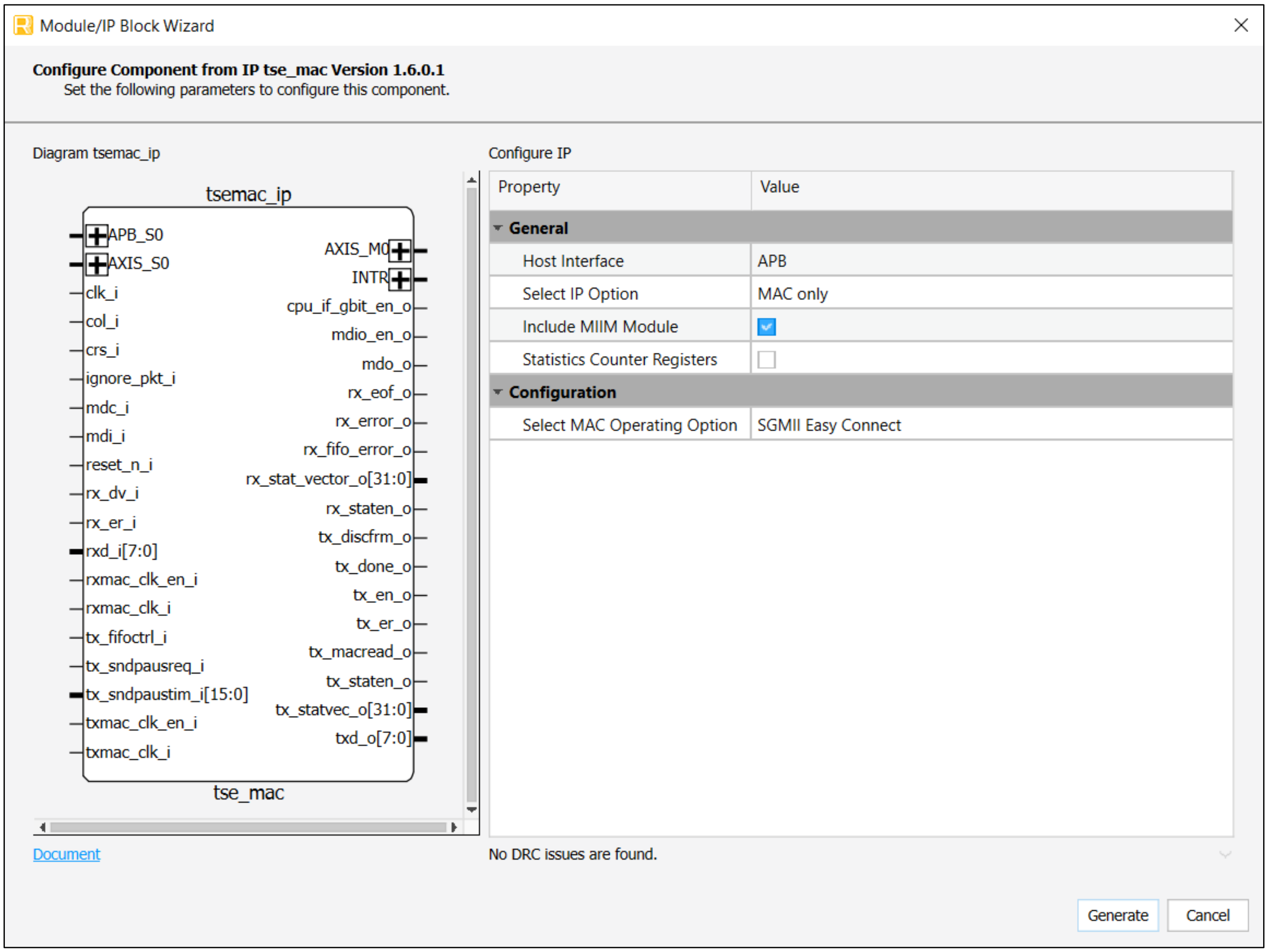
TSEMAC Signals	Description
APB Interface	
apb_paddr_i	APB Address signal. Size: Interface Address Width.
apb_prdata_o	APB Read Data signal. Size: Interface Data Width.
apb_pwdata_i	APB Write Data signal. Size: Interface Data Width.
apb_pready_o	APB Ready signal.
apb_pslverr_o	APB Completer Error signal.
apb_pwrite_i	APB Direction signal.
apb_psel_i	APB Select signal.
apb_penable_i	APB Enable signal.
AXI4-Stream Transmit Interface	
axis_tx_tdata_i	Data from the client. It is the primary payload that is used to provide the data that is passing through the interface.
axis_tx_tvalid_i	Data Validation. Read transaction is complete and axis_tx_tvalid_i contains the valid data.
axis_tx_tlast_i	Indicates the boundary of a packet.
axis_tx_tready_o	Ready to Start Transaction. Indicates that the MAC can accept a transfer in the current cycle.
AXI4-Stream Receiver Interface	
axis_rx_tdata_o	Data from PHY. It is the primary payload that is used to provide the data that is passing through the interface.

TSEMAC Signals	Description
axis_rx_tvalid_o	Data Validation. Indicates that the Transmitter is driving a valid transfer. A transfer takes place when both this signal and axis_rx_tready_i are asserted.
axis_rx_tlast_o	Indicates the boundary of a packet.
axis_rx_tready_i	Ready to Start Transaction. Indicates that the client can accept a transfer in the current cycle.
Interrupt	
int_o	Interrupt. Stays high if any enabled interrupt is pending.
Transmit MAC Control and Status Signal	
tx_sndpaustim_i	PAUSE Frame Timer. This signal indicates the PAUSE time value that should be sent in the PAUSE frame. Tie to 0 in the reference design.
tx_sndpausreq_i	PAUSE Frame Request. When asserted, the TSEMAC IP core transmits a PAUSE frame. This is also the qualifying signal for the tx_sndpausetim_i bus. Tie to 0 in the reference design.
tx_fifoctrl_i	FIFO Control Frame. This signal indicates whether the current frame in the Transmit FIFO is a control frame or a data frame. Tie to 0 for Normal Frame.
tx_staten_o	Transmit Statistics Vector Enable. When asserted, the contents of the statistics vector bus tx_statvec_o is valid.
tx_macread_o	Transmit FIFO Read. This is the TSEMAC IP core Transmit FIFO read request, asserted by the TSEMAC IP core when it intends to read the FIFO.
tx_statvec_o	Transmit Statistics Vector. This bus includes useful information about the frame that is just transmitted.
tx_done_o	Transmit Done. This signal is asserted for one clock cycle after transmitting a frame if no errors are present in transmission.
tx_discfrm_o	Discard Frame. This signal is asserted at the end of a frame transmit process if the TSEMAC IP core detects an error. The possible conditions are: • A FIFO under-run • Late collision (10/100 mode only) • Excessive Collisions (10/100 mode only) The user application normally moves the pointer to the next frame in these conditions.
Receive MAC Control and Status Signal	
rx_stat_vector_o	Receive Statistics Vector. This bus indicates the events encountered during frame reception.
ignore_pkt_i	Ignore Next Packet. The host asserts this signal to prevent a Receive FIFO Full condition. The Receive MAC continues dropping packets as long as this signal is asserted. This is an asynchronous signal.
rx_error_o	Receive Packet Error. When asserted, this signal indicates the packet contains error(s). This signal is qualified with the rx_eof_o signal. The rx_error_o signal is asserted for any of the following three conditions: • The rx_er* signal on the GMII is asserted by the PHY during frame reception • There are RX FCS errors on received frames • There is a length check error on the received frame
rx_eof_o	Receive the End of Frame flag.
rx_fifo_error_o	Receive FIFO Error. This signal is asserted when the RX FIFO is full, and the RX FIFO is being written to by the RX MAC. When this error signal is asserted, the RX MAC stops writing to the RX FIFO. The rx_fifo_error_o signal is de-asserted when the end of packet exits the RX FIFO.

TSEMAC Signals	Description
Management Interface (Available in MAC Only Option and When MIIM is Selected)	
mdi_i	Management Data Input. Used to transfer information from the PHY to the management module.
mdo_o	Management Data Output. Used to transmit information from the management module to the PHY.
mdio_en_o	Management Data Out Enable. Asserted whenever mdo_o is valid. This can be used to implement a bi-directional signal for mdi_i and mdo_o.
GMII Interface	
txd_o[7:0]	txd_o[7:0] - Transmit Data Sent to the PHY Interface. These GMII Tx data outputs go to the SGMII PCS IP core (SGMII Easy Connect option) or to the 1G GMII PHY interface (Gigabit interface option).
tx_en_o	Transmit Enable. Asserted by the TSEMAC IP core to indicate the txd_o bus contains valid frame.
tx_er_o	Transmit Error. Asserted when the TSEMAC IP core generates a coding error on the byte currently being transferred.
rx_dv_i	Receive Data Valid. Indicates the data on the rxd_o bus is valid.
rxd_i	rxd_i[7:0] - Receive Data from the PHY Interface. These GMII Rx data inputs (valid whenever rx_dv_i is asserted) come from the SGMII_PCS IP core (SGMII Easy Connect option) or from the 1G GMII PHY interface (Gigabit MAC option).
rx_er_i	rx_er_i - Receive Data Error from the PHY Interface.
col_i	Collision. This active-high signal indicates a collision occurred during transmission. This signal is valid for half-duplex operation in Fast Ethernet (10/100) for the Classic and SGMII Easy Connect options only. Otherwise, it is ignored.
crs_i	Carrier Sense. This signal, when logic high, indicates the network has activity. Otherwise, it indicates the network is idle. This signal is valid for half-duplex operation in Fast Ethernet (10/100) for the Classic and SGMII Easy Connect options only.
Clock and Reset	
rxmac_clk_i	Always 125 MHz in Easy Connect mode.
txmac_clk_i	Always 125 MHz in Easy Connect mode.
clk_i	Host Interface (APB, AHB-Lite, or AXI4-Lite) Clock.
reset_n_i	Reset. This is an active low asynchronous signal that resets the internal registers and internal logic. When activated, the I/O signals are driven to their inactive levels.
txmac_clk_en_i	TX Clock Enable. This input signal is a clock enable used only in the SGMII Easy Connect option. The SGMII_PCS IP core drives this signal. The clock enable is always high for 1G operation. For 100 Mbps operation the clock enable is asserted high once every ten (125 MHz) clocks, and for 10 Mbps operation the clock enable is asserted high once every hundred (125 MHz) clocks.
rxmac_clk_en_i	RX Clock Enable. This input signal is a clock enable used only in the SGMII Easy Connect option. The SGMII_PCS IP core drives this signal. The clock enable is always high for 1G operation. For 100 Mbps operation the clock enable is asserted high once every ten (125 MHz) clocks, and for 10 Mbps operation the clock enable is asserted high once every hundred (125 MHz) clocks.
cpu_if_gbit_en_o	CPU Interface 1G Mode Enabled Indication. This signal, when high, is an indication from the CPU interface that the 1G mode is enabled. This signal reflects the state of bit 0 of the MAC mode register.

4.1.1. TSEMAC IP Wizard Configuration

In the reference design, the TSEMAC IP is configured as shown in Figure 4.4.



Module/IP Block Wizard

Configure Component from IP tse_mac Version 1.6.0.1
Set the following parameters to configure this component.

Diagram tsemac_ip

Configure IP

Property	Value
General	
Host Interface	APB
Select IP Option	MAC only
Include MIIM Module	☑
Statistics Counter Registers	☐
Configuration	
Select MAC Operating Option	SGMII Easy Connect

No DRC issues are found.

Figure 4.4. TSEMAC IP Wizard Configuration

The following lists the configuration settings for the TSEMAC IP:

- Host Interface: APB
- Included MIIM Module: Yes
- Select MAC Operating Option: SGMII Easy Connect

4.2. SGMII IP

The Lattice SGMII and Gb Ethernet PCS IP core implements the PCS functions of both the Cisco SGMII and the IEEE 802.3z (1000BASE-X) specifications. The PCS mode is pin selectable. This IP core can be used in bridging applications and/or PHY implementations.

In the reference design, the SGMII style is configured as TSEMAC Easy Connection. This will enable the clk_enable signals for TX and RX, which can be provided to the TSEMAC directly to control the operating speed.

The following table lists the port description of the SGMII IP.

Table 4.2. SGMII IP Signal Description

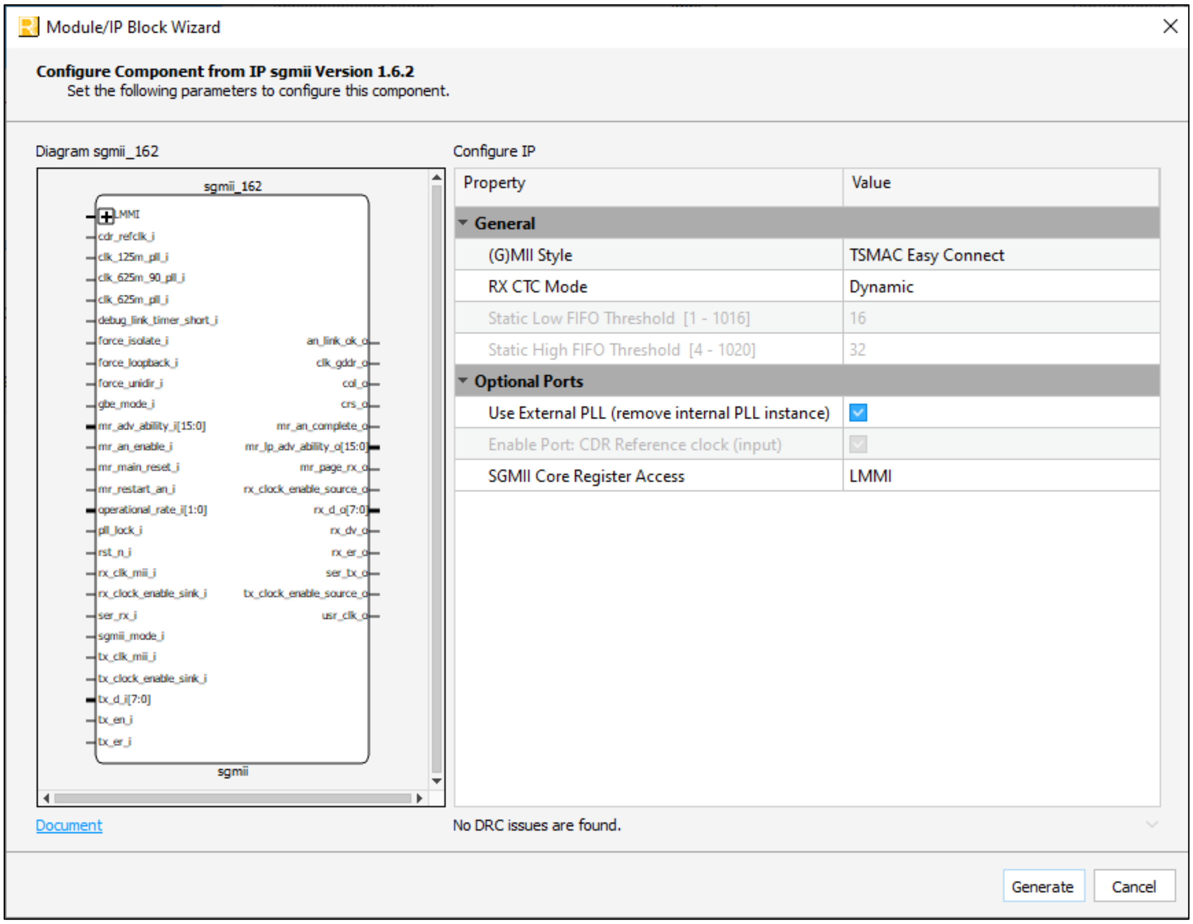
SGMII Signals	Description
Clock and Reset	
tx_clock_enable_source_o	Transmit Clock Enable Source – This signal is only present when the IP core is generated using the TSEMAC Easy Connect (G)MII option.
rx_clk_mii_i	Data is sampled on the rising edge of this clock. For the TSEMAC Easy Connect option, this clock is always 125 MHz.
tx_clock_enable_sink_i	Transmit Clock Enable Sink – This signal is only present when the IP core is generated using the TSEMAC Easy Connect (G)MII option. This signal is used in combination with the transmit 125 MHz clock to regulate the flow of transmit (G)MII data. When the clock enable is high and the transmit clock edge rises, (G)MII data is sampled.
rx_clock_enable_source_o	Receive Clock Enable Source - This signal too only present when the IP core is generated using the TSEMAC Easy Connect (G)MII option.
tx_clk_mii_i	Data is launched on the rising edge of this clock. For TSEMAC Easy Connect option, this clock is always 125 MHz.
rx_clock_enable_sink_i	Receive Clock Enable Sink – This signal is only present when the IP core is generated using the TSEMAC Easy Connect (G)MII option. This signal is used in combination with the receive 125 MHz clock to regulate the flow of receiving (G)MII data. When the clock enable is high and the receive clock edge rises, (G)MII data is launched.
rst_n_i	Reset – Active low global reset.
cdr_refclk_i	CDR Reference Clock – 125 MHz user-provided CDR reference clock input. Note that this signal is only available for non-LAV-AT devices and if Enable Port: CDR reference clock is enabled. This input clock should be coming from the device's Generic PLL.
clk_125m_pll_i	125 MHz PLL Clock – 125 MHz clock input. Note that this signal is only available if the *Use External PLL* option is enabled in the GUI.
clk_625m_pll_i	625 MHz PLL Clock – 625 MHz clock input. Note that this signal is only available if the *Use External PLL* option is enabled in the GUI.
clk_625m_90_pll_i	90-degree Phase Shift 625 MHz PLL Clock – 625 MHz clock input with 90-degree phase shift. Note that this signal is only available if the *Use External PLL* option is enabled in the GUI.
usr_clk_o	User Clock – 125 MHz clock from ECLKDIV output. Note that this signal is only present when the IP core is generated using the TSEMAC Easy Connect (G)MII option.
clk_gddr_o	DDR Clock – assumes an LVDS buffer.
lmmi_clk_i	LMMI Clock.
mdc_i	Management Data Clock – Clock source for the serial management interface.
GMII	
tx_d_i	Transmit Data – Incoming (G)MII data. Note that this port's behavior varies depending on the (G)MII option used when generating the IP core. For Classic mode, when the (G)MII data rate is 1 Gbps, all 8 bits of tx_d_i are valid. However, for 100 Mbps and 10 Mbps, only bits 3:0 of tx_d_i are valid. For the TSEMAC Easy Connect mode all 8 bits of tx_d_i are valid for all (G)MII data rates (1 Gbps, 100 Mbps, 10 Mbps).
tx_en_i	Transmit Enable – Active high signal; asserts when incoming data is valid.
tx_er_i	Transmit Error – Active high signal used to denote transmission errors and carrier extension on incoming (G)MII data port.
rx_d_o	Receive Data – Outgoing (G)MII data. Note that this port's behavior varies depending on the (G)MII option used when generating the IP core. For Classic mode, when the (G)MII data rate is 1 Gbps, all 8 bits of rx_d_o are valid. However, for 100 Mbps and 10 Mbps, only bits 3:0 of rx_d_o is valid. For the TSEMAC Easy Connect mode, all 8 bits of rx_d_o are valid for all (G)MII data rates (1 Gbps, 100 Mbps, 10

SGMII Signals	Description
	Mbps).
rx_dv_o	Receive Data Valid – Active high signal, asserts when outgoing data is valid.
rx_er_o	Receive Error – Active high signal used to denote transmission errors and carrier extension on outgoing (G)MII data port.
col_o	Collision Detect – Active high signal, asserts when tx_en_i and rx_dv_o are active at the same time.
crs_o	Carrier Sense Detect – Active high signal, asserts when rx_dv_o is high.
Management	
force_loopback_i	Force PCS Loopback – Active high signal that activates the PCS loopback function. When asserted, the 10-bit code-group output of the transmit state machine is looped back to the 10-bit code-group input of the receive state machine. When de-asserted, the loopback function is deactivated.
operational_rate_i	Operational rate – When the core operates in SGMII PCS mode, this port controls the regulation rate of the rate adaptation circuit blocks as follows: 10 = 1 Gbps rate 01 = 100 Mbps rate 00 = 10 Mbps rate Note that in Gigabit Ethernet PCS mode, the rate adaptation blocks always operate at the 1 Gbps rate regardless of the settings on the operational_rate_i control pins.
force_unidir_i	Force PCS Unidirectional Mode – Active high signal that activates the PCS unidirectional mode. When asserted, the transmit state machine path between the TX (G)MII input and the TX 10-bit code-group output will remain operational regardless of what happens on the RX data path. (Normally RX loss of sync, invalid code-group reception, auto-negotiation restarts can force the transmit state machine to temporarily ignore inputs from the TX (G)MII port). When de-asserted, the unidirectional mode is deactivated. This signal is optional. If you choose not to use the unidirectional function, then this signal should be tied low.
mr_adv_ability_i	Advertised Ability – Configuration status transmitted by PCS during auto-negotiation process. This signal must not change during auto-negotiation.
mr_an_enable_i	Auto-Negotiation Enable – Active high signal that enables auto-negotiation state machine to function. This signal must not change during auto-negotiation.
mr_main_reset_i	Main Reset – Active high signal that forces all PCS state machines to reset.
mr_restart_an_i	Auto-Negotiation Restart – Active high signal that forces auto-negotiation process to restart.
Serial Interface	
ser_tx_o	Serial Transmit Data – DDR data. Assumes an LVDS buffer.
ser_rx_I	Serial Receive Data – DDR data. Assumes an LVDS buffer.
LMMI	
lmmi_resetn_i	LMMI Active low reset.
lmmi_request_i	Start transaction.
lmmi_offset_i	Register offset, starting at offset 0
lmmi_wr_rdn_i	Write and read operations enable signal 1’b0 - Read 1’b1 - Write
lmmi_wdata_i	Output data bus.
lmmi_rdata_o	Input data bus.
lmmi_rdata_valid_o	Read transaction is complete and lmmi_rdata_o contains valid data.
lmmi_ready_o	IP is ready to receive a new transaction. This is always asserted (tied to 1’b1).
Miscellaneous	
sgmii_mode_i	SGMII Mode – Controls the behavior of the auto-negotiation process when the core is

SGMII Signals	Description
	operating in SGMII mode. 0 = operates as MAC-side entity, 1 = operates as PHY-side entity.
gbe_mode_i	Gigabit Ethernet Mode – Controls the core's PCS function. 0 = operates as SGMII PCS, 1 = operates as Gigabit Ethernet PCS (1000BASE-X)
operational_rate_i	Operational rate – When the core operates in SGMII PCS mode, this port controls the regulation rate of the rate adaptation circuit blocks as follows: 10 = 1 Gbps rate 01 = 100 Mbps rate 00 = 10 Mbps rate Note in Gigabit Ethernet PCS mode, the rate adaptation blocks always operate at 1 Gbps rate regardless of the settings on the operational_rate_i control pins.
debug_link_timer_short_i	Debug Link Timer Mode – Active high signal that forces the auto-negotiation link timer to run much faster than normal. This mode is provided for debugging purposes (for example, allowing simulations to run through the auto-negotiation process much faster than normal). This signal must not change during the auto-negotiation process.
pll_lock_i	PLL Lock – External PLL lock signal. Note that this signal is only available if the *Use External PLL* option is enabled in the GUI.

4.2.1. SGMII IP Wizard Configuration

In the reference design, the SGMII IP is configured as shown in Figure 4.5.



Configure Component from IP sgmii Version 1.6.2
Set the following parameters to configure this component.

Configure IP

Property	Value
General	
(G)MII Style	TSMAC Easy Connect
RX CTC Mode	Dynamic
Static Low FIFO Threshold [1 - 1016]	16
Static High FIFO Threshold [4 - 1020]	32
Optional Ports	
Use External PLL (remove internal PLL instance)	☑
Enable Port: CDR Reference clock (input)	☑
SGMII Core Register Access	LMMI

No DRC issues are found.

Figure 4.5. SGMII IP Wizard Configuration

The following lists the configuration settings for SGMII:

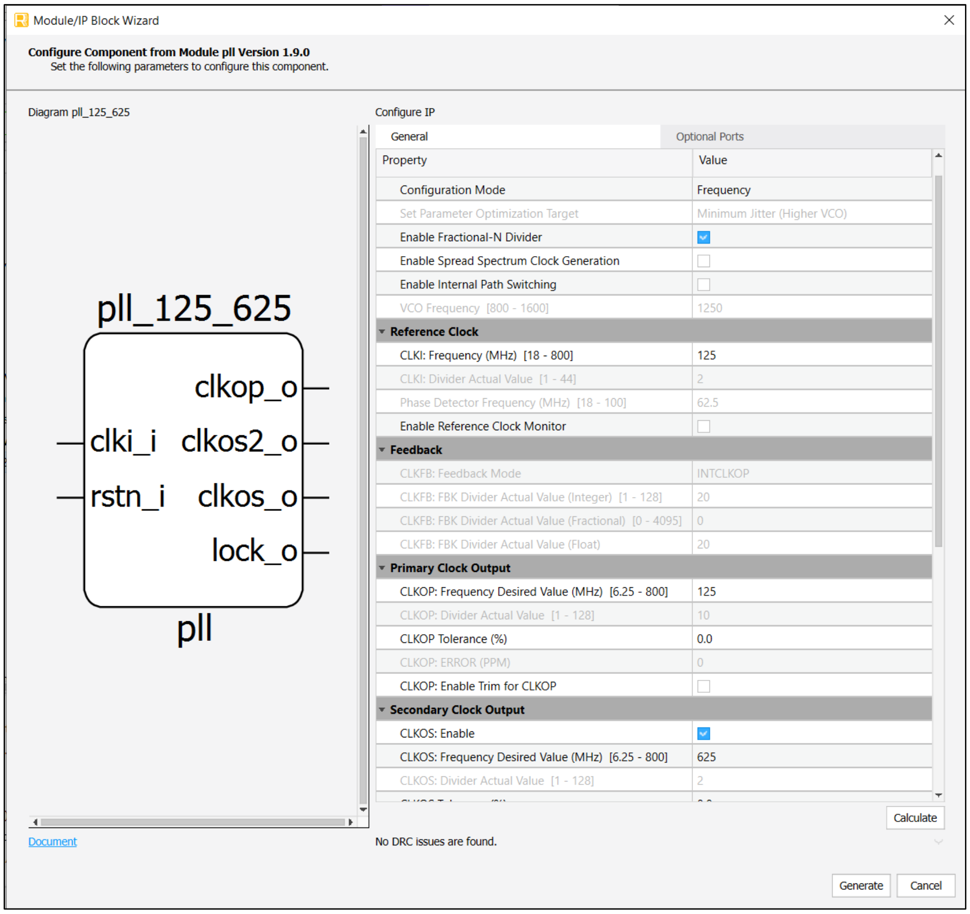
- GMII Style: TSEMAC Easy Connect
- RX CTC Mode: Dynamic
- Use External PLL: Yes (in case of any additional SGMII)
- Enable Port: CDR Reference clock: Yes
- SGMII Core Register Access: LMMI

4.3. PLL IP

In the reference design, an external PLL is used to provide clock source of 125 MHz and 625 MHz.

The following lists the configuration settings for the PLL IP:

- The Fractional-N Divider: Enabled
- Clock Frequency: 125 MHz
- CLKOP: Frequency desired value: 125 MHz
- CLKOS: Enable
- CLKOS: Frequency desired value: 625 MHz
- CLKOS2: Enable
- CLKOS2: Frequency desired value: 625 MHz
- CLKOS2: Static Phase Shift: 90 degrees



Configure Component from Module pll Version 1.9.0
Set the following parameters to configure this component.

Property	Value
Configuration Mode	Frequency
Set Parameter Optimization Target	Minimum Jitter (Higher VCO)
Enable Fractional-N Divider	☑
Enable Spread Spectrum Clock Generation	☐
Enable Internal Path Switching	☐
VCO Frequency [800 - 1600]	1250
Reference Clock	
CLKI: Frequency (MHz) [18 - 800]	125
CLKI: Divider Actual Value [1 - 44]	2
Phase Detector Frequency (MHz) [18 - 100]	62.5
Enable Reference Clock Monitor	☐
Feedback	
CLKFB: Feedback Mode	INTCLKOP
CLKFB: FBK Divider Actual Value (Integer) [1 - 128]	20
CLKFB: FBK Divider Actual Value (Fractional) [0 - 4095]	0
CLKFB: FBK Divider Actual Value (Float)	20
Primary Clock Output	
CLKOP: Frequency Desired Value (MHz) [6.25 - 800]	125
CLKOP: Divider Actual Value [1 - 128]	10
CLKOP Tolerance (%)	0.0
CLKOP: ERROR (PPM)	0
CLKOP: Enable Trim for CLKOP	☐
Secondary Clock Output	
CLKOS: Enable	☑
CLKOS: Frequency Desired Value (MHz) [6.25 - 800]	625
CLKOS: Divider Actual Value [1 - 128]	2

No DRC issues are found.

Figure 4.6. PLL IP Wizard Configuration

4.4. Pattern Generator and Checker

The pattern generator and checker module are used to generate the Ethernet packets and transmit it to the TSEMAC IP through the AXI-4 interface. You can configure the required pattern to generate and transmit to the TSEMAC.

There are two data types that you can select: counter data, or Ethernet format pattern data. In counter data pattern, the complete data is considered as incremental counter data. There will be no UDP or Internet Protocol information. In Ethernet pattern data type, the complete Ethernet format is considered and generated from the generator.

The checker compares the data that is received from the TSEMAC IP. The complete header part is compared directly with the constant value that is provided to the generator. After comparing all the constant values, it will run its own counter pattern and it is compared directly with the received signal. In case of any byte mismatch that occurs then the compare fail signal will assert continuously until you press the restart button or do a power cycle.

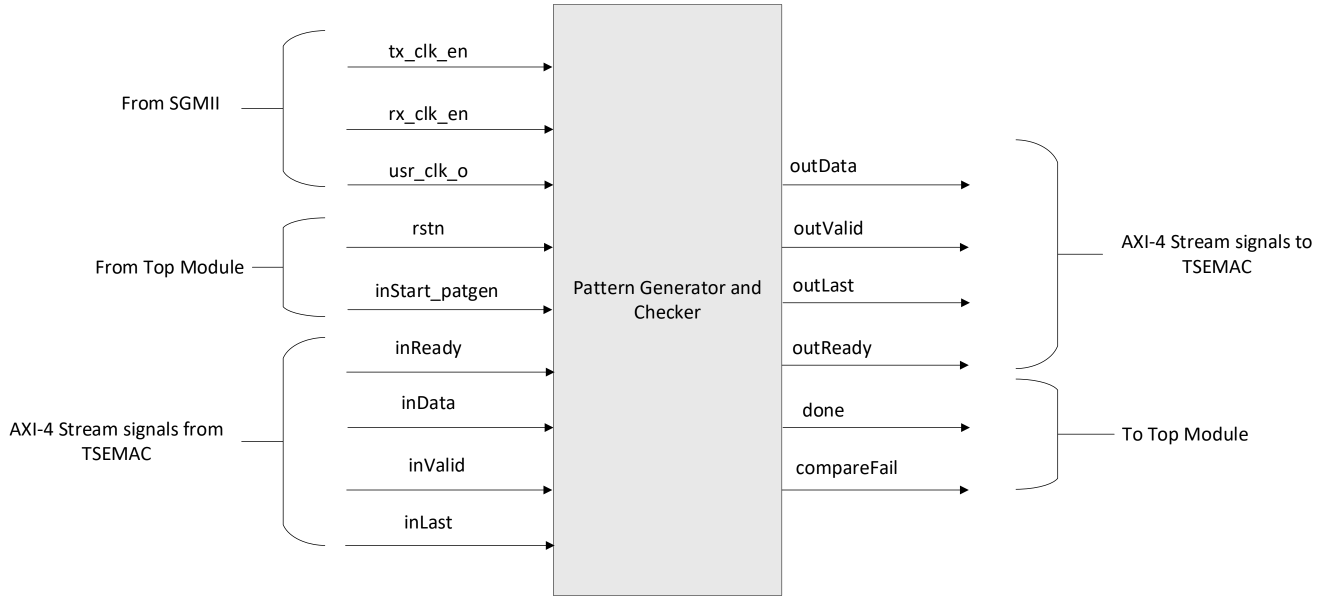


Figure 4.7. Pattern_gen_checker Module Signals

Table 4.3. Pattern Generator and Checker Module Signal Description

Signal	I/O Type	I/O Width	Description
tx_clk_en	I	1	This clock enable is provided by the SGMII. According to the operating speed the clock enable is generated from the SGMII, which is provided as clock to tx_mac_clk of the TSEMAC. Hence, the pattern generator also needs to be generated with the same clock in the reference design that you provide to the pattern generator. This is provided for the pattern generator section only.
rx_clk_en	I	1	This clock enable is provided by the SGMII. According to the operating speed the clock enable will be generated from the SGMII which is provided as clock to rx_mac_clk of the TSEMAC. Hence the pattern generator also needs to be generated with the same clock in the reference design that you provide to the pattern checker. This is provided for the pattern checker section only.
clk	I	1	The main clock for this module is system clock. For example, 125 MHz.
rstn	I	1	Active low reset signal.
inStart_patgen	I	1	This inStart_patgen will start pattern generation. This signal depends on three signals—PLL_lock, Auto_link_ok, and packet_gen_start.
InReady	I	1	The inReady signal informs the pattern generator that the TSEMAC is ready to accept the data.
inData	I	8	inData is the data transmitted from the TSEMAC RX FIFO. This is provided to the pattern checker. This data will only be valid when inValid signal is high.

Signal	I/O Type	I/O Width	Description
inValid	I	1	This signal indicates whether the data transmitted from the TSEMAC is Valid. Only when this signal is high, the data from the TSEMAC is considered and compared.
inLast	I	1	This signal indicates last frame/byte of the packet.
OutReady	O	1	This is always kept high to indicate that the application layer is always ready to accept frames from the TSEMAC.
OutData	O	8	The generated data from the generator is provided to the TSEMAC through this signal.
OutValid	O	1	This is set to '1' only when you are sending the valid data.
OutLast	O	1	On the last frame/byte, this signal is set to '1', which informs the TSEMAC that the last data is being transmitted.
Done	O	1	This signal asserts only when all the defined NUM_PKT is served.
compareFail	O	1	This signal will assert when any byte/frame mismatch is found in valid transactions.

4.4.1. Data Format

The following figure shows the IEEE 802.3 frame.

DESTINATION ADDRESS 6 bytes	SOURCE ADDRESS 6 bytes	LENGTH/ TYPE 2 bytes	DATA/PAD 46-1,500 bytes

Figure 4.8. Frame Format Considered in Pattern Generator in Counter Data

The destination address and source address are fixed in the pattern generator. You can change the destination address accordingly while interfacing with the PC or laptop. The following describes the data format in the pattern generator:

- Destination address: The destination address is of 6-bytes. This is the MAC address of the destination device.
- Source address: The Source address is of 6-bytes. This is the MAC address of the source device.
- Length/type: Length/type describes the type of packet. For IPv4 - 0x0800, IPv6 - 0x08DD and less than 05DC bytes is considered as Ethernet length. There are different types of length indicator values for different protocols.
- Data/pad: Data/ad consists of the data that is sent to the destination device. In the counter data type, the incremental data is considered.

The Ethernet format contains many layers of information. The pattern generator generates these layer data type patterns. The IP and UDP MAC header are inside the data field. The following list describes the Ethernet frame format with IP and UDP header:

- Preamble and SFD: This information is added by the TSEMAC.
- MAC Header: This header contains destination MAC address, source MAC address, and length/type.
- IP Header: This header contains many fields with information of IPv4, as shown in Figure 4.9.
- UDP Header: This header contains information on the UDP destination and source address and other information. The field of this header is shown in Figure 4.10.
- Data: This is the incremental counter data.
- FCS: This FCS is added by the TSEMAC IP. By default, the Transmit MAC is configured to generate the FCS pattern for the frame to be transmitted. The register 0x02 bit [2] determines whether the FCS is required or prevented. A CRC is used by transmit and receive algorithms to generate a CRC value for the FCS field. The FCS field contains a 4-octet (32-bit) CRC value. This value is computed as a function of the contents of the source address, destination address, type/length, and data.

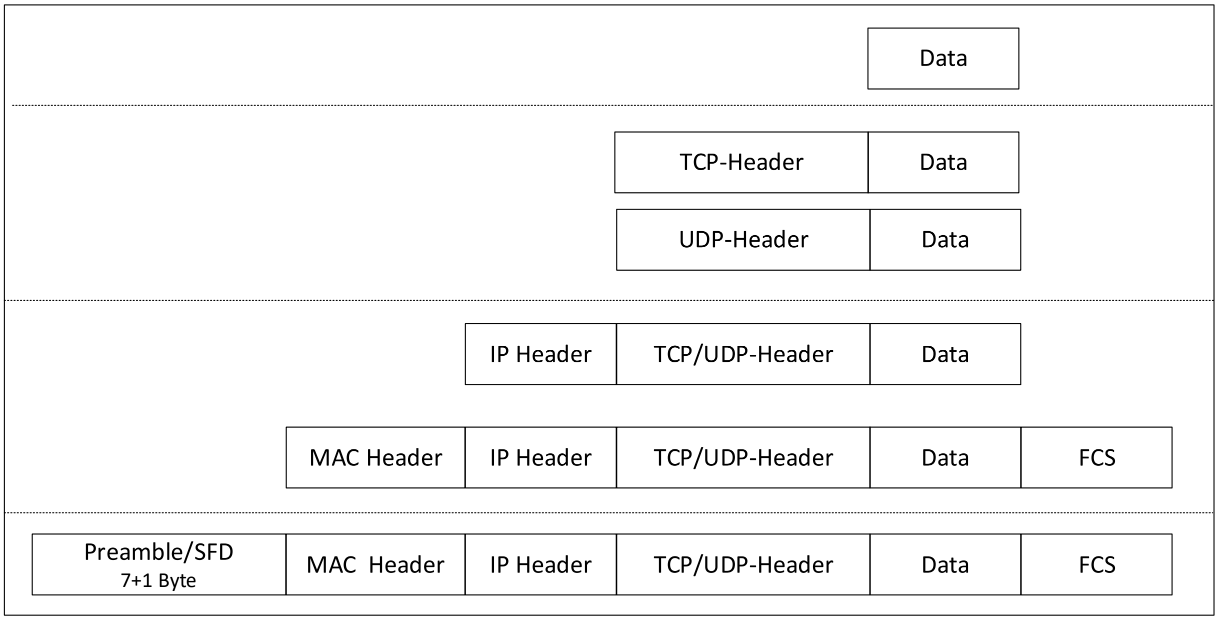


Figure 4.9. Ethernet Frame Format with IP and UDP Header

The following figure shows the IP header and UDP header format.

IP Bit Offset	0-3	4-7	8-13	14-15	16-18	19-31
0	Version	Header Length	DSCP	ECN	Total Length	
32	Identification				Flags	Fragments Offset
64	Time to Live		Protocol		Header Checksum	
96	Source Address					
128	Destination Address					
UDP Bit Offset	0-15				16-31	
160	Source Port Number				Destination Port Number	
192	Length				Checksum	
224	Payload					

Figure 4.10. IP and UDP Header Packet Format

4.5. APB Module

This module is used to configure the registers of the TSEMAC IP. To enable certain functions, it is necessary to write some register of the TSEMAC. The registers are configured with the data provided in Table 4.4.

Table 4.4. APB Module Register Configuration Table

Registers Address	Data	Description
8'h04	8'h01	Bit- [0] indicates Promiscuous Mode - When set to 1, all filtering schemes are abandoned, and the RX MAC receives frames with any address. Bit- [1] indicates Discard FCS - RX Discard FCS and Pad. When set, the FCS and any padding bytes of an IEEE 802.3 frame are stripped off before it is transferred to the RX FIFO. When low, the entire frame is transferred into the RX FIFO. Note: Discarding padding bytes is only applicable to pure IEEE 802.3 frames (such as in backplane applications) and does not function on Ethernet frames (IP, UDP, ICMP, and so on.) where the length field is now interpreted as a protocol type field.
8'h00	8'h0C	Mode Register - Transmit and Receive Control Register.

4.6. Testbench Module

This is the simulation testbench top module. In this module, the pattern generator activates only when the pll_lock_out signal asserts. Upon pattern generation completion, the patgen_done_out signal is asserted. When any byte is missing on the receiving side, then the compare fail signal is asserted. 125 MHz clock is generated in the testbench and provided to the top module.

The following table lists the settings that you can configure in the testbench top module.

Table 4.5. Testbench Top Module Configuration Parameter

Configurations	Description
CONTINUOUS_TRAFFIC	1 - For continued traffic generation, 0 - The traffic will generate for assigned NUM_PKT.
OPERATION_SPEED	0 – 10 Mbps, 1 – 100 Mbps, 2 – 1 Gbps.
NUM_PKT	Number of packets to be generated. Applicable when CONTINUOUS_TRAFFIC = 0.
SIM	0 - Hardware Run, 1 - Simulation Run. This parameter must be set to 1 for simulation.

The following figure shows the block diagram of a testbench top module.

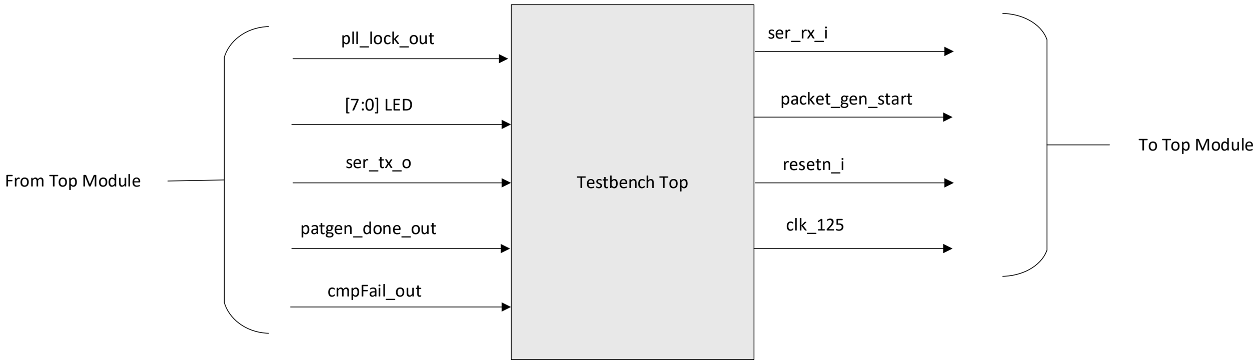
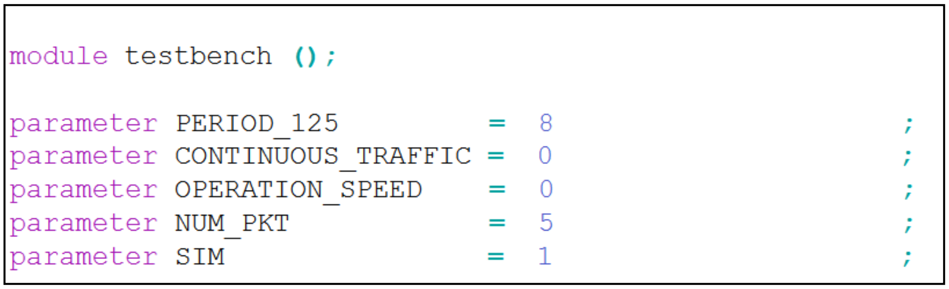


Figure 4.11. Testbench Top Module

For signal description, refer to Table 4.1.

The following figure shows the parameters for the testbench module in code base.



```
module testbench ();

parameter PERIOD_125          =  8              ;
parameter CONTINUOUS_TRAFFIC  =  0              ;
parameter OPERATION_SPEED     =  0              ;
parameter NUM_PKT             =  5              ;
parameter SIM                 =  1              ;
```

Figure 4.12. Testbench Top Parameters

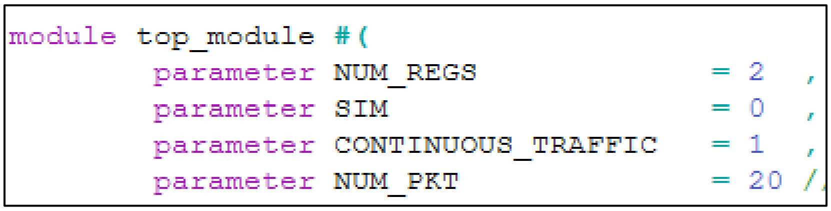
4.7. Top Module

This is the top-level module of the reference design that instantiates a PLL, APB module, TSEMAC, SGMII, and pattern generator/checker. The top module architecture is shown in Figure 4.2. The port description is shown in Table 4.1. In the top module, you can configure the required parameters before generating the bit file, as shown in Figure 4.13. The following table lists the parameter description of the top module.

Table 4.6. Top Module Parameter Description

Parameters	Description
NUM_REGS	Number of registers that are needed to be configured through the APB interface.
SIM	0 - Hardware Run, 1 - Simulation Run.
CONTINUOUS_TRAFFIC	1 - For continues traffic generation 0 - The traffic will generate for the assigned number of packets (NUM_PKT).
NUM_PKT	Number of packets to be generated. Considered only when CONTINUOUS_TRAFFIC = 0.

The following figure shows the parameters of the top module.



```
module top_module #(
        parameter NUM_REGS              = 2  ,
        parameter SIM                   = 0  ,
        parameter CONTINUOUS_TRAFFIC    = 1  ,
        parameter NUM_PKT               = 20 /
```

Figure 4.13. Top Module Parameters

5. Simulation Waveforms

The following figures show the timing waveforms of the reference design.

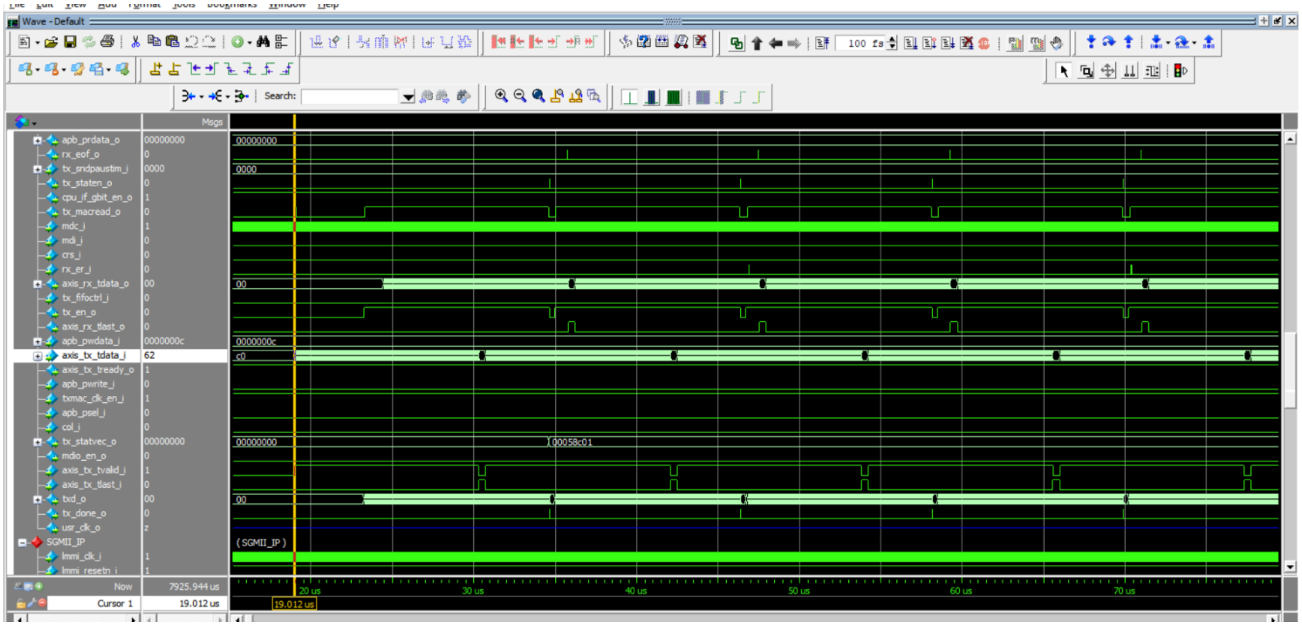


Figure 5.1. Timing Waveforms for the TSEMAC

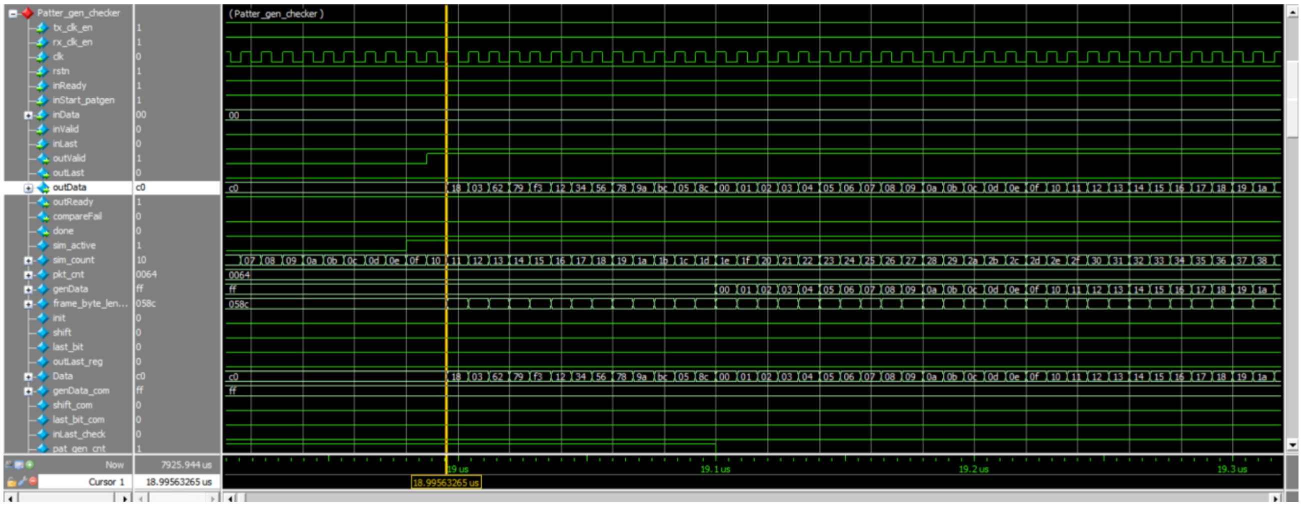


Figure 5.2. Timing Waveforms for the Pattern Generator

6. Implementation

This design is implemented in Verilog. When using this design in a different device, density, or speed grade, the performance and utilization can vary. Default settings are used during the fitting of the design.

Table 6.1. Performance and Resource Utilization

Device Family	Language	Speed Grade	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Architecture Resources
CertusPro™-NX*	Verilog	-9	3492	>125	17	1 PLL, 1 ECLKDIV
CrossLink™-NX	Verilog	-8	3490	>125	15	1 PLL, 1 ECLKDIV

*** Note:** Performance and utilization characteristics are generated using LFCPNX-100-9LFG672C and LIFCL-40-8BG400C with the Lattice Radiant™ software version 2023.2 and Lattice Synthesis Engine (LSE).

Note: The maximum clock frequency is obtained by running the timing analysis with the Lattice design software. Timing simulation must run after any changes are made and the reference design is merged with the overall design.

7. Reference Design Simulation

The following figure shows the simulation design with the top_module. The Testbench_top inputs are provided to the top_module. The Rx_i and Tx_o serial signals are looped in Testbench_top. The clock is also provided to the top module from the testbench.

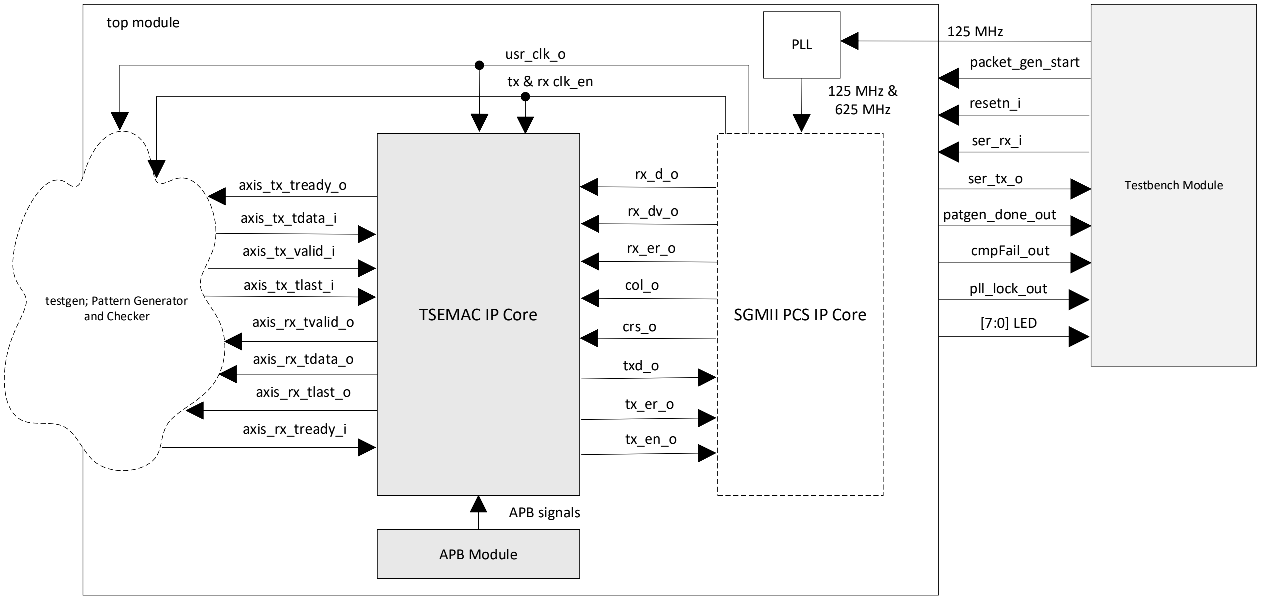


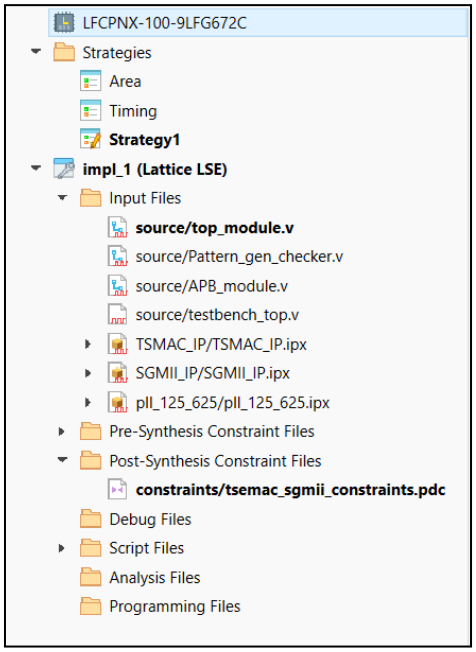
Figure 7.1. Reference Design Simulation Architecture

7.1. Running the Design Simulation

To run the design simulation, follow these steps:

1. Unzip the reference design zip file into a directory.
2. Open the reference design project file (*tsemac_sgmii.rdf*) from the Lattice Radiant software.

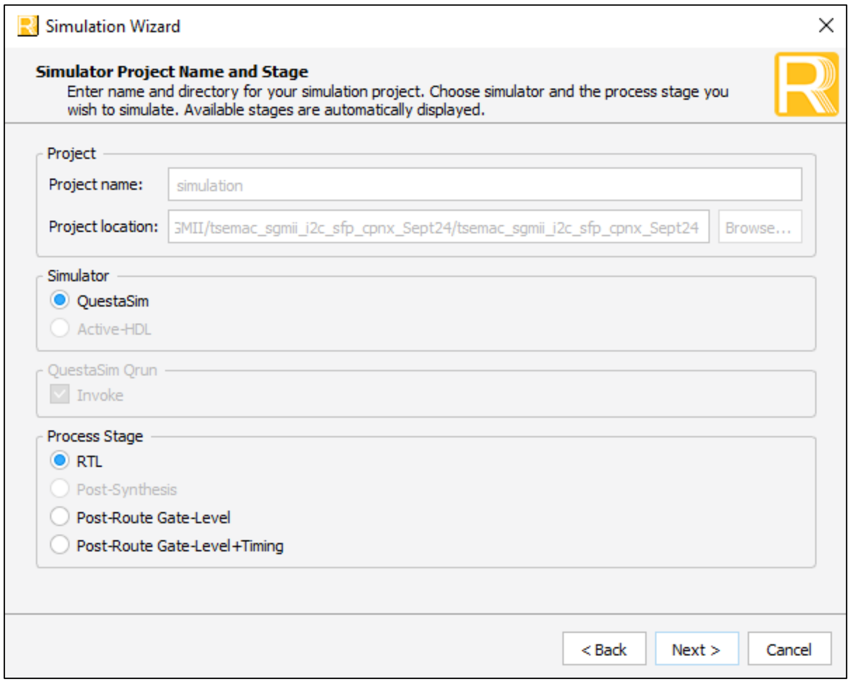
3. The reference design file list appears.



- LFCPNX-100-9LFG672C
- Strategies
  - Area
  - Timing
  - **Strategy1**
- **impl_1 (Lattice LSE)**
  - Input Files
    - **source/top_module.v**
    - source/Pattern_gen_checker.v
    - source/APB_module.v
    - source/testbench_top.v
    - TSMAC_IP/TSMAC_IP.ipx
    - SGMII_IP/SGMII_IP.ipx
    - pll_125_625/pll_125_625.ipx
  - Pre-Synthesis Constraint Files
  - Post-Synthesis Constraint Files
    - **constraints/tsemac_sgmii_constraints.pdc**
  - Debug Files
  - Script Files
  - Analysis Files
  - Programming Files

Figure 7.2. Reference Design File List

4. Click on **Tools > Simulation Wizard**.
5. Create a new *.spf* file and click **Next**.



Simulation Wizard

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: simulation
Project location: 3MII/tsemac_sgmii_i2c_sfp_cpnx_Sept24/tsemac_sgmii_i2c_sfp_cpnx_Sept24 Browse...

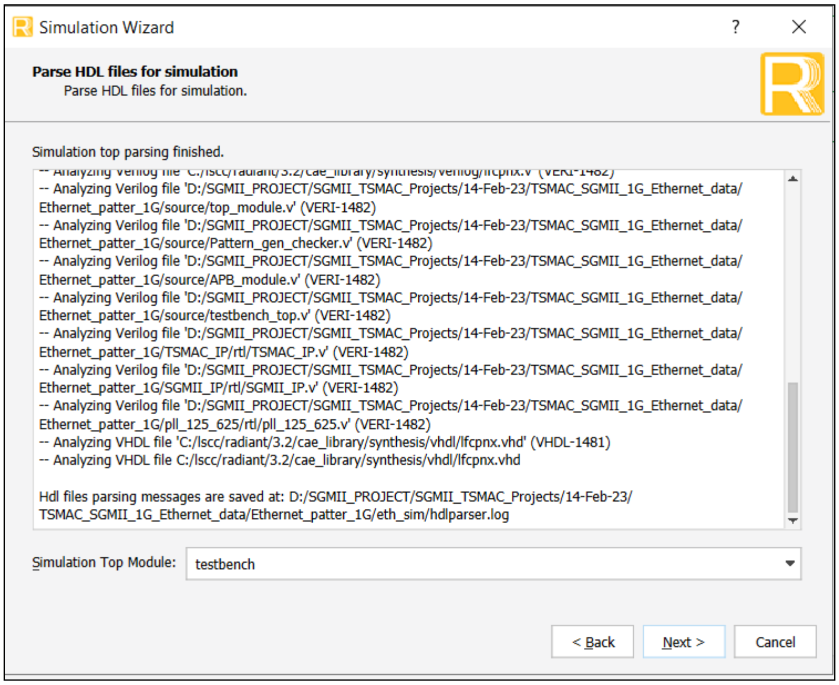
Simulator
● QuestaSim
○ Active-HDL

QuestaSim Qrun
☑ Invoke

Process Stage
● RTL
○ Post-Synthesis
○ Post-Route Gate-Level
○ Post-Route Gate-Level+Timing

Figure 7.3. Simulation Wizard: Create Simulation Project

6. Set the simulation top module as the testbench and click **Next**.



Simulation Wizard
?
Parse HDL files for simulation
Parse HDL files for simulation.
< Back
Next >
Cancel

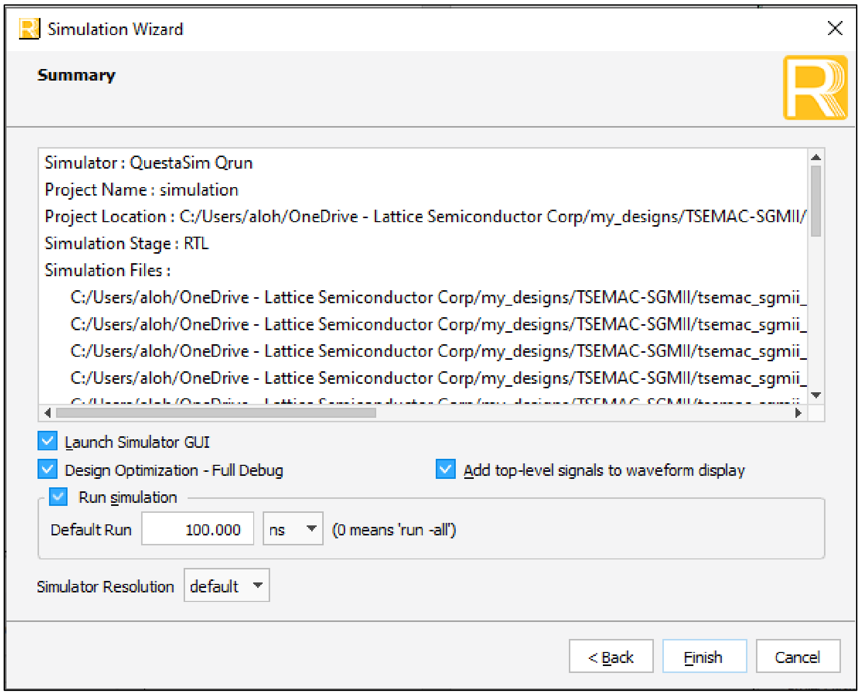
Simulation top parsing finished.

```
-- Analyzing Verilog file 'C:/lscc/radiant/3.2/cae_library/synthesis/verilog/lfcpnx.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/source/top_module.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/source/Pattern_gen_checker.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/source/APB_module.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/source/testbench_top.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/TSMAC_IP/rtl/TSMAC_IP.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/SGMII_IP/rtl/SGMII_IP.v' (VERI-1482)
-- Analyzing Verilog file 'D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/TSMAC_SGMII_1G_Ethernet_data/
Ethernet_patter_1G/pll_125_625/rtl/pll_125_625.v' (VERI-1482)
-- Analyzing VHDL file 'C:/lscc/radiant/3.2/cae_library/synthesis/vhdl/lfcpnx.vhd' (VHDL-1481)
-- Analyzing VHDL file C:/lscc/radiant/3.2/cae_library/synthesis/vhdl/lfcpnx.vhd

Hdl files parsing messages are saved at: D:/SGMII_PROJECT/SGMII_TSMAC_Projects/14-Feb-23/
TSMAC_SGMII_1G_Ethernet_data/Ethernet_patter_1G/eth_sim/hdlparser.log
```

Simulation Top Module: testbench

Figure 7.4. Simulation Wizard: Select Simulation Top Module

7. Select the configuration as shown in the following figure and click **Finish**.



```
Simulator : QuestaSim Qrun
Project Name : simulation
Project Location : C:/Users/aloh/OneDrive - Lattice Semiconductor Corp/my_designs/TSEMAC-SGMII/
Simulation Stage : RTL
Simulation Files :
    C:/Users/aloh/OneDrive - Lattice Semiconductor Corp/my_designs/TSEMAC-SGMII/tsemac_sgmii_
    C:/Users/aloh/OneDrive - Lattice Semiconductor Corp/my_designs/TSEMAC-SGMII/tsemac_sgmii_
    C:/Users/aloh/OneDrive - Lattice Semiconductor Corp/my_designs/TSEMAC-SGMII/tsemac_sgmii_
    C:/Users/aloh/OneDrive - Lattice Semiconductor Corp/my_designs/TSEMAC-SGMII/tsemac_sgmii_
```

☑ 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 7.5. Simulation Wizard: Summary Page

8. Wait until the QuestaSim™ Lattice FPGA Edition software finishes loading.

9. From the **View** menu, select **Wave**.

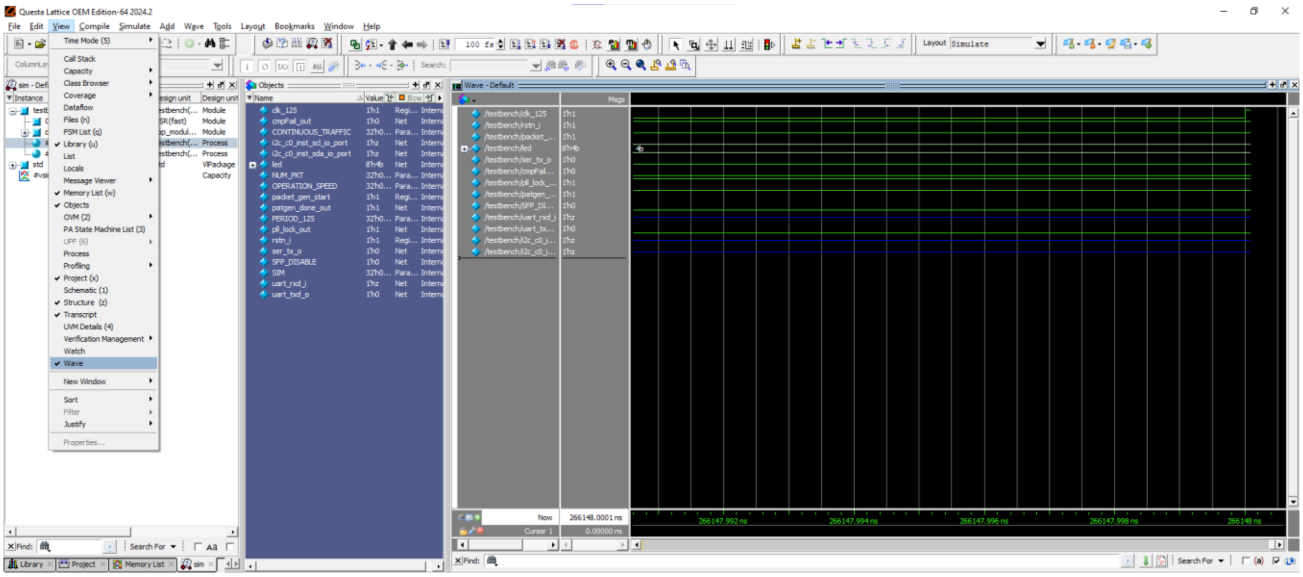


Figure 7.6. QuestaSim Lattice FPGA Software Interface

10. To load the .do file, click on **Tools** > **TCL** > **Execute Macro**, and select the *wave.do* file from the zip file.

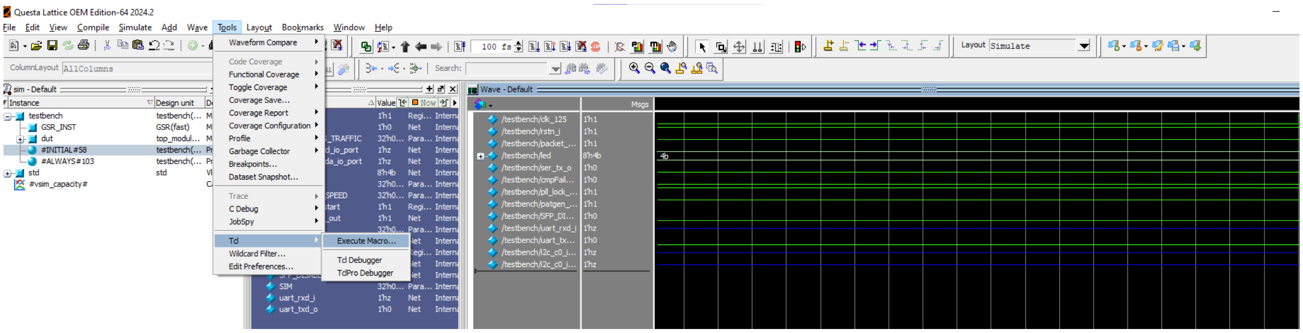
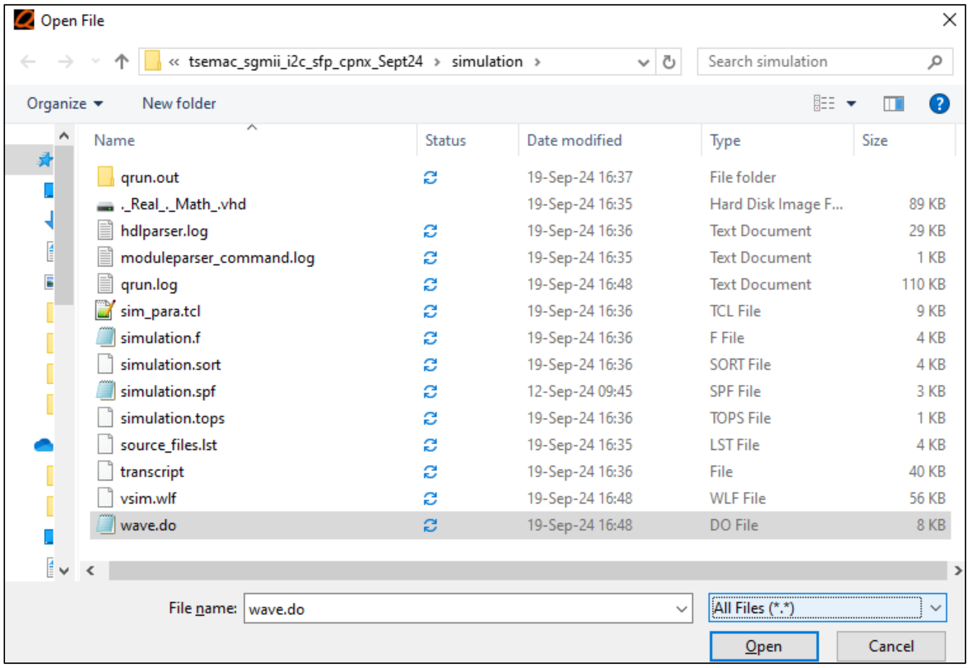


Figure 7.7. QuestaSim Lattice FPGA Edition Software: Execute wave.do File

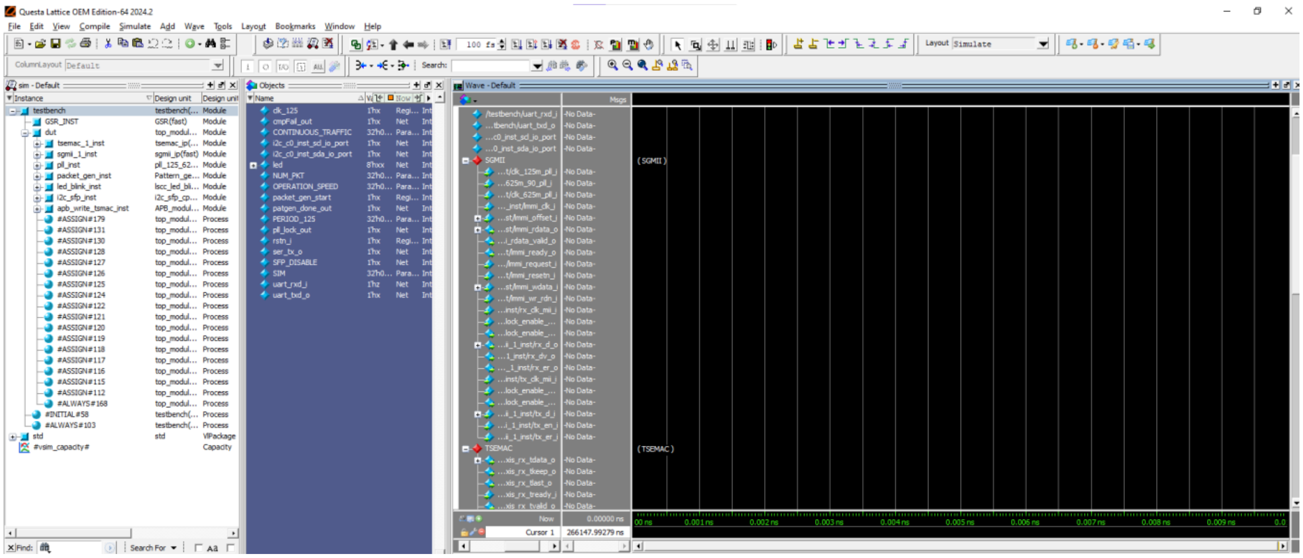


Name	Status	Date modified	Type	Size
qrun.out		19-Sep-24 16:37	File folder	
._Real_._Math_.vhd		19-Sep-24 16:35	Hard Disk Image F...	89 KB
hdlparser.log		19-Sep-24 16:36	Text Document	29 KB
moduleparser_command.log		19-Sep-24 16:35	Text Document	1 KB
qrun.log		19-Sep-24 16:48	Text Document	110 KB
sim_para.tcl		19-Sep-24 16:36	TCL File	9 KB
simulation.f		19-Sep-24 16:36	F File	4 KB
simulation.sort		19-Sep-24 16:36	SORT File	4 KB
simulation.spf		12-Sep-24 09:45	SPF File	3 KB
simulation.tops		19-Sep-24 16:36	TOPS File	1 KB
source_files.lst		19-Sep-24 16:35	LST File	4 KB
transcript		19-Sep-24 16:36	File	40 KB
vsim.wlf		19-Sep-24 16:48	WLF File	56 KB
wave.do		19-Sep-24 16:48	DO File	8 KB

File name: wave.do
All Files (*.*)

Figure 7.8. QuestaSim Lattice FPGA Edition Software: Location of the wave.do File

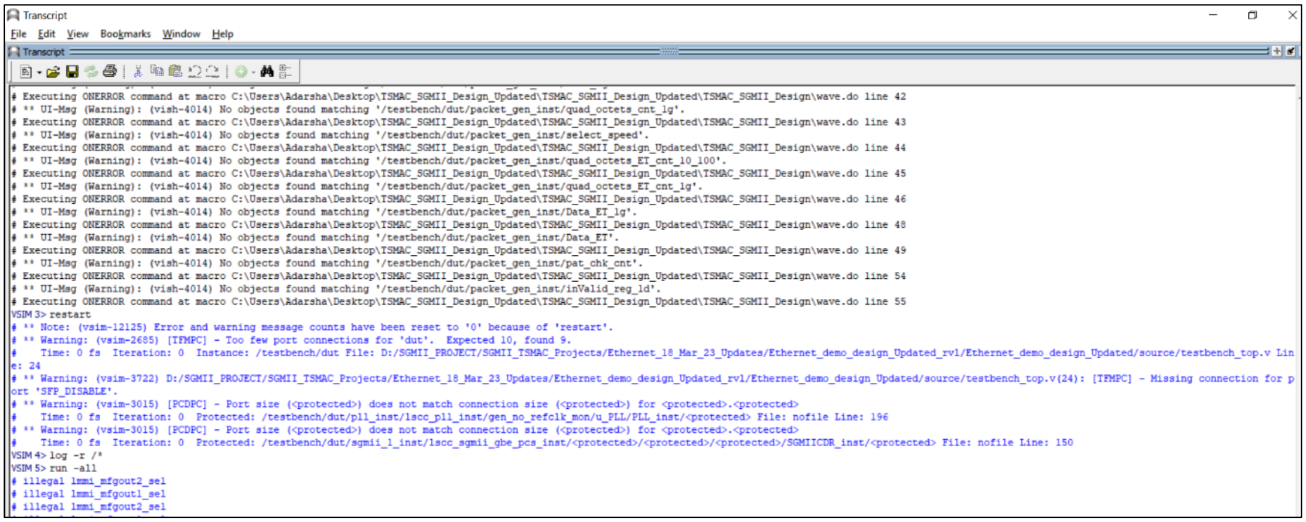
11. The following figure shows that the signals that are added are displayed in the wave window.



Questa Lattice OEM Edition-64 2024.2
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
sim - Default
Objects
Wave - Default
Msgs
(SGMII)
(TSEMAC)
Now
Cursor 1

Figure 7.9. QuestaSim Lattice FPGA Edition Software: Signals Added

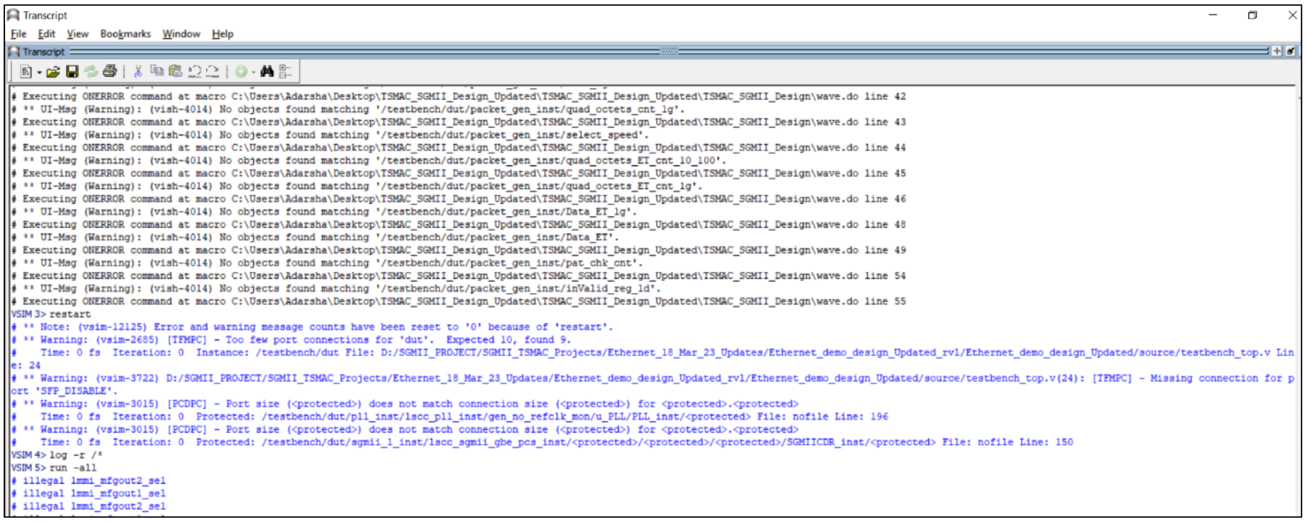
12. To run the simulation, type “**restart,**” “**log -r /***” and “**run -all**” on the Transcript window.



```
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 42
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_cnt_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 43
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/select_speed'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 44
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_ET_cnt_10_100'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 45
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_ET_cnt_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 46
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/Data_ET_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 48
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/Data_ET'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 49
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/pat_chk_cnt'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 54
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/inValid_reg_ld'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 55
VSIM 3> restart
# ** Note: (vsim-12125) Error and warning message counts have been reset to '0' because of 'restart'.
# ** Warning: (vsim-2685) [TFMPC] - Too few port connections for 'dut'.  Expected 10, found 9.
#    Time: 0 fs  Iteration: 0  Instance: /testbench/dut File: D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_Updated_rvl/Ethernet_demo_design_Updated/source/testbench_top.v Lin
e: 24
# ** Warning: (vsim-3722) D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_Updated_rvl/Ethernet_demo_design_Updated/source/testbench_top.v(24): [TFMPC] - Missing connection for p
ort 'SFP_DISABLE'.
# ** Warning: (vsim-3015) [PCDPC] - Port size (<protected>) does not match connection size (<protected>) for <protected>.<protected>
#    Time: 0 fs  Iteration: 0  Protected: /testbench/dut/pll_inst/lscc_pll_inst/gen_no_refclk_mon/u_PLL/PLL_inst/<protected> File: nofile Line: 196
# ** Warning: (vsim-3015) [PCDPC] - Port size (<protected>) does not match connection size (<protected>) for <protected>.<protected>
#    Time: 0 fs  Iteration: 0  Protected: /testbench/dut/sgmii_1_inst/lscc_sgmii_gbe_pcs_inst/<protected>/<protected>/<protected>/SGMIICDR_inst/<protected> File: nofile Line: 150
VSIM 4> log -r /*
VSIM 5> run -all
# illegal lmmi_mfgout2_sel
# illegal lmmi_mfgout1_sel
# illegal lmmi_mfgout2_sel
```

Figure 7.10. QuestaSim Lattice FPGA Edition Software: Restart Simulation to Capture the Signals in the Initial Timeframe

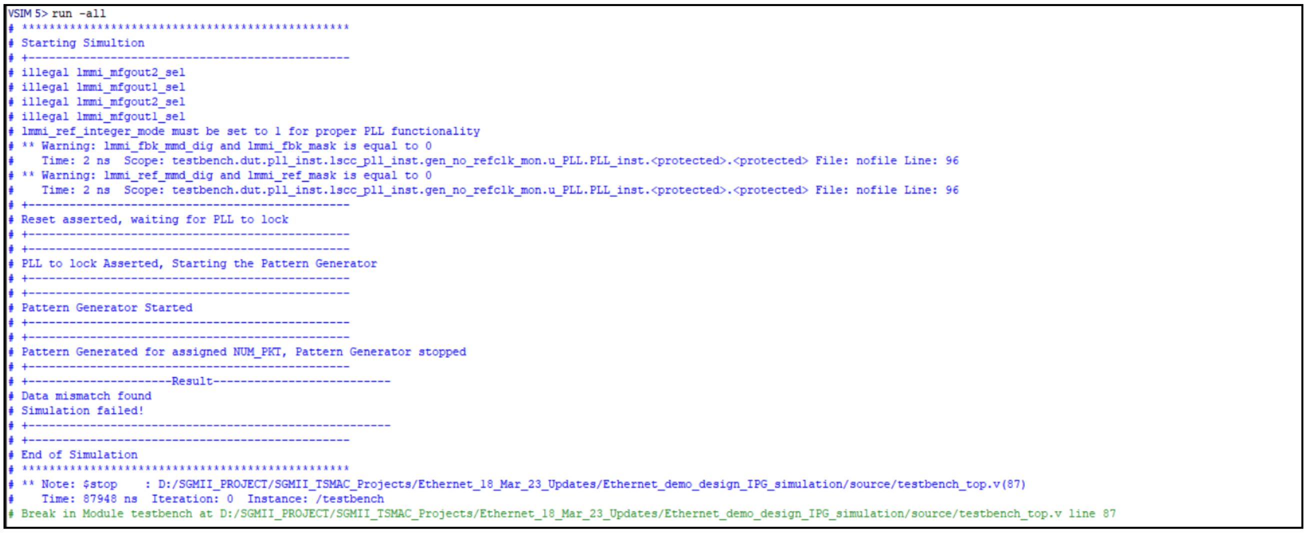
13. Wait for the simulation to complete to see the waveforms. If the continuous traffic is disabled, the message shown in the following figure displays. The following figure shows the passing condition.



```
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 42
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_cnt_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 43
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/select_speed'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 44
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_ET_cnt_10_100'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 45
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/quad_octets_ET_cnt_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 46
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/Data_ET_lg'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 48
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/Data_ET'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 49
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/pat_chk_cnt'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 54
# ** UI-Msg (Warning): (vish-4014) No objects found matching '/testbench/dut/packet_gen_inst/inValid_reg_ld'.
# Executing ONERROR command at macro C:\Users\Adarsha\Desktop\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design_Updated\TSMAC_SGMII_Design\wave.do line 55
VSIM 3> restart
# ** Note: (vsim-12125) Error and warning message counts have been reset to '0' because of 'restart'.
# ** Warning: (vsim-2685) [TFMPC] - Too few port connections for 'dut'.  Expected 10, found 9.
#    Time: 0 fs  Iteration: 0  Instance: /testbench/dut File: D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_Updated_rvl/Ethernet_demo_design_Updated/source/testbench_top.v Lin
e: 24
# ** Warning: (vsim-3722) D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_Updated_rvl/Ethernet_demo_design_Updated/source/testbench_top.v(24): [TFMPC] - Missing connection for p
ort 'SFP_DISABLE'.
# ** Warning: (vsim-3015) [PCDPC] - Port size (<protected>) does not match connection size (<protected>) for <protected>.<protected>
#    Time: 0 fs  Iteration: 0  Protected: /testbench/dut/pll_inst/lscc_pll_inst/gen_no_refclk_mon/u_PLL/PLL_inst/<protected> File: nofile Line: 196
# ** Warning: (vsim-3015) [PCDPC] - Port size (<protected>) does not match connection size (<protected>) for <protected>.<protected>
#    Time: 0 fs  Iteration: 0  Protected: /testbench/dut/sgmii_1_inst/lscc_sgmii_gbe_pcs_inst/<protected>/<protected>/<protected>/SGMIICDR_inst/<protected> File: nofile Line: 150
VSIM 4> log -r /*
VSIM 5> run -all
# illegal lmmi_mfgout2_sel
# illegal lmmi_mfgout1_sel
# illegal lmmi_mfgout2_sel
```

Figure 7.11. QuestaSim Lattice FPGA Edition Software: Passing Condition

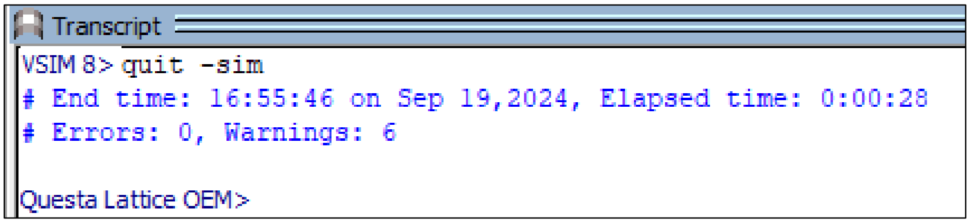
The following figure shows the failing condition.



```
VSIM 5> run -all
# ************************************************
# Starting Simultion
# +-----------------------------------------------
# illegal lmmi_mfgout2_sel
# illegal lmmi_mfgout1_sel
# illegal lmmi_mfgout2_sel
# illegal lmmi_mfgout1_sel
# lmmi_ref_integer_mode must be set to 1 for proper PLL functionality
# ** Warning: lmmi_fbk_mmd_dig and lmmi_fbk_mask is equal to 0
#    Time: 2 ns  Scope: testbench.dut.pll_inst.lscc_pll_inst.gen_no_refclk_mon.u_PLL.PLL_inst.<protected>.<protected> File: nofile Line: 96
# ** Warning: lmmi_ref_mmd_dig and lmmi_ref_mask is equal to 0
#    Time: 2 ns  Scope: testbench.dut.pll_inst.lscc_pll_inst.gen_no_refclk_mon.u_PLL.PLL_inst.<protected>.<protected> File: nofile Line: 96
# +-----------------------------------------------
# Reset asserted, waiting for PLL to lock
# +-----------------------------------------------
# +-----------------------------------------------
# PLL to lock Asserted, Starting the Pattern Generator
# +-----------------------------------------------
# +-----------------------------------------------
# Pattern Generator Started
# +-----------------------------------------------
# +-----------------------------------------------
# Pattern Generated for assigned NUM_PKT, Pattern Generator stopped
# +-----------------------------------------------
# +---------------------Result--------------------------
# Data mismatch found
# Simulation failed!
# +-----------------------------------------------------
# +-----------------------------------------------
# End of Simulation
# ************************************************
# ** Note: $stop    : D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_IPG_simulation/source/testbench_top.v(87)
#    Time: 87948 ns  Iteration: 0  Instance: /testbench
# Break in Module testbench at D:/SGMII_PROJECT/SGMII_TSMAC_Projects/Ethernet_18_Mar_23_Updates/Ethernet_demo_design_IPG_simulation/source/testbench_top.v line 87
```

Figure 7.12. QuestaSim Lattice FPGA Edition Software: Failing Condition

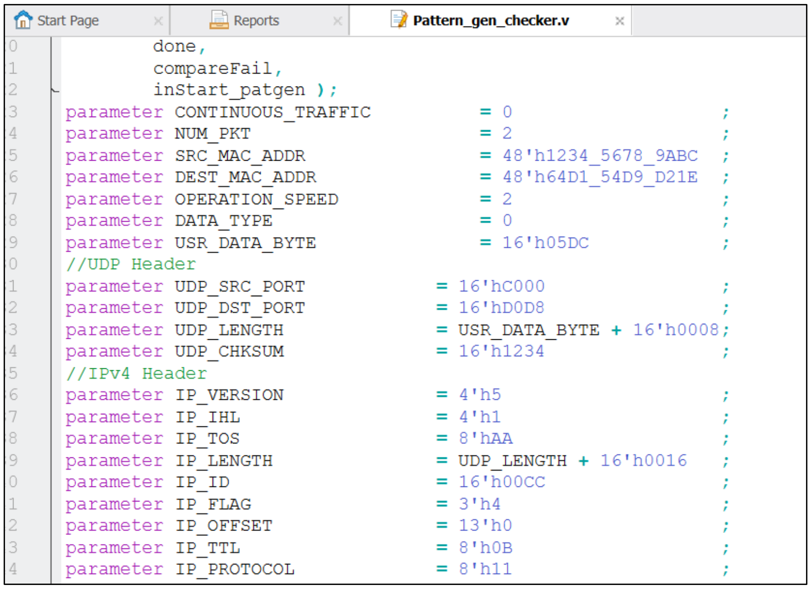
14. Type **quit -sim** in the QuestaSim Lattice FPGA Edition software console to terminate the simulation.



```
VSIM 8> quit -sim
# End time: 16:55:46 on Sep 19,2024, Elapsed time: 0:00:28
# Errors: 0, Warnings: 6

Questa Lattice OEM>
```

Figure 7.13. QuestaSim Lattice FPGA Edition Software: Quit the Simulation

15. You can modify the packet generator parameter in *Pattern_gen_checker.v* to generate packets of different lengths and specify the number of packets to be generated.



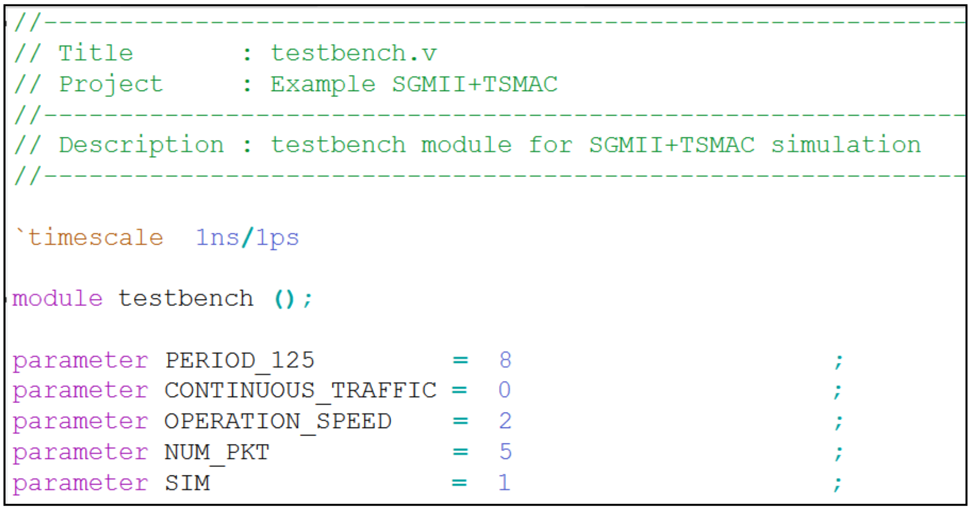
```
        done,
        compareFail,
        inStart_patgen );
parameter CONTINUOUS_TRAFFIC          = 0                 ;
parameter NUM_PKT                     = 2                 ;
parameter SRC_MAC_ADDR                = 48'h1234_5678_9ABC ;
parameter DEST_MAC_ADDR               = 48'h64D1_54D9_D21E ;
parameter OPERATION_SPEED             = 2                 ;
parameter DATA_TYPE                   = 0                 ;
parameter USR_DATA_BYTE               = 16'h05DC          ;
//UDP Header
parameter UDP_SRC_PORT            = 16'hC000              ;
parameter UDP_DST_PORT            = 16'hD0D8              ;
parameter UDP_LENGTH              = USR_DATA_BYTE + 16'h0008;
parameter UDP_CHKSUM              = 16'h1234              ;
//IPv4 Header
parameter IP_VERSION              = 4'h5                  ;
parameter IP_IHL                  = 4'h1                  ;
parameter IP_TOS                  = 8'hAA                 ;
parameter IP_LENGTH               = UDP_LENGTH + 16'h0016 ;
parameter IP_ID                   = 16'h00CC              ;
parameter IP_FLAG                 = 3'h4                  ;
parameter IP_OFFSET               = 13'h0                 ;
parameter IP_TTL                  = 8'h0B                 ;
parameter IP_PROTOCOL             = 8'h11                 ;
```

Figure 7.14. Packet Pattern Defined in the Pattern Generator

16. The reference design can be modified to support different operating speed and data types.

7.2. 1 Gbps Operation Speed Selection

1. In testbench file (*testbench_top.sv*), set the OPERATION_SPEED parameter to **2**. Check the LED description to identify the operating speed selected.



```
//-----------------------------------------------------------------
// Title        : testbench.v
// Project      : Example SGMII+TSMAC
//-----------------------------------------------------------------
// Description : testbench module for SGMII+TSMAC simulation
//-----------------------------------------------------------------

`timescale  1ns/1ps

module testbench ();

parameter PERIOD_125          =  8                        ;
parameter CONTINUOUS_TRAFFIC =  0                        ;
parameter OPERATION_SPEED     =  2                        ;
parameter NUM_PKT             =  5                        ;
parameter SIM                 =  1                        ;
```

Figure 7.15. OPERATION_SPEED=2 for 1 Gbps Testing

2. The following figure shows a simulation screenshot of LED indication. This is based on a failed condition in the User Datagram Protocol (UDP). The source port: 49152 identifies the process that sent the data and destination port: 53464 identifies the process that is to receive the data. The LED status is shown in the following figure.

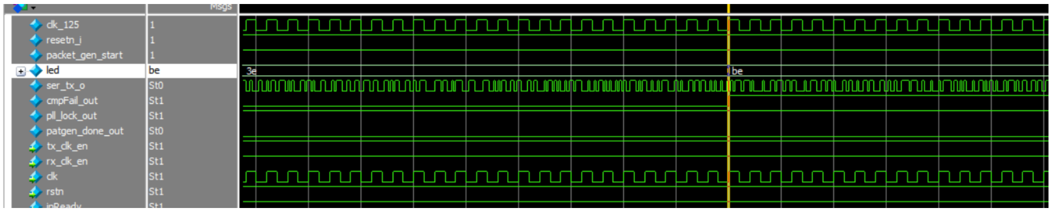
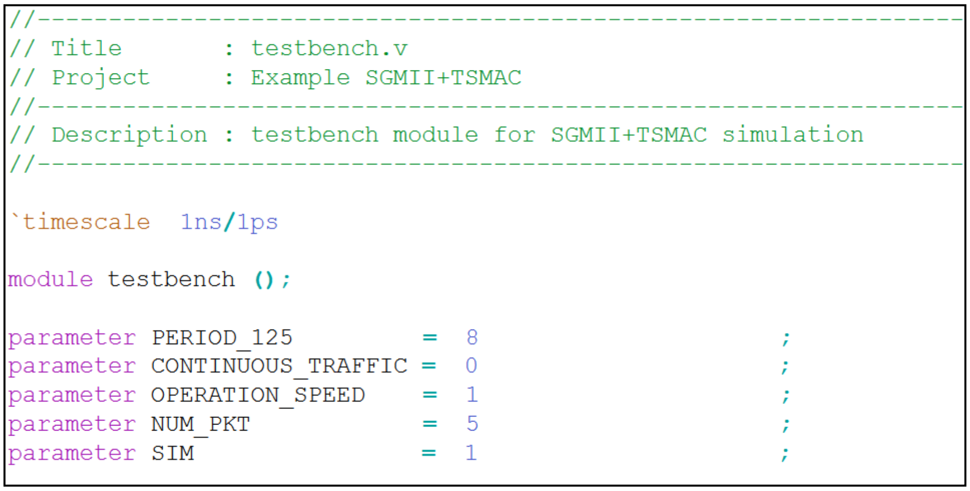


Figure 7.16. LED Status Signal in Simulation

7.3. 100 Mbps Operation Speed Selection

In the testbench file (*testbench_top.sv*), set the OPERATION_SPEED parameter to **1**. Check the LED description to identify the operating speed selected.

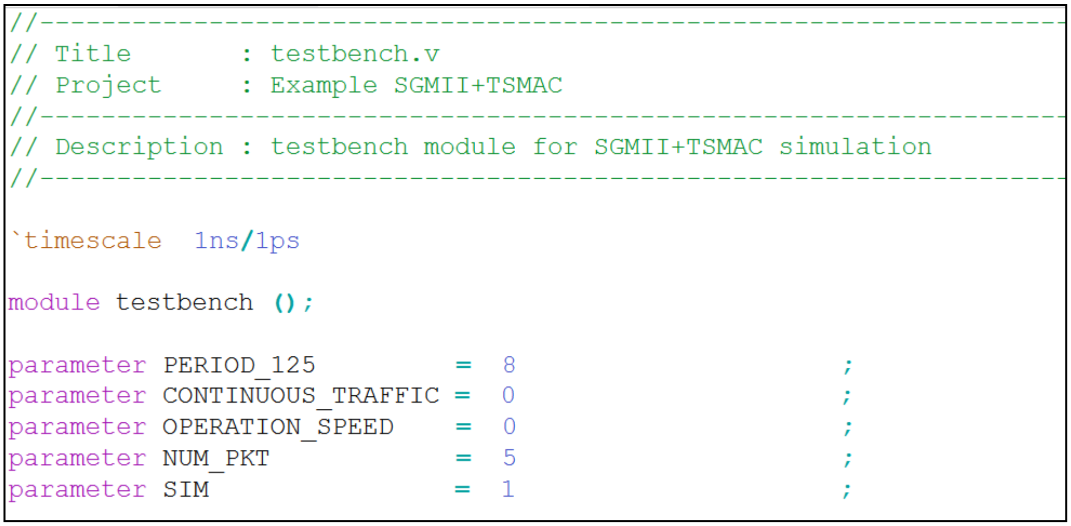


```
//--------------------------------------------------------------------
// Title        : testbench.v
// Project      : Example SGMII+TSMAC
//--------------------------------------------------------------------
// Description : testbench module for SGMII+TSMAC simulation
//--------------------------------------------------------------------

`timescale  1ns/1ps

module testbench ();

parameter PERIOD_125         =  8                      ;
parameter CONTINUOUS_TRAFFIC =  0                      ;
parameter OPERATION_SPEED    =  1                      ;
parameter NUM_PKT            =  5                      ;
parameter SIM                =  1                      ;
```

Figure 7.17. OPERATION_SPEED=1 for 100 Mbps Testing

7.4. 10 Mbps Operation Speed Selection

In the testbench file (*testbench_top.sv*), set the OPERATION_SPEED parameter to **0**. Check the LED description to identify the operating speed selected.



```
//--------------------------------------------------------------------
// Title        : testbench.v
// Project      : Example SGMII+TSMAC
//--------------------------------------------------------------------
// Description : testbench module for SGMII+TSMAC simulation
//--------------------------------------------------------------------

`timescale  1ns/1ps

module testbench ();

parameter PERIOD_125         =  8                      ;
parameter CONTINUOUS_TRAFFIC =  0                      ;
parameter OPERATION_SPEED    =  0                      ;
parameter NUM_PKT            =  5                      ;
parameter SIM                =  1                      ;
```

Figure 7.18. OPERATION_SPEED=0 for 10 Mbps Testing

8. Reference Design Implementation on Board

The following figure shows the interface between each module for Board Bring Up. On the board, the TX and RX ports are looped back externally.

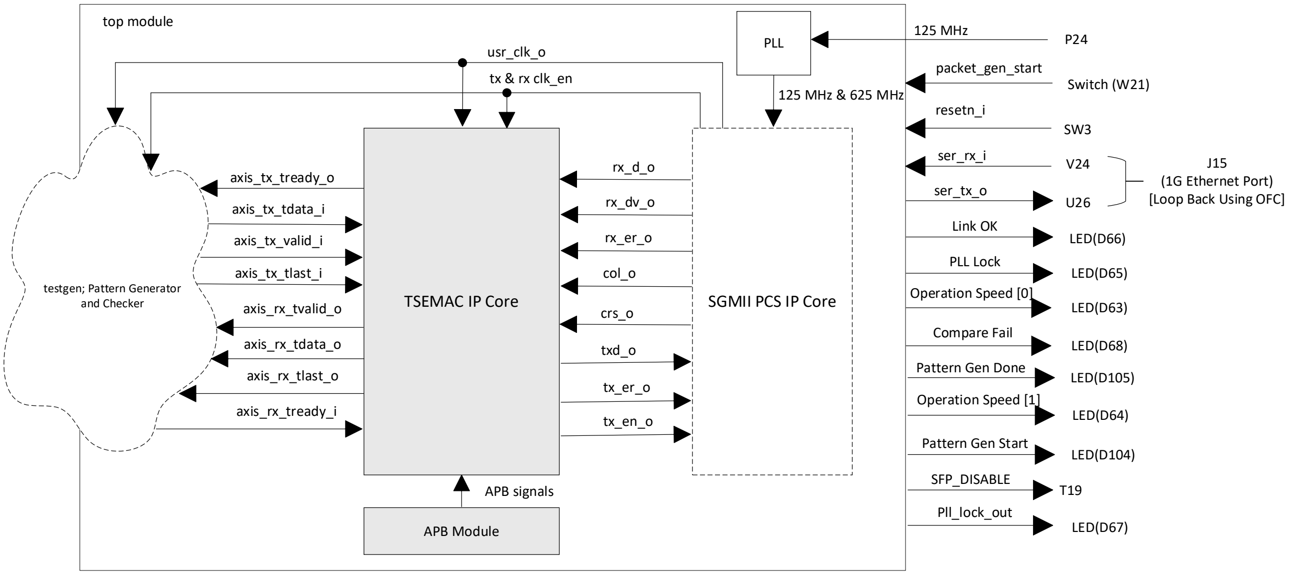


Figure 8.1. Top Module Architecture for Board Bring Up

8.1. Requirements

The following lists the requirements for the reference design implementation on board:

- CertusPro-NX Versa Board or CrossLink-NX PCIe Bridge Board
- USB cable for programming
- Power supply
- Optical fiber cables
- SFP

8.2. Device Hardware

This section describes two types of device hardware board for the reference design implementation—CertusPro-NX Versa Board and CrossLink-NX PCIe Bridge Board.

8.2.1. CertusPro-NX Versa Board

This section outlines the fundamental information for the CertusPro-NX Versa Board.

8.2.1.1. Status LED

The following figure shows the three status LEDs for the CertusPro-NX Versa Board.

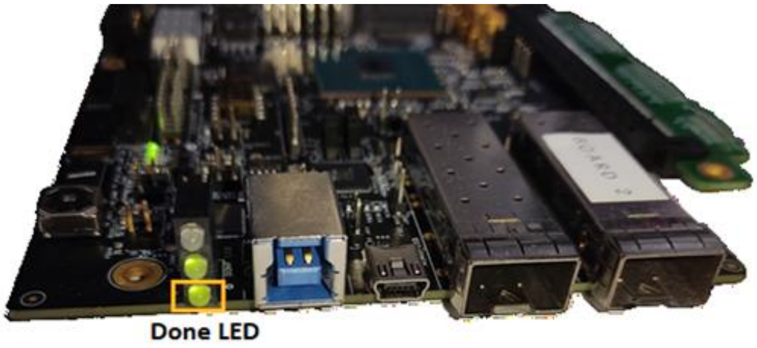


Figure 8.2. Status LEDs on the Board

The following table lists the description of each status LED.

Table 8.1. Board Status LED

SI.No.	Name	Description
1	DONE	Green: Lights up if successful configuration.
		Red: Board is not programmed.

8.2.1.2. Applying Power to the Board

Power is applied to the CertusPro-NX Versa Board using a 12 V power supply adaptor or from the PCIe Edge connector. Select the power supply source via the Power Selection Switch as shown in the following figure.

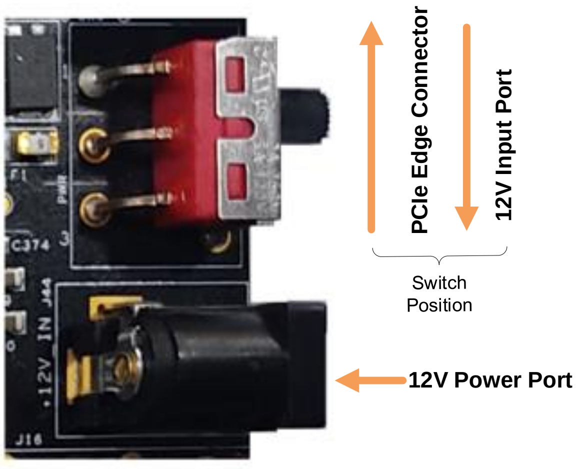


Figure 8.3. Power Supply and Power Switch on the Board

Table 8.2. Board Power Supply for CertusPro-NX Versa Board

Part Designator	Description
J44	12 V DC input DC jack
SW6	Power selection switch
F1	12 V DC input supply fuse

All the power LEDs light are turned on after applying 12 V power to the CertusPro-NX Versa Board, which indicates that the board is powered on and functioning (see the following figure). When the configuration is successfully completed, the DONE LED (D38-3 Bottom) lights up.

Caution: The CertusPro-NX Versa Board contains ESD-sensitive components. ESD safe practices must be followed while handling and using the development board.

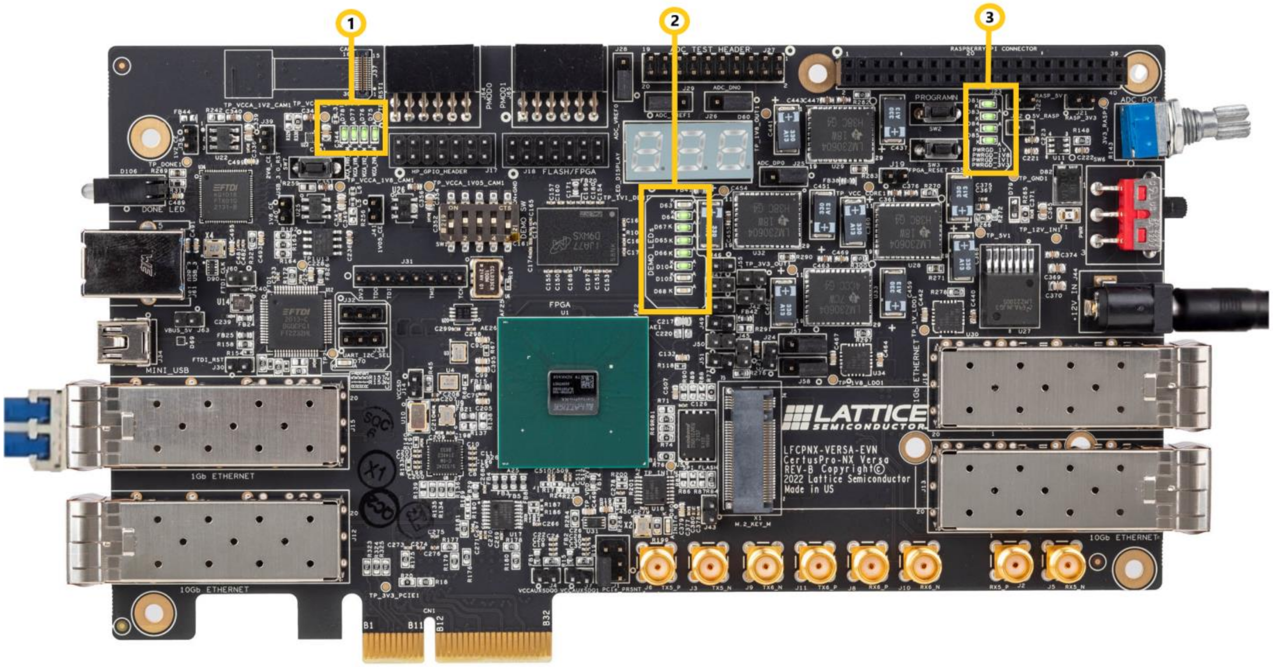


Figure 8.4. LEDs on the CertusPro-NX Versa Board

Table 8.3. Board LED Descriptions

Sl. No.	LEDs	Signal Name	Color	Purpose
1	D81	PWRGD 1V	Green	Lights up if voltage is present.
	D83	PWRGD 1V8	Green	Lights up if voltage is present.
	D84	PWRGD 1V1	Green	Lights up if voltage is present.
	D85	PWRGD 3V3	Green	Lights up if voltage is present.
	D63	PLL Lock	Green	Lights up when PLL lock asserts.
	D64	Link OK	Green	Lights up when the linkup is established.
2[1]	D67	Operation Speed0	Green	Light glows based on the operating speed. 10 Mbps: D67 and D65 LEDs will not glow. 100 Mbps: D67 LED glows. 1 Gbps: D65 LED glows.
	D65	Operation Speed1	Green	
	D66	Packet transmitting (TX)	Green	Blinks when packets are transmitting.
	D104	Packet receiving (RX)	Green	Blinks when packets are received.

Sl. No.	LEDs	Signal Name	Color	Purpose
	D105	Pattern Generation Completed	Green	Lights up when the pattern generator completes generating packet number defined.
	D68	Compare Fail	Green	Lights up when comparison fails/error in the received transaction.
3	AD78	1.05 V	Green	Lights up if voltage is present.
	AD77	2.8 V	Green	Lights up if voltage is present.
	AD76	1.2 V	Green	Lights up if voltage is present.
	AD75	1.8 V	Green	Lights up if voltage is present.

Notes:

1. These eight demo LEDs are mapped to the corresponding signals in the design file *top_module.v*. The LEDs are only meaningful after bit file programming.
2. Short 1-2 jumper at J32 and J33 for UART-I2C selection.
3. Table 8.4 shows the DIP switch configuration for CertusPro-NX device operational speed.

Table 8.4. Operation Speed Setting for CertusPro-NX Devices

DIP Switch 1	DIP Switch 2	Speed	LED Changes
0	0	10 Mbps	D67 and D65 are off.
0	1	100 Mbps	D67 lights up.
1	0	1 Gbps	D65 lights up.

8.2.2. CrossLink-NX PCIe Bridge Board

This section outlines the fundamental information for the CrossLink-NX PCIe Bridge Board.

8.2.2.1. Status LED

Figure 8.5 shows the status of the three front facial LEDs for the CrossLink-NX PCIe Bridge Board.

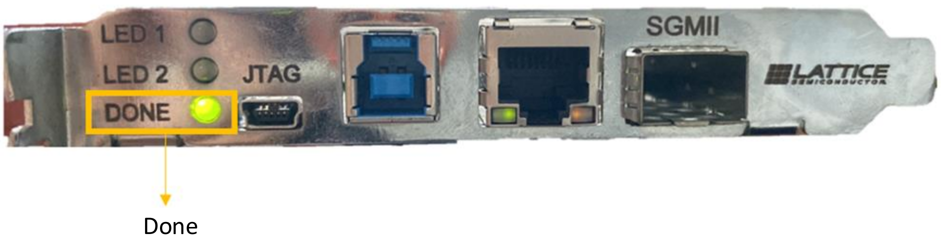


Figure 8.5. Front Facial LEDs Status for CrossLink-NX PCIe Bridge Board

Table 8.5. Front Facial LED Status

SI.No.	Name	Description
1	DONE	Green: FPGA status has been booted.

8.2.2.2. Applying Power to the Board

Power is applied to the CrossLink-NX PCIe Bridge Board using a 12 V power supply adapter or from the PCIe Edge Connector. Select the power supply source via the Power Selection Switch (SW2) as shown in Figure 8.6.

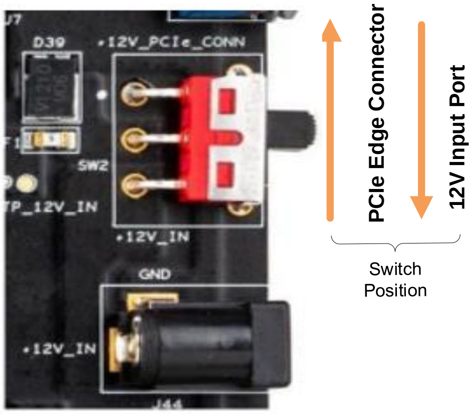


Figure 8.6. Power Supply for CrossLink-NX PCIe Bridge Board

Table 8.6. Power Supply for the CrossLink-NX PCIe Bridge Board

Part Designator	Description
J44	12 V DC input DC jack
SW2	Power selection switch

All the power LEDs light are turned on after applying 12 V power to the CrossLink-NX PCIe Bridge Board, which indicates that the board is powered on and functioning (see Figure 8.7.). When the configuration is successfully completed, the DONE LED (front facial most bottom) lights up.

Caution: The CrossLink-NX PCIe Bridge Board contains ESD-sensitive components. ESD safe practices must be followed while handling and using the development board.

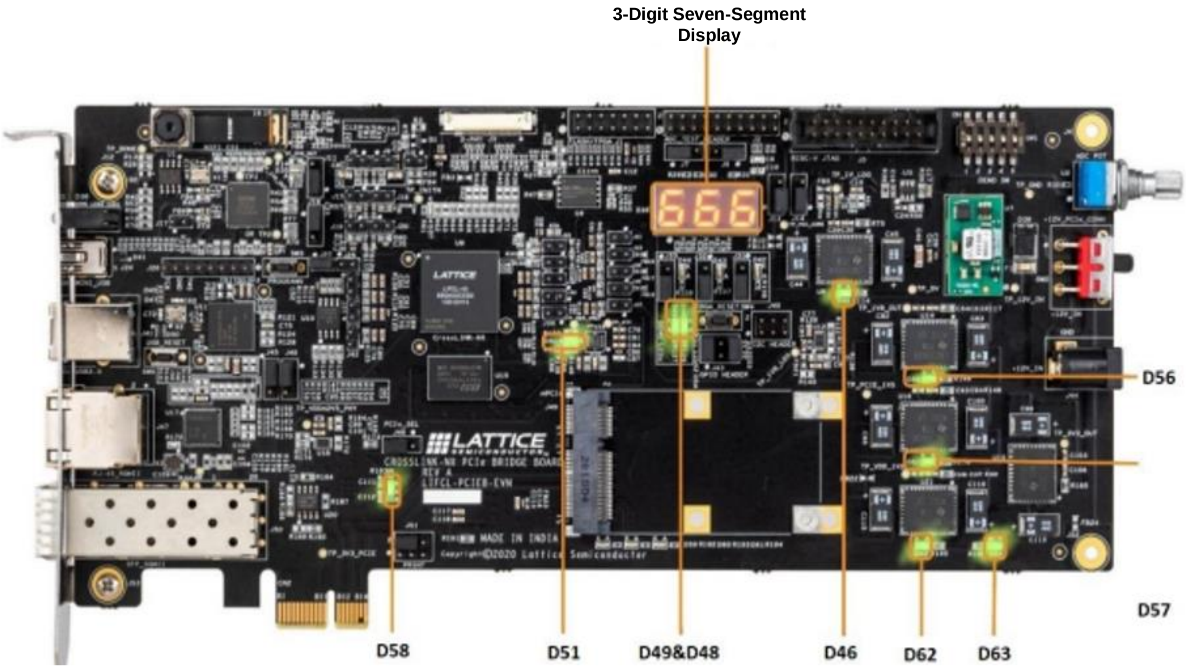


Figure 8.7. LEDs on the CrossLink-NX PCIe Bridge Board

Table 8.7. Board LED Description

Sl. No.	LEDs	Signal Name	Color	Purpose
1	D51	PWRGD 0.75V	Green	Lights up if DDR reference voltage is present.
	D46	PWRGD 1V	Green	Lights up if voltage is present.
	D62	PWRGD 1V2	Green	Lights up if voltage is present.
	D57	PWRGD 1V5	Green	Lights up if voltage is present.
	D56	PWRGD 1V8	Green	Lights up if voltage is present.
	D63	PWRGD 3V3	Green	Lights up if voltage is present.

The 7-segment LED glows according to the description in Figure 8.8 and Table 8.8.

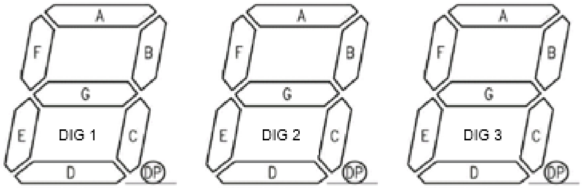


Figure 8.8. 7-Segment LED

Table 8.8. 7-Segment LED Description

Digit	Segment/LED	Description
1	A	Lights up when PLL lock asserts.
	B	Lights up when the linkup is established.
	C	Light glows based on the operating speed: 10 Mbps: C and D will not glow. 100 Mbps: C lights up, D will not glow. 1 Gbps: D lights up, C will not glow. Auto-negotiation: C and D light up.
	D	
	E	Blinks when packet is transmitting.
	F	Blinks when packet is receiving.
	G	Lights up when the pattern generator completes generating packet number defined.
	DP	N/A
2	D49	Lights up when comparison fails or error in the received transaction.

Note: These demo LEDs (excluding DP LED) are mapped to corresponding signals in the design file *top_module.v*. The LEDs are only meaningful after bit file programming. Toggle the DIP switch 1. Refer to Figure 8.19 to start the pattern generator.

The following table shows the DIP switch configuration for CertusPro-NX device operational speed.

Table 8.9. Operation Speed Setting for CrossLink-NX Devices

DIP Switch 4	DIP Switch 5	Speed	LED Changes
0	0	10 Mbps	7-Segment DIG 1 – C and D will not glow.
0	1	100 Mbps	7-Segment DIG1 – C lights up, D will not glow.
1	0	1 Gbps	7-Segment DIG 1 – D lights up, C will not glow.

8.3. Board Testing

This section provides the details of the reference design hardware testing on the board.

8.3.1. Bit File Programming

This section guides you through the process of uploading the .bit file.

Make sure the board is connected to the PC with the mini USB Type A cable, as shown in Figure 8.9 and Figure 8.10.

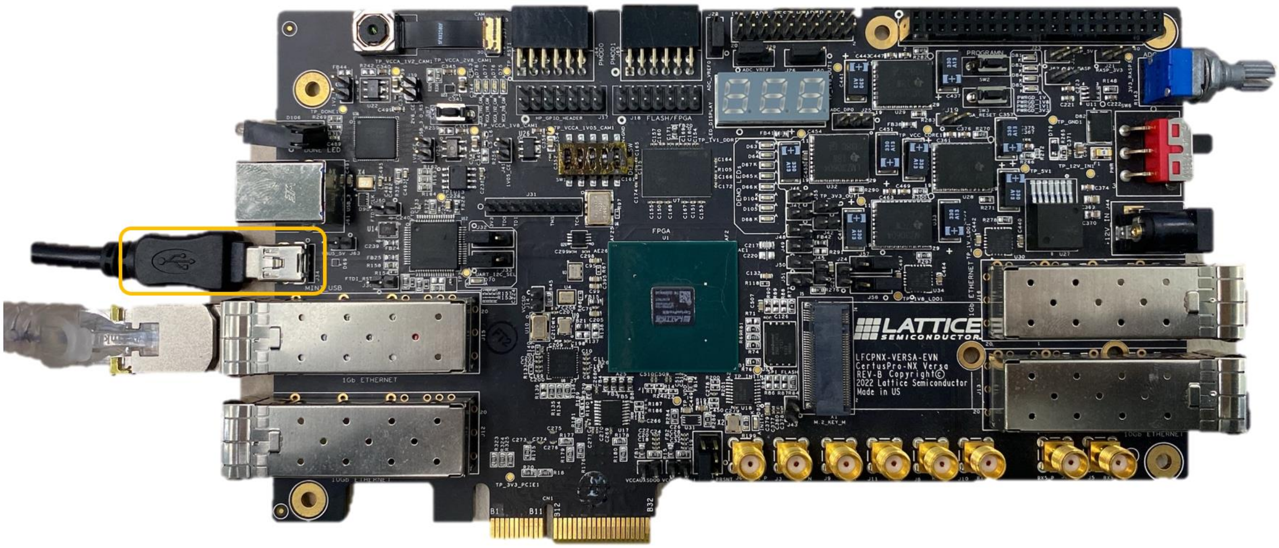


Figure 8.9. CertusPro-NX Versa Board

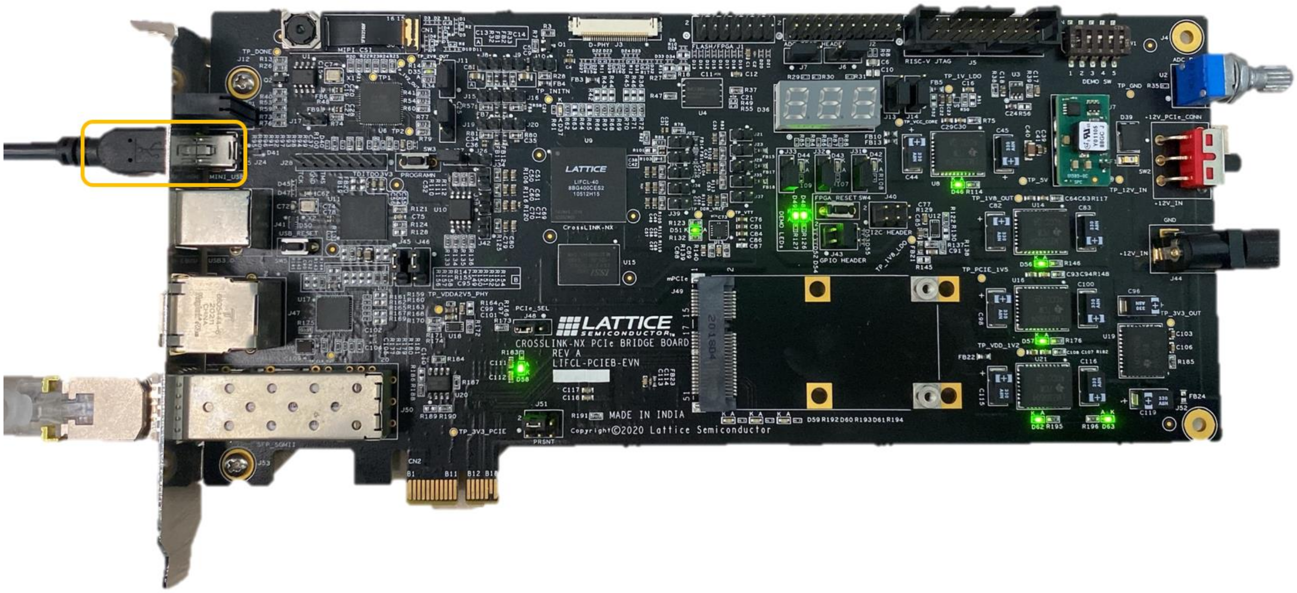
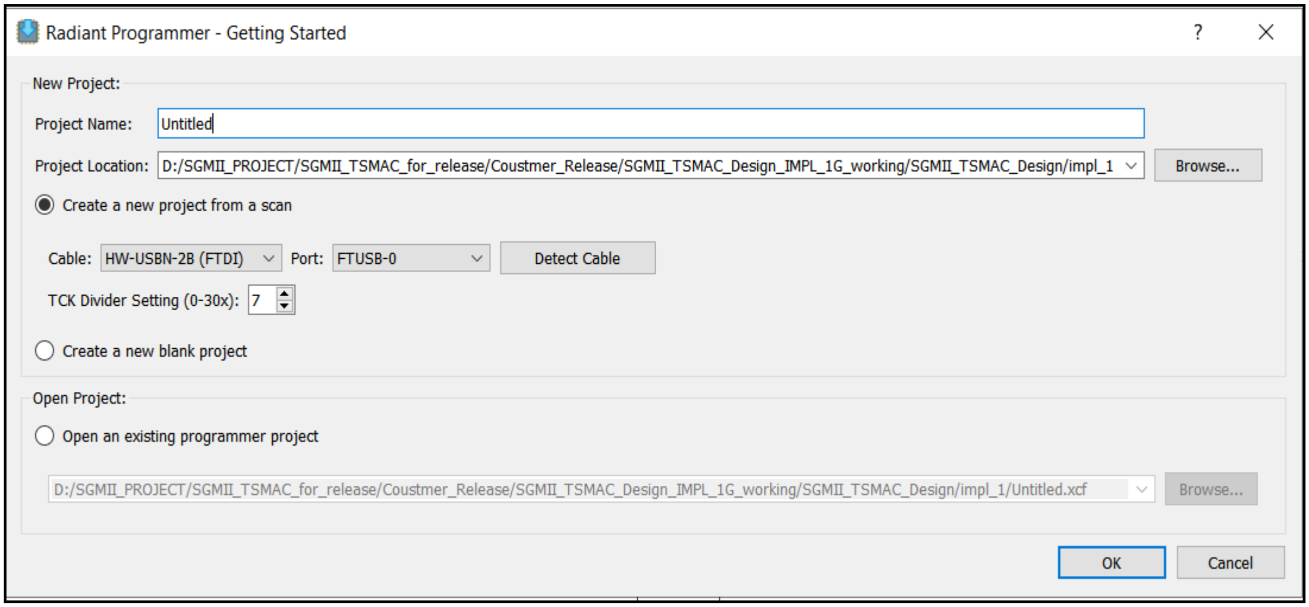

MINI_USB
CROSSLINK-NX PCIe BRIDGE BOARD
REV A
LIFCL-PCIEB-EVN
MADE IN INDIA
Copyright©2020 Lattice Semiconductor

Figure 8.10. CrossLink-NX PCIe Bridge Board

1. Click on the Radiant Programmer icon ( ) if the project is already opened inside the Radiant software. Otherwise, launch the standalone Radiant Programmer. Enter the **Project Name**, **Project Location**, and click **OK**.



Radiant Programmer - Getting Started

New Project:

Project Name: Untitled

Project Location: D:/SGMII_PROJECT/SGMII_TSMAC_for_release/Coustmer_Release/SGMII_TSMAC_Design_IMPL_1G_working/SGMII_TSMAC_Design/impl_1

● Create a new project from a scan

Cable: HW-USBN-2B (FTDI) Port: FTUSB-0

TCK Divider Setting (0-30x): 7

○ Create a new blank project

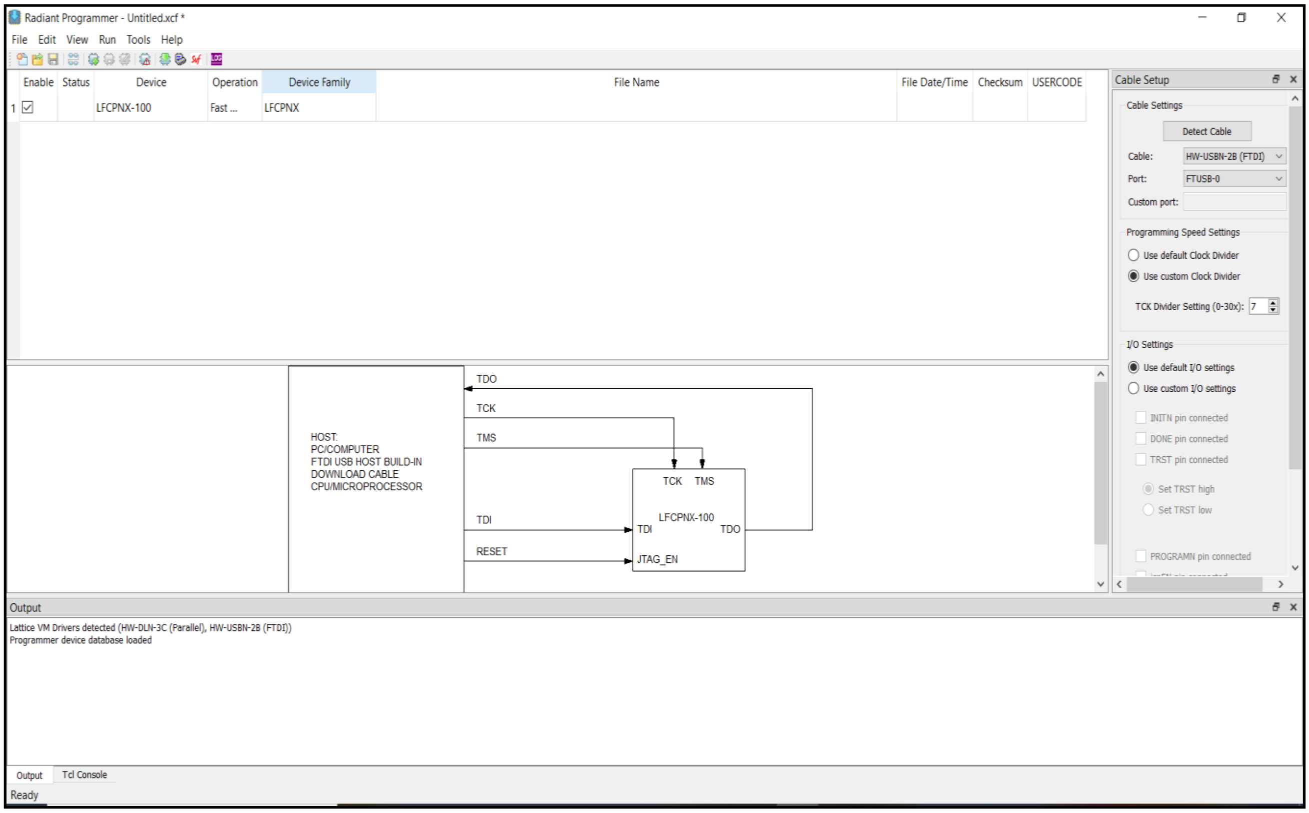
Open Project:

○ Open an existing programmer project

D:/SGMII_PROJECT/SGMII_TSMAC_for_release/Coustmer_Release/SGMII_TSMAC_Design_IMPL_1G_working/SGMII_TSMAC_Design/impl_1/Untitled.xcf

Figure 8.11. Standalone Radiant Programmer

2. The Radiant Programmer window is shown in Figure 8.12.



Radiant Programmer - Untitled.xcf *

	Enable	Status	Device	Operation	Device Family	File Name	File Date/Time	Checksum	USERCODE
1	☑		LFCPNX-100	Fast ...	LFCPNX				

Cable Setup

Cable Settings

Cable: HW-USBN-2B (FTDI)

Port: FTUSB-0

Custom port:

Programming Speed Settings

○ Use default Clock Divider

● Use custom Clock Divider

TCK Divider Setting (0-30x): 7

I/O Settings

● Use default I/O settings

○ Use custom I/O settings

□ INITN pin connected

□ DONE pin connected

□ TRST pin connected

○ Set TRST high

○ Set TRST low

□ PROGRAMN pin connected

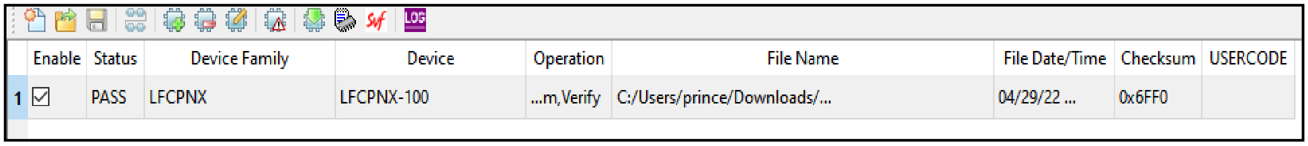
Output

```
Lattice VM Drivers detected (HW-DLN-3C (Parallel), HW-USBN-2B (FTDI))
Programmer device database loaded
```

Ready

Figure 8.12. Radiant Programmer Window

3. Configure the **Device Family**, **Device**, and the bit file fields.



	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	☑	PASS	LFCPNX	LFCPNX-100	...m,Verify	C:/Users/prince/Downloads/...	04/29/22 ...	0x6FF0	

Figure 8.13. Radiant Programmer: Configure the Settings

4. Click on the program device toolbar icon as shown in Figure 8.14.

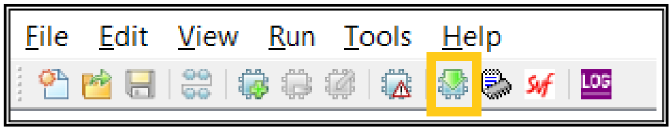
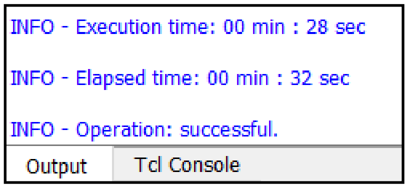


Figure 8.14. Program Device Toolbar Icon

5. On the output window, verify that the status of the program device is *Operation: successful*, as shown in Figure 8.15.



```
INFO - Execution time: 00 min : 28 sec

INFO - Elapsed time: 00 min : 32 sec

INFO - Operation: successful.
```

Figure 8.15. Message on Successful Programming

6. After programming, restart the board and check the status of done LED. It indicates that the .bit file has been booted successfully in the FPGA.

8.3.2. External Loopback Setup

1. Connect the optical cable back-to-back (TX-RX), as shown in Figure 8.16.

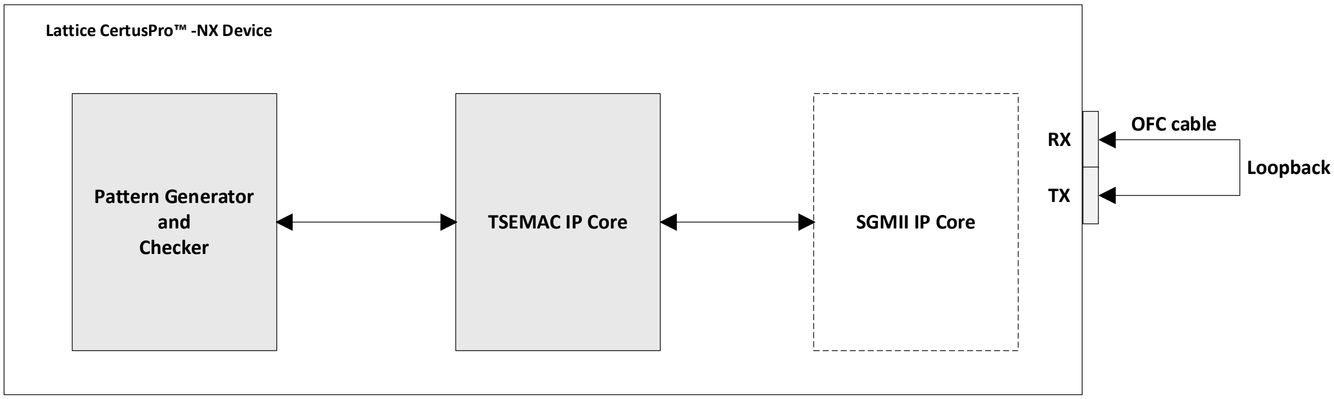


Figure 8.16. External Loopback Connection

2. Upon back-to-back connection, press the reset switch (SW3) and the Link OK LED glows green.
3. For the CertusPro-NX Versa Board, toggle switch 5 in SW1 to start the pattern generator (see Figure 8.17) and toggle switch 1 and 2 in SW1 to change the operational speed (see Figure 8.19).
4. Table 8.4 shows the DIP switch configuration of the operational rate for the CertusPro-NX Versa board.
5. For the CrossLink-NX PCIe Bridge Board, toggle switch 1 in SW1 to start the pattern generator (see Figure 8.19). and toggle switch 4 and 5 in SW1 to change the operational speed (see Figure 8.20).

6. Table 8.9 shows the DIP switch configuration of operational rate for Crosslink-NX PCIe bridge board.

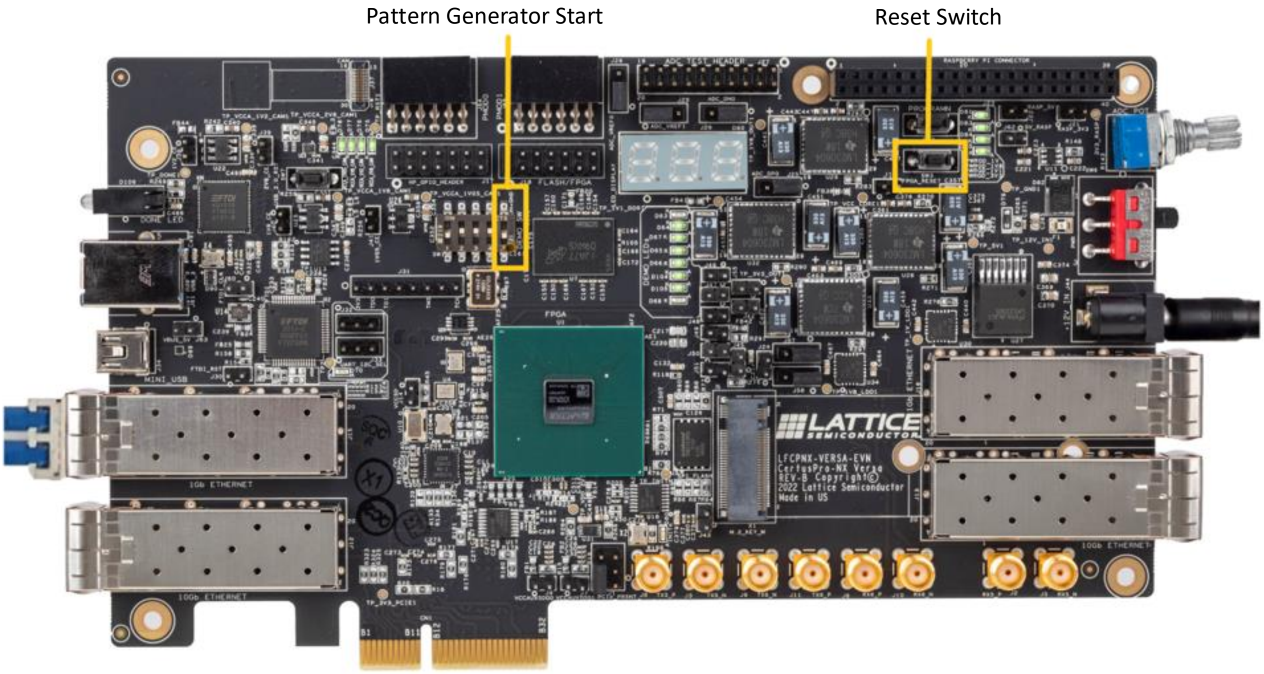


Figure 8.17. Toggle Switch 5 in SW1 to Initiate Pattern Generation for CertusPro-NX Versa Board

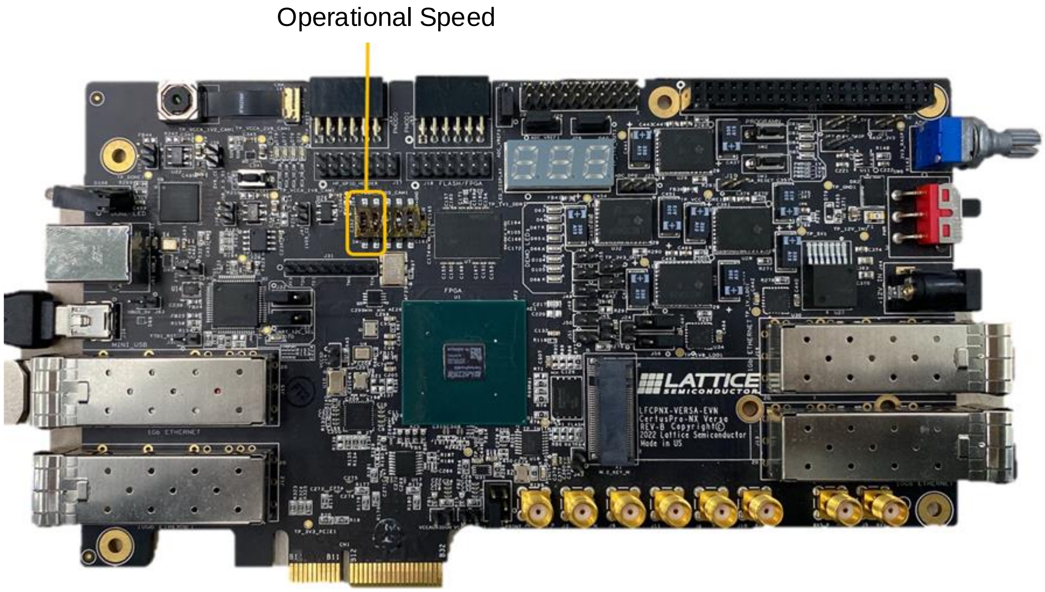


Figure 8.18. Toggle Switch 1 and Switch 2 in SW1 to change the speed operation for CertusPro-NX Versa Board

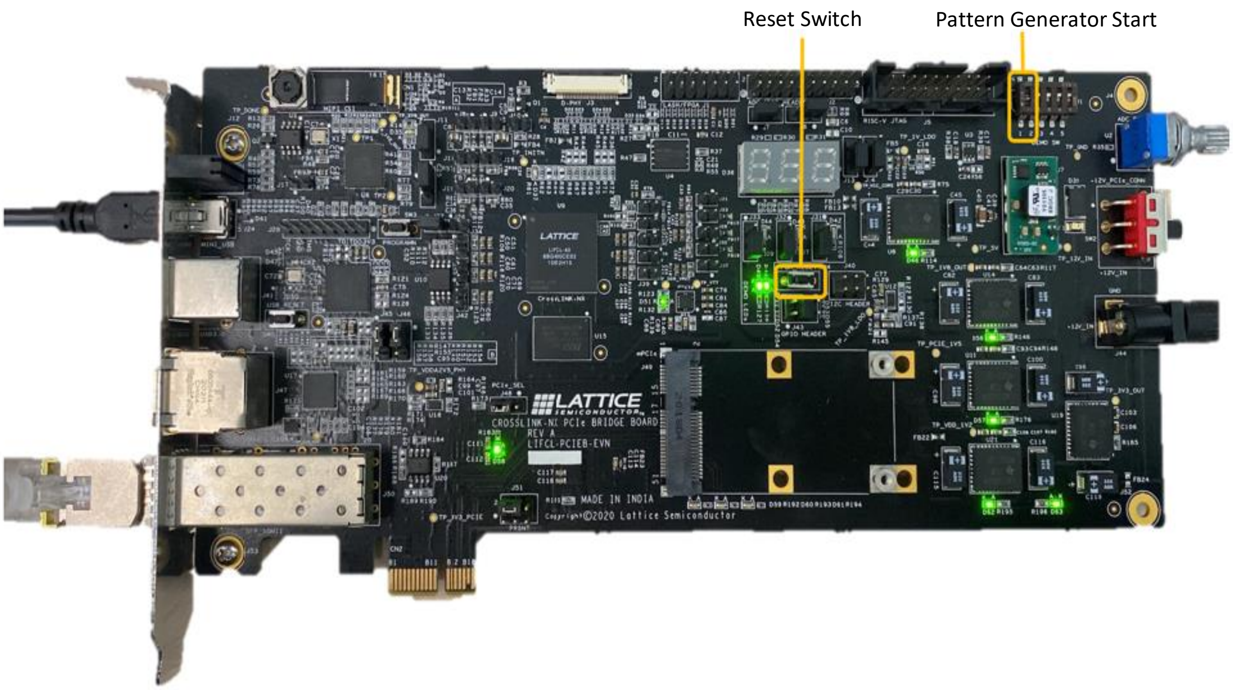


Figure 8.19. Toggle Switch 1 in SW1 to Initiate Pattern Generation for CrossLink-NX PCIe Bridge Board

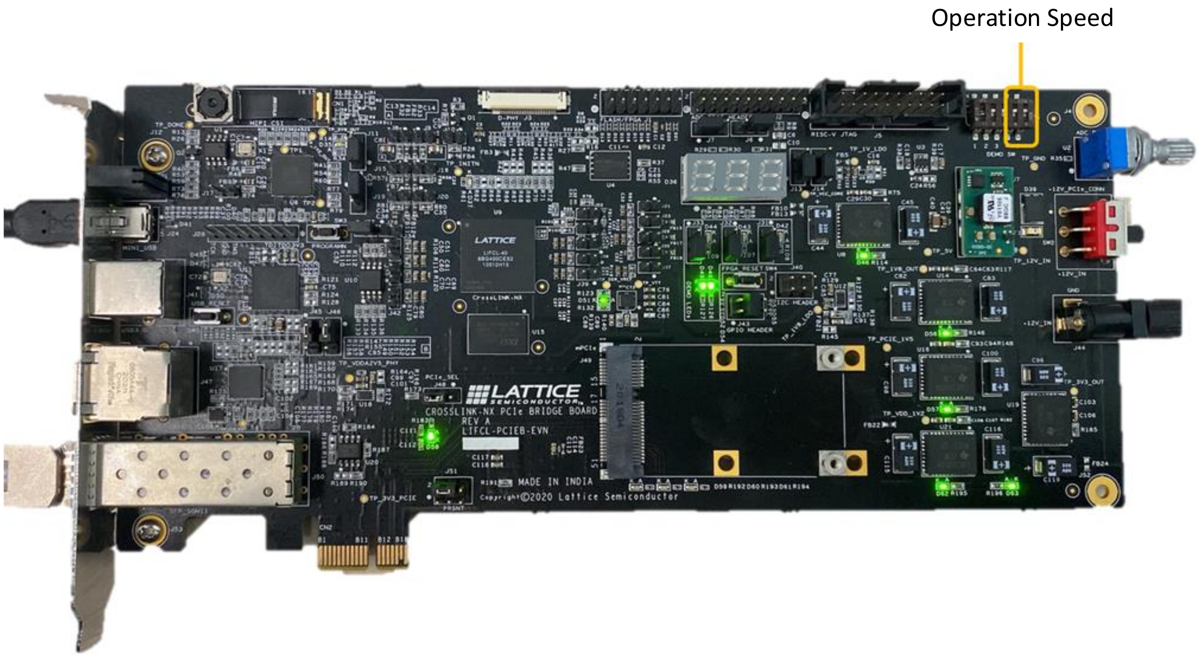


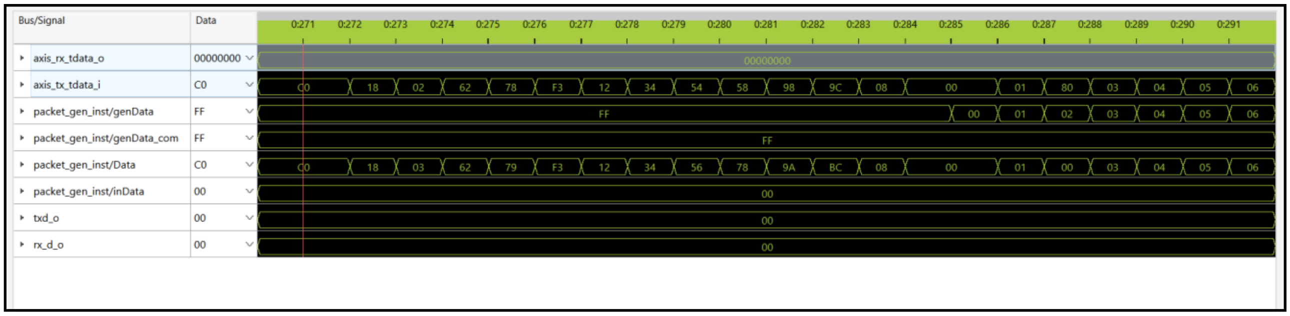
Figure 8.20. Toggle Switch 4 and Switch 5 in SW1 to Change the Speed Operation for CrossLink-NX Versa Board

8.3.3. Waveforms Captured Using the Reveal Tool

This section describes the waveforms that are captured using the Reveal tool in the Radiant software.

The following describes the Pattern Generator Module output to the TSEMAC through the AXI interface:

- The generated pattern (Data) is transmitted to the TSEMAC through the axis_tx_tdata_I interface.
- The genData is a counter that starts after generating the fixed patterns (destination and source MAC address, IP length).

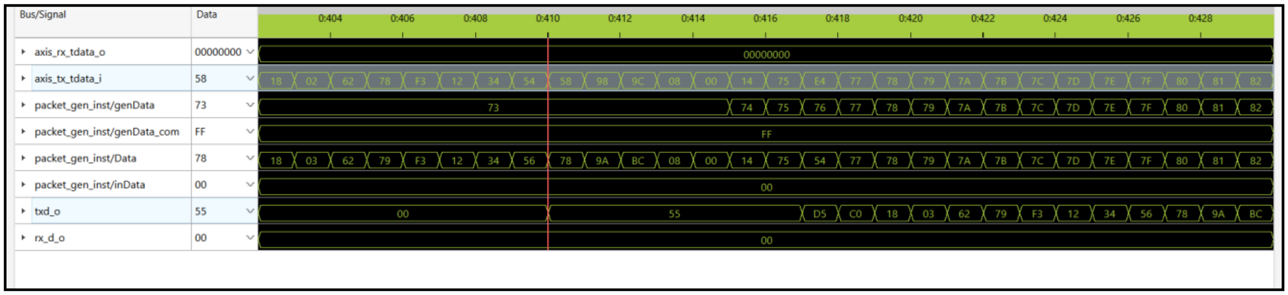


Bus/Signal	Data
axis_rx_tdata_o	00000000
axis_tx_tdata_i	C0
packet_gen_inst/genData	FF
packet_gen_inst/genData_com	FF
packet_gen_inst/Data	C0
packet_gen_inst/inData	00
txd_o	00
rx_d_o	00

Figure 8.21. Pattern Generator Transmitting Packets to TSEMAC through the AXI-Stream Interface

TSEMAC Transmits Data to the SGMII Module

The TSEMAC adds the preamble and SFD before transmitting the data to the SGMII module, as shown in Figure 8.22.



Bus/Signal	Data
axis_rx_tdata_o	00000000
axis_tx_tdata_i	58
packet_gen_inst/genData	73
packet_gen_inst/genData_com	FF
packet_gen_inst/Data	78
packet_gen_inst/inData	00
txd_o	55
rx_d_o	00

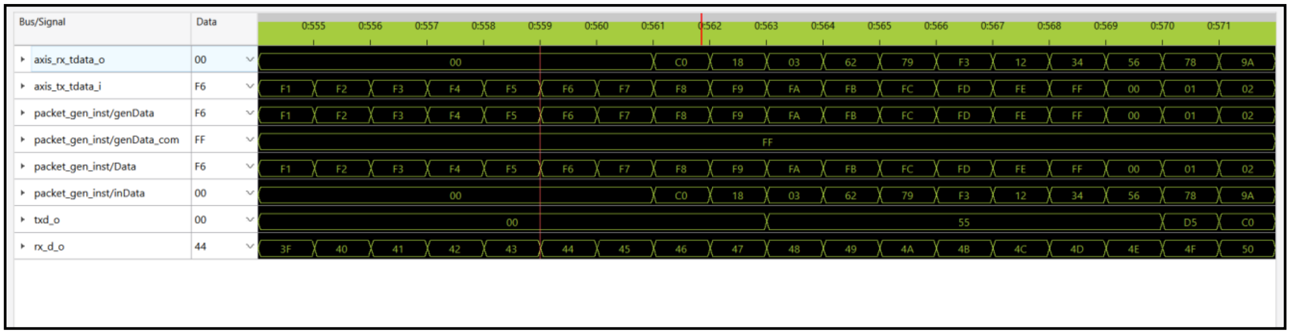
0:404 0:406 0:408 0:410 0:412 0:414 0:416 0:418 0:420 0:422 0:424 0:426 0:428
00000000
18 02 62 78 F3 12 34 54 58 98 9C 08 00 14 75 E4 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82
73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82
FF
18 03 62 79 F3 12 34 56 78 9A BC 08 00 14 75 54 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82
00
00 55 D5 C0 18 03 62 79 F3 12 34 56 78 9A BC
00

Figure 8.22. TSEMAC Transmitting Packets to the SGMII Module

TSEMAC Receives Signal from the SGMII Module and Transmits the Signal to the Pattern Checker

The TSEMAC receives signal from the SGMII module and transmits the signal to the RX FIFO by removing the preamble and SFD internally. The TSEMAC shares the received signal through the AXI interface (axis_rx_tdata_o) and sends it back to the Pattern_gen_checker module for the received pattern to be verified.

The inData signal is the input pattern signal from the TSEMAC through the AXI-Stream interface, which is used in the pattern checker.



Bus/Signal	Data
axis_rx_tdata_o	00
axis_tx_tdata_i	F6
packet_gen_inst/genData	F6
packet_gen_inst/genData_com	FF
packet_gen_inst/Data	F6
packet_gen_inst/inData	00
txd_o	00
rx_d_o	44

0:555 0:556 0:557 0:558 0:559 0:560 0:561 0:562 0:563 0:564 0:565 0:566 0:567 0:568 0:569 0:570 0:571
00 C0 18 03 62 79 F3 12 34 56 78 9A
F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 00 01 02
F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 00 01 02
FF
F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 00 01 02
00 C0 18 03 62 79 F3 12 34 56 78 9A
00 55 D5 C0
3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50

Figure 8.23. TSEMAC Receives Packets from the SGMII and Sends the Packets to the Pattern Checker through the AXI-Stream Interface

If there is any invalid or missing data coming from the TSEMAC to the Pattern_gen_checker module, the cmpFail signal or LED will be asserted.

Failed Condition:

In the loopback condition, you can remove one end of the optical cable to produce the error condition. In this case, the cmpFail LED will glow. To reset, press the Reset button (SW3) and the connections will re-establish again.

The cmpFail LED lights up when the failed condition is met, as shown in Figure 8.24. for the CertusPro-NX Versa Board and Figure 8.25. for the CrossLink-NX PCIe Bridge Board.

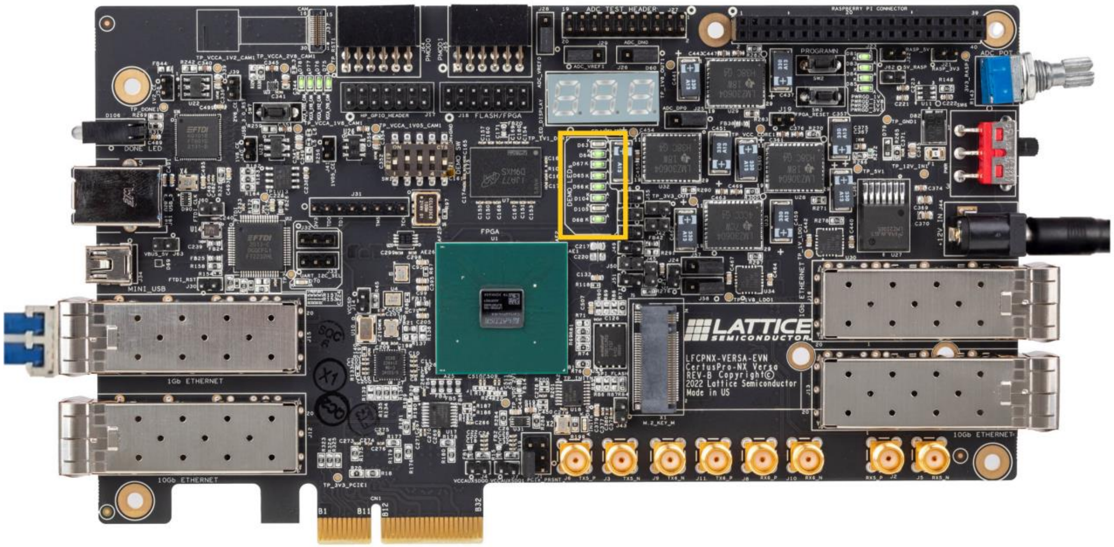


Figure 8.24. cmpFail LED Lights Up on the CertusPro-NX Versa Board

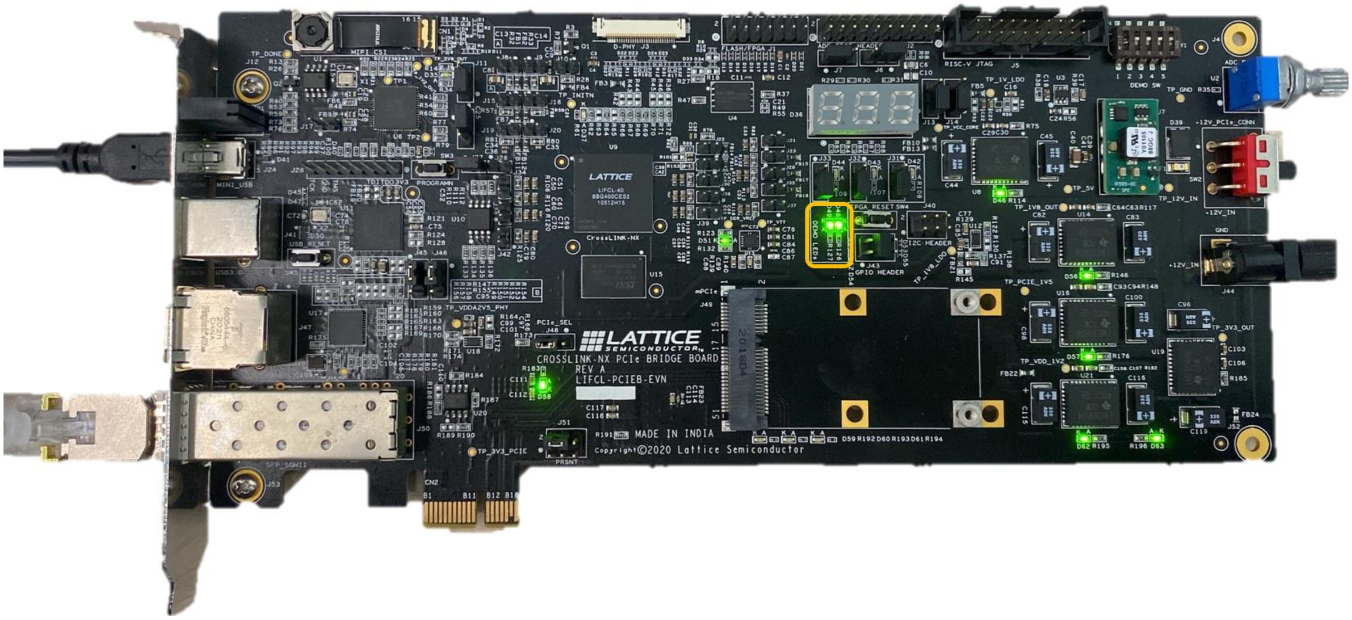


Figure 8.25. cmpFail LED Lights Up on the CrossLink-NX PCIe Bridge Board

Continues Traffic Disabled (CONTINUOUS_TRAFFIC=0)

When the continuous traffic generator condition is disabled, the pattern generator will only transmit the number of packets specified in the parameter NUM_PKT, and the pattern generator done LED will turn on after all of the packets have been sent, as shown in Figure 8.26 for the CertusPro-NX Versa Board and Figure 8.27 for the CrossLink-NX PCIe Bridge Board.

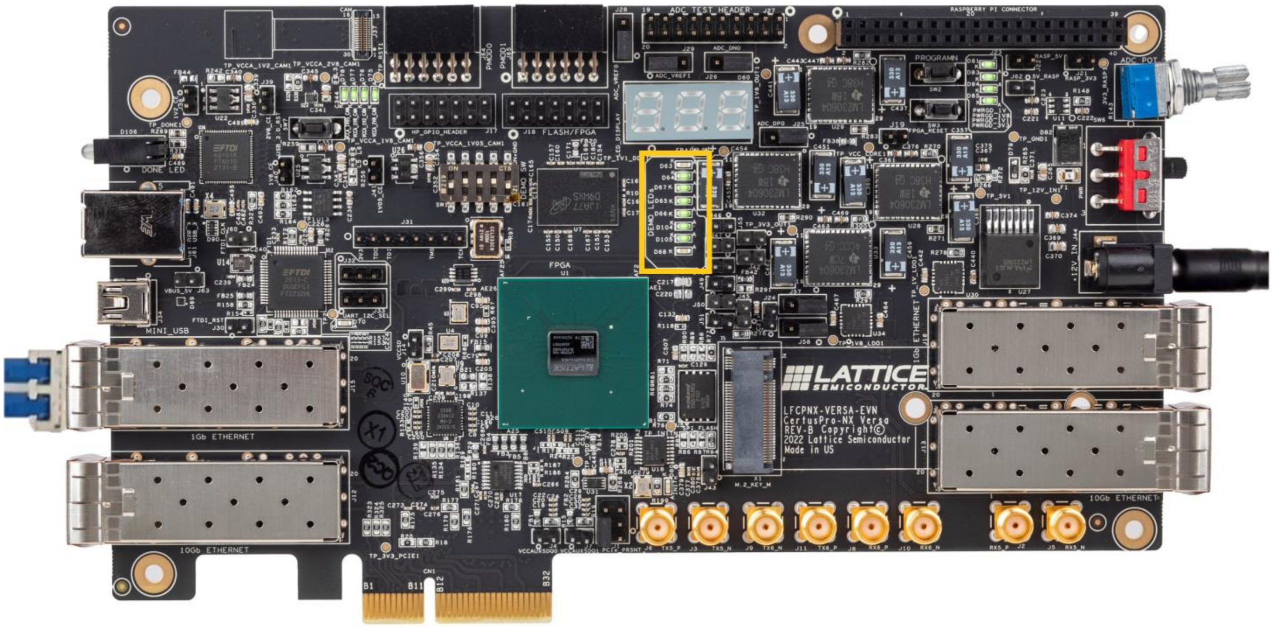


Figure 8.26. Pattern_gen_done LED Glows on Continuous Traffic Disabled on the CertusPro-NX Versa Board

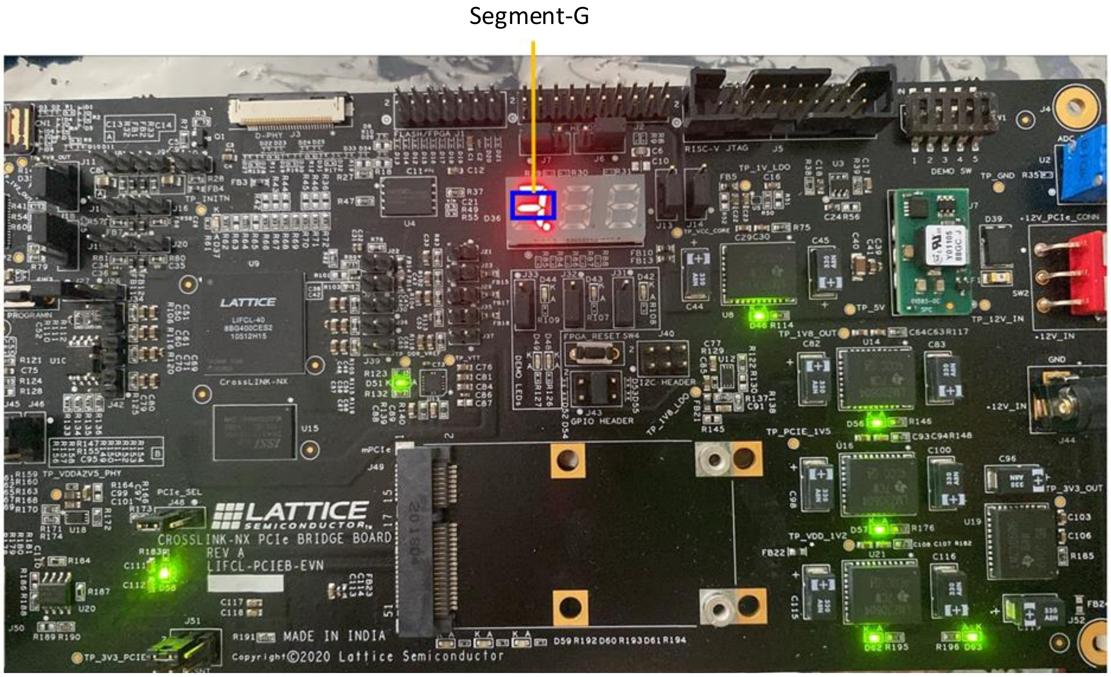


Figure 8.27. Pattern_gen_done LED Glows on Continuous Traffic Disabled on the CrossLink-NX PCIe Bridge Board

8.3.4. Pattern Generator Analysis Using the Wireshark Software

The Wireshark software is used to capture packets generated from the pattern generator with a fiber optic cable that communicates with the PC through a switch between the board and the PC. This switch converts optical signals into electrical signals and vice versa. The RTL checker fails to operate properly because the PC is transmitting an Address Resolution Protocol (ARP) signal. As a result, the RTL checker will not work as expected and will turn on the Compare Fail LED.

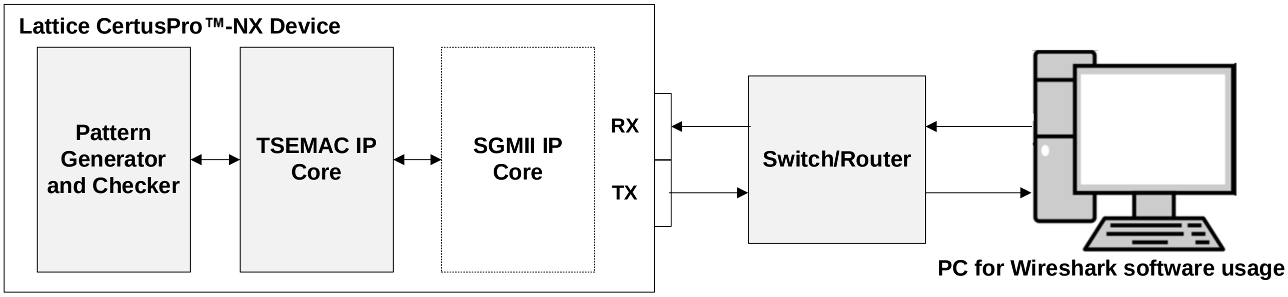
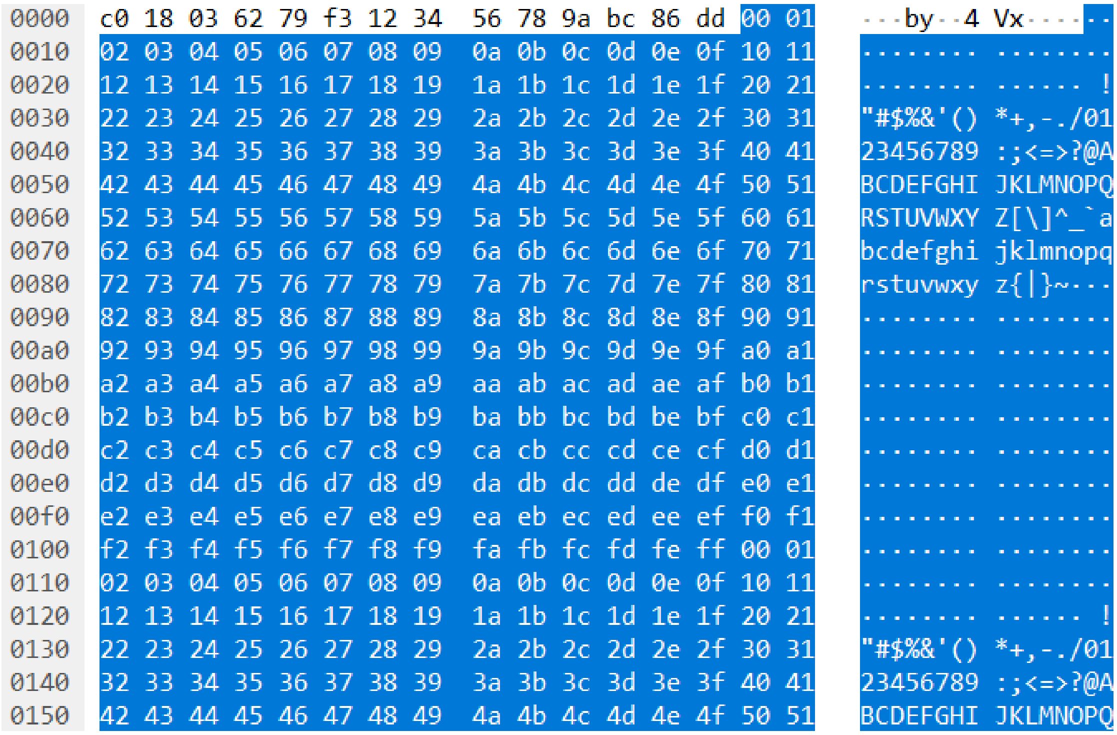

Lattice CertusPro™-NX Device
Pattern Generator and Checker
TSEMAC IP Core
SGMII IP Core
RX
TX
Switch/Router
PC for Wireshark software usage

Figure 8.28. Pattern Generator Checking through Wireshark Software

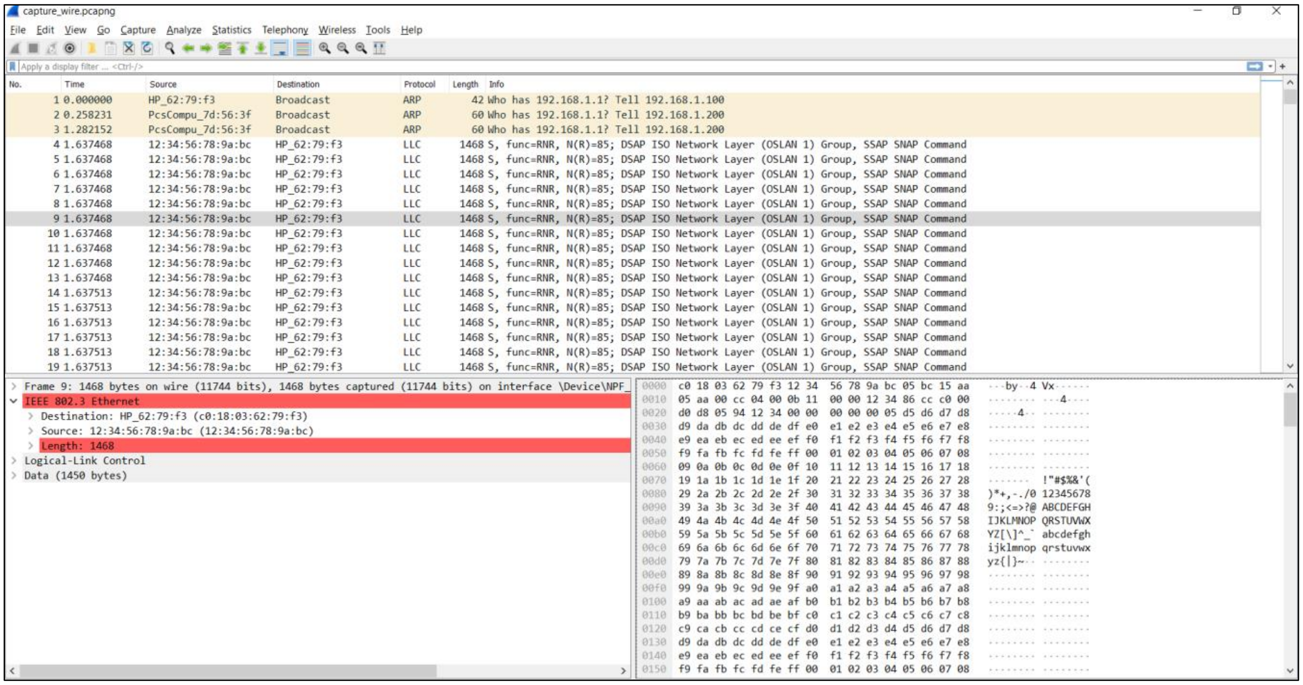
You can analyze the pattern generator data using the Wireshark software. The data captured by the Wireshark software must match the pattern defined in the pattern generator file— *Pattern_gen_checker.v*. Figure 8.29 shows an example of a frame captured by the Wireshark software.

Note that the frame content of the pattern generator has been modified during testing and might not match with the original frame pattern in the reference design package.



```
0000  c0 18 03 62 79 f3 12 34  56 78 9a bc 86 dd 00 01   ···by··4 Vx······
0010  02 03 04 05 06 07 08 09  0a 0b 0c 0d 0e 0f 10 11   ········ ········
0020  12 13 14 15 16 17 18 19  1a 1b 1c 1d 1e 1f 20 21   ········ ······ !
0030  22 23 24 25 26 27 28 29  2a 2b 2c 2d 2e 2f 30 31   "#$%&'() *+,-./01
0040  32 33 34 35 36 37 38 39  3a 3b 3c 3d 3e 3f 40 41   23456789 :;<=>?@A
0050  42 43 44 45 46 47 48 49  4a 4b 4c 4d 4e 4f 50 51   BCDEFGHI JKLMNOPQ
0060  52 53 54 55 56 57 58 59  5a 5b 5c 5d 5e 5f 60 61   RSTUVWXY Z[\]^_`a
0070  62 63 64 65 66 67 68 69  6a 6b 6c 6d 6e 6f 70 71   bcdefghi jklmnopq
0080  72 73 74 75 76 77 78 79  7a 7b 7c 7d 7e 7f 80 81   rstuvwxy z{|}~···
0090  82 83 84 85 86 87 88 89  8a 8b 8c 8d 8e 8f 90 91   ········ ········
00a0  92 93 94 95 96 97 98 99  9a 9b 9c 9d 9e 9f a0 a1   ········ ········
00b0  a2 a3 a4 a5 a6 a7 a8 a9  aa ab ac ad ae af b0 b1   ········ ········
00c0  b2 b3 b4 b5 b6 b7 b8 b9  ba bb bc bd be bf c0 c1   ········ ········
00d0  c2 c3 c4 c5 c6 c7 c8 c9  ca cb cc cd ce cf d0 d1   ········ ········
00e0  d2 d3 d4 d5 d6 d7 d8 d9  da db dc dd de df e0 e1   ········ ········
00f0  e2 e3 e4 e5 e6 e7 e8 e9  ea eb ec ed ee ef f0 f1   ········ ········
0100  f2 f3 f4 f5 f6 f7 f8 f9  fa fb fc fd fe ff 00 01   ········ ········
0110  02 03 04 05 06 07 08 09  0a 0b 0c 0d 0e 0f 10 11   ········ ········
0120  12 13 14 15 16 17 18 19  1a 1b 1c 1d 1e 1f 20 21   ········ ······ !
0130  22 23 24 25 26 27 28 29  2a 2b 2c 2d 2e 2f 30 31   "#$%&'() *+,-./01
0140  32 33 34 35 36 37 38 39  3a 3b 3c 3d 3e 3f 40 41   23456789 :;<=>?@A
0150  42 43 44 45 46 47 48 49  4a 4b 4c 4d 4e 4f 50 51   BCDEFGHI JKLMNOPQ
```

Figure 8.29. Wireshark Software: Data Captured



capture_wire.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.   Time      Source             Destination   Protocol Length Info
    1 0.000000  HP_62:79:f3        Broadcast     ARP        42 Who has 192.168.1.1? Tell 192.168.1.100
    2 0.258231  PcsCompu_7d:56:3f  Broadcast     ARP        60 Who has 192.168.1.1? Tell 192.168.1.200
    3 1.282152  PcsCompu_7d:56:3f  Broadcast     ARP        60 Who has 192.168.1.1? Tell 192.168.1.200
    4 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
    5 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
    6 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
    7 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
    8 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
    9 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   10 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   11 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   12 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   13 1.637468  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   14 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   15 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   16 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   17 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   18 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
   19 1.637513  12:34:56:78:9a:bc  HP_62:79:f3   LLC      1468 S, func=RNR, N(R)=85; DSAP ISO Network Layer (OSLAN 1) Group, SSAP SNAP Command
```

```
> Frame 9: 1468 bytes on wire (11744 bits), 1468 bytes captured (11744 bits) on interface \Device\NPF_
v IEEE 802.3 Ethernet
  > Destination: HP_62:79:f3 (c0:18:03:62:79:f3)
  > Source: 12:34:56:78:9a:bc (12:34:56:78:9a:bc)
  > Length: 1468
> Logical-Link Control
> Data (1450 bytes)
```

```
0000  c0 18 03 62 79 f3 12 34  56 78 9a bc 05 bc 15 aa   ···by··4 Vx······
0010  05 aa 00 cc 04 00 0b 11  00 00 12 34 86 cc c0 00   ········ ···4····
0020  d0 d8 05 94 12 34 00 00  00 00 00 05 d5 d6 d7 d8   ·····4·· ········
0030  d9 da db dc dd de df e0  e1 e2 e3 e4 e5 e6 e7 e8   ········ ········
0040  e9 ea eb ec ed ee ef f0  f1 f2 f3 f4 f5 f6 f7 f8   ········ ········
0050  f9 fa fb fc fd fe ff 00  01 02 03 04 05 06 07 08   ········ ········
0060  09 0a 0b 0c 0d 0e 0f 10  11 12 13 14 15 16 17 18   ········ ········
0070  19 1a 1b 1c 1d 1e 1f 20  21 22 23 24 25 26 27 28   ·······   !"#$%&'(
0080  29 2a 2b 2c 2d 2e 2f 30  31 32 33 34 35 36 37 38   )*+,-./0 12345678
0090  39 3a 3b 3c 3d 3e 3f 40  41 42 43 44 45 46 47 48   9:;<=>?@ ABCDEFGH
00a0  49 4a 4b 4c 4d 4e 4f 50  51 52 53 54 55 56 57 58   IJKLMNOP QRSTUVWX
00b0  59 5a 5b 5c 5d 5e 5f 60  61 62 63 64 65 66 67 68   YZ[\]^_` abcdefgh
00c0  69 6a 6b 6c 6d 6e 6f 70  71 72 73 74 75 76 77 78   ijklmnop qrstuvwx
00d0  79 7a 7b 7c 7d 7e 7f 80  81 82 83 84 85 86 87 88   yz{|}~·· ········
00e0  89 8a 8b 8c 8d 8e 8f 90  91 92 93 94 95 96 97 98   ········ ········
00f0  99 9a 9b 9c 9d 9e 9f a0  a1 a2 a3 a4 a5 a6 a7 a8   ········ ········
0100  a9 aa ab ac ad ae af b0  b1 b2 b3 b4 b5 b6 b7 b8   ········ ········
0110  b9 ba bb bc bd be bf c0  c1 c2 c3 c4 c5 c6 c7 c8   ········ ········
0120  c9 ca cb cc cd ce cf d0  d1 d2 d3 d4 d5 d6 d7 d8   ········ ········
0130  d9 da db dc dd de df e0  e1 e2 e3 e4 e5 e6 e7 e8   ········ ········
0140  e9 ea eb ec ed ee ef f0  f1 f2 f3 f4 f5 f6 f7 f8   ········ ········
0150  f9 fa fb fc fd fe ff 00  01 02 03 04 05 06 07 08   ········ ········
```

Figure 8.30. Wireshark Software: Frames Captured

8.4. Constraint File (.pdc)

The constraint file (*.pdc*) of the reference design is in the following folder:

<reference design folder>/constraints/tsemac_sgmii_constraints.pdc

References

- Lattice Radiant Software User Guide
- TSEMAC IP User Guide (FPGA-IPUG-02084)
- SGMII and Gb Ethernet PCS IP User Guide (FPGA-IPUG-02077)
- Lattice Radiant Timing Constraints Methodology Application Note (FPGA-AN-02059)
- CertusPro-NX web page
- CrossLink-NX web page
- Tri-Speed Ethernet MAC IP Core web page
- SGMII and Gb Ethernet PCS IP Core web page
- 10Gb Ethernet MAC IP Core web page
- Lattice Insights web page for 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

Revision 1.3, September 2024

Section	Change Summary
Abbreviations in This Document	Replaced *acronyms* with *abbreviations* in this section.
Features	Added the following new feature to the TSEMAC IP Features list: *RISC-V to configure SFP PHY register to SGMII mode via I2C controller.*
Functional Description	• Added Figure 3.2. Overall Reference Design Block Diagram and a new paragraph that describes the reference design diagram. • Updated Table 3.1. Signal Description. • Added a new paragraph that describes the I2C handling process and Figure 4.1. I2C Handling Flow. • Updated Figure 4.4. TSEMAC IP Wizard Configuration. • Updated the description for pll_lock_i signal in Table 4.2. SGMII IP Signal Description. • Updated Figure 4.5. SGMII IP Wizard Configuration. • Added the PLL IP section. • Added Figure 4.6. PLL IP Wizard Configuration. • Removed the OPERATION_SPEED parameter from Table 4.6. Top Module Parameter Description. • Updated Figure 4.13. Top Module Parameters. • Updated the file name of the testbench file to *testbench_top_sv*.
Design Architecture	Updated Figure 4.5. SGMII IP Wizard Configuration.
Reference Design Simulation	Replaced all occurrences of *ModelSim* with *QuestaSim* and updated the following figures: • Figure 7.3. Simulation Wizard: Create Simulation Project. • Figure 7.5. Simulation Wizard: Summary Page. • Figure 7.6. QuestaSim Lattice FPGA Software Interface. • Figure 7.7. QuestaSim Lattice FPGA Edition Software: Execute wave.do File. • Figure 7.8. QuestaSim Lattice FPGA Edition Software: Location of the wave.do File. • Figure 7.9. QuestaSim Lattice FPGA Edition Software: Signals Added. • Figure 7.10. QuestaSim Lattice FPGA Edition Software: Restart Simulation to Capture the Signals in the Initial Timeframe. • Figure 7.11. QuestaSim Lattice FPGA Edition Software: Passing Condition. • Figure 7.13. QuestaSim Lattice FPGA Edition Software: Quit the Simulation.
Reference Design Implementation on Board	• Updated Table 8.3. Board LED Descriptions. • Added Table 8.4. Operation Speed Setting for CertusPro-NX Devices. • Updated Figure 8.5. Front Facial LEDs Status for CrossLink-NX PCIe Bridge Board. • Added Figure 8.8. 7-Segment LED. • Added Table 8.8. 7-Segment LED Description. • Added Table 8.9. Operation Speed Setting for CrossLink-NX Devices. • Updated the External Loopback Setup section. • Updated Figure 8.27. Pattern_gen_done LED Glows on Continuous Traffic Disabled on the CrossLink-NX PCIe Bridge Board.

Revision 1.2, January 2024

Section	Change Summary
Disclaimers	Updated the disclaimer.
Implementation	Added the CrossLink device family to Table 6.1. Performance and Resource Utilization.
Reference Design Implementation on Board	• Added CrossLink-NX PCIe Bridge Board to section 8.1 Requirements. • Updated section 8.2 Device Hardware to include CrossLink-NX PCIe Bridge Board figures and table descriptions for CrossLink-NX PCIe Bridge Board LEDS and board power supply. • Updated section 8.3.2 External Loopback Setup with CrossLink-NX PCIe Bridge Board information.

Section	Change Summary
References	Updated this section to include the following references: • CrossLink-NX web page • Lattice Insights web page • Lattice Radiant Timing Constraints Methodology Application Note (FPGA-AN-02059)

Revision 1.1, July 2023

Section	Change Summary
Introduction	Changed ethernet to Ethernet.
Functional Description	Changed ethernet to Ethernet.
Design Architecture	• Updated 1000BaseX to 1000BASE-X. • Updated 802.3 IEEE to IEEE 802.3.
Reference Design Implementation on Board	Rearranged the ordering of Sl.No.2 LEDs in Table 8.3. Board LED Descriptions.
References	Added this section.

Revision 1.0, April 2023

Section	Change Summary
All	Initial release.

LATTICE SEMICONDUCTOR®

www.latticesemi.com