XAUI IP

IP Version: v1.1.0

User Guide

FPGA-IPUG-02160-1.1

December 2025

Disclaimers

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

Inclusive Language

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

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
DDR	Double Data Rate
LMMI	Lattice Memory Mapped Interface
MAC	Media Access Control
OIF	Optical Internetworking Forum
PCS	Physical Coding Sublayer
PMA	Physical Medium Attachment
PMD	Physical Medium Dependent
RTL	Register Transfer Language
SDR	Single=Data Rate
SERDES	Serializer/Deserializer
WIS	WAN Interface Sublayer
XGMII	10-Gigabit Media Independent Interface
XGXS	10Gb Ethernet Extended Sublayer
XAUI	10Gb Attachment Unit Interface

1. Introduction

1.1. Overview of the IP

The Lattice Semiconductor 10Gb Ethernet Attachment Unit Interface (XAUI) provides a solution for bridging between XAUI and 10-Gigabit Media Independent Interface (XGMII) devices. The IP core implements the 10Gb Ethernet Extended Sublayer (XGXS) capabilities in soft logic that together with PCS and serializer/deserializer (SERDES) functions implemented in the FPGA provides a complete XAUI-to-XGMII solution.

This document describes the use of the XAUI Interface IP and Lattice FPGA technology for Ethernet interface applications. The design that can be applied in multiple configurations is implemented in Verilog HDL.

1.2. Quick Facts

The table below shows a summary of the XAUI IP.

Table 1.1. XAUI IP Quick Facts

IP Requirements	Supported Devices	CertusPro™-NX, Avant™-AT-G, Avant-AT-X.
	IP Changes[1]	For a list of changes to the IP, refer to the XAUI IP Release Notes (FPGA-RN-02104).
Resource Utilization	Supported User Interface	10 Gb PHY serial to/from XGMII.
	Resources	See Table A.1.
Design Tool Support	Lattice Implementation	IP core v1.1.0 – Lattice Radiant™ software 2025.2 or later.
	Synthesis	Synopsys® Synplify Pro® for Lattice.
	Simulation	For a list of supported simulators, see the Lattice Radiant Software User Guide.

Note:

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

1.3. IP Support Summary

The following table provides IP support information on the XAUI IP core.

Table 1.2. 10G Ethernet XAUI IP Core

Device Family	Mode	Radiant Timing Model	Hardware Validated
Avant-AT-G/X	XAUI Core	Preliminary	Yes
CertusPro-NX	XAUI Core	Final	No

1.4. Features

The key features of the XAUI IP include:

- XAUI compliant functionality supported by embedded PCS implemented in the CertusPro-NX device, including four channels of 3.125 Gbps SERDES with 8b10b encoding/decoding.
- Complete 10Gb Ethernet Extended Sublayer (XGXS) solution
- TX phase FIFO and TX clock compensation FIFO
- Implements XGXS functionality conforming to IEEE 802.3-2005, including:
  - 10 GbE Media Independent Interface (XGMII).
  - XAUI compliant lane-by-lane synchronization.
  - Lane de-skew functionality.

- Supports AMBA 3 APB interface for CertusPro-NX and Avant-AT-G/X devices, and AXI4-Lite interface for Avant-AT-G/X devices.

1.5. Licensing and Ordering Information

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

1.6. Minimum Device Requirements

The minimum device requirements for the XAUI core are as follows:

Table 1.3. Minimum Device Requirements for XAUI IP Core

Devices	Speed Grades
CertusPro-NX	Speed grade 7*
Avant-AT-G/X	All speed grades

***Note:** 10G Ethernet MAC supports speed grade 9 for CertusPro-NX devices.

1.7. Conventions

1.7.1. Nomenclature

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

1.7.2. Signal Names

Signal names that end with:

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

1.7.3. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

2. Functional Descriptions

2.1. IP Architecture Overview

XAUI is a high-speed interconnect that offers reduced pin count. Each XAUI comprises four self-timed 8b10b encoded serial lanes each operating at 3.125 Gbps and thus can transfer data at an aggregate rate of 10 Gbps.

XGMII is a 156.25 MHz double data rate (DDR), parallel, short-reach interconnect interface. It supports interfacing to 10 Gbps Ethernet Media Access Control (MAC) and PHY devices.

The locations of XAUI and XGXS in the 10GbE protocol stack is shown in Figure 2.1.

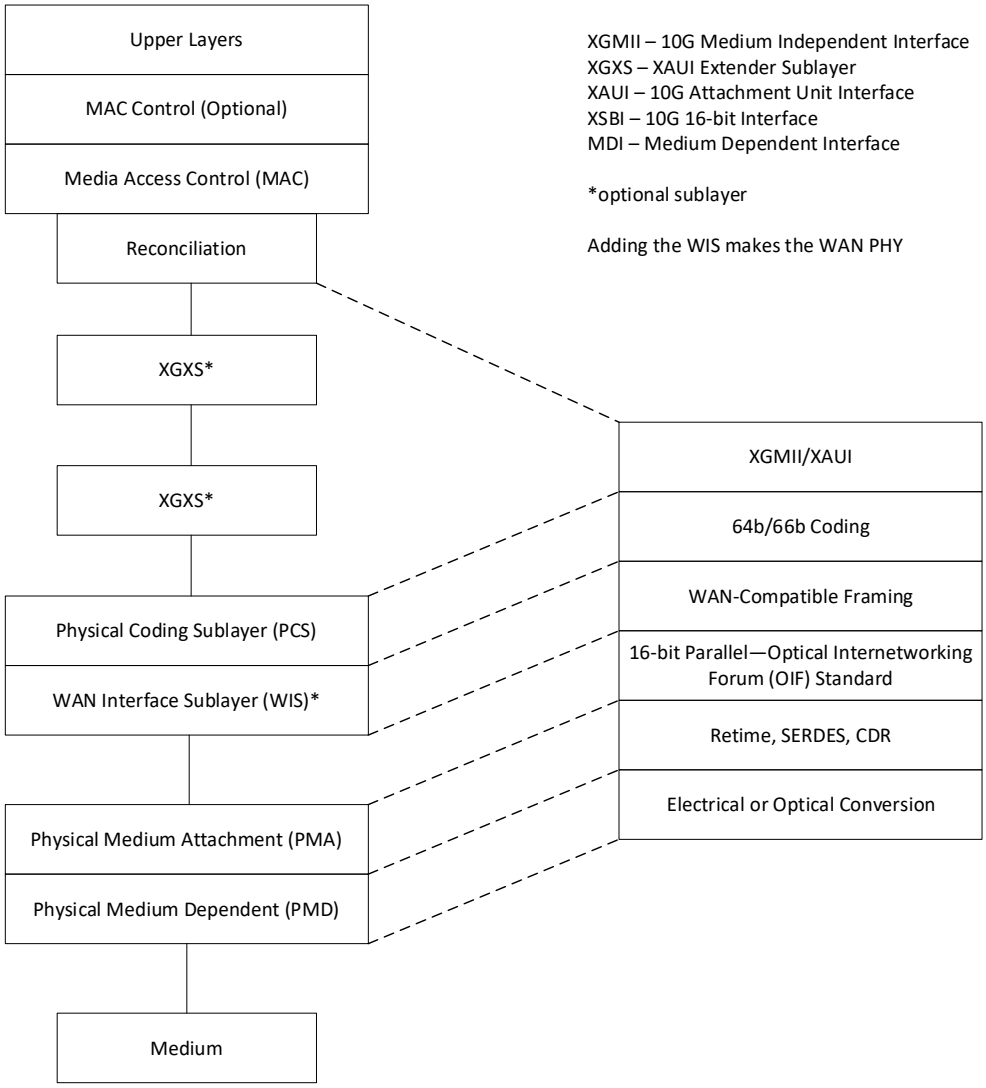

Figure 2.1. XAUI and XGXS Locations in 10 GbE Protocol Stack

The functional block diagram of the XAUI IP core is shown in Figure 2.2. The dashed lines in the figure are optional components or signals, which means they may not be available in the IP when disabled in the attribute.

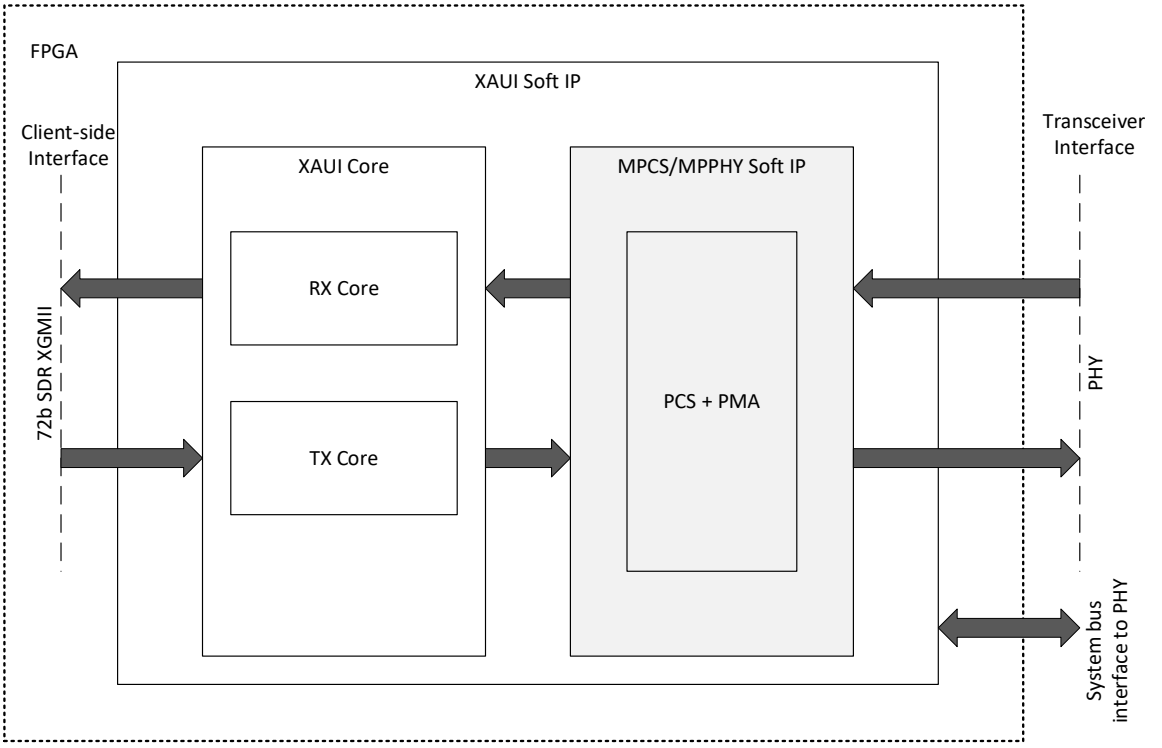

Figure 2.2. Functional Block Diagram

The XAUI IP includes either MPCS or MPPHY IP blocks configured for XAUI operation. An APB interface is provided for direct access to MPCS hard IP registers, while both APB and AXI4-Lite interfaces are available for accessing MPPHY hard IP registers.

2.2. XGMII Interface

The 10-Gigabit Media Independent Interface (XGMII) supported by XAUI IP core solution conforms to Clause 46 of IEEE 802.3-2005. The XGMII is composed of independent transmit and receive paths. Each direction uses 32-bit data signals, four (4) control signals and a clock. The 32-bit data signals in each direction are organized into four lanes of 8-bit signals each. Each lane is associated with a control signal. The XGMII supports DDR transmission, which is sampling the data and control input signals on both the rising and falling edge of the corresponding clock as shown in Figure 2.3.

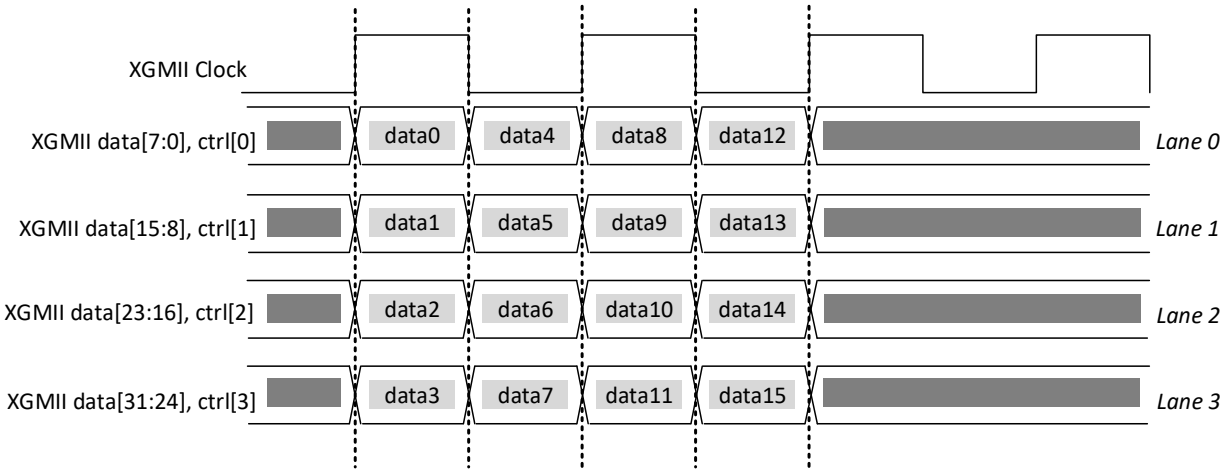

Figure 2.3. XGMII Timing Diagram

The client-side interface of the XAUI soft IP is a 64-bit single-data rate (SDR) and is based on the 32-bit XGMII-like interface. Before the XGMII data goes to the soft IP, the 32-bit wide DDR data must be de-multiplexed into 64-bits wide on a single clock edge. This de-multiplexing is done by extending the bus upwards so that there are now 8-lanes of data as shown in Figure 2.4.

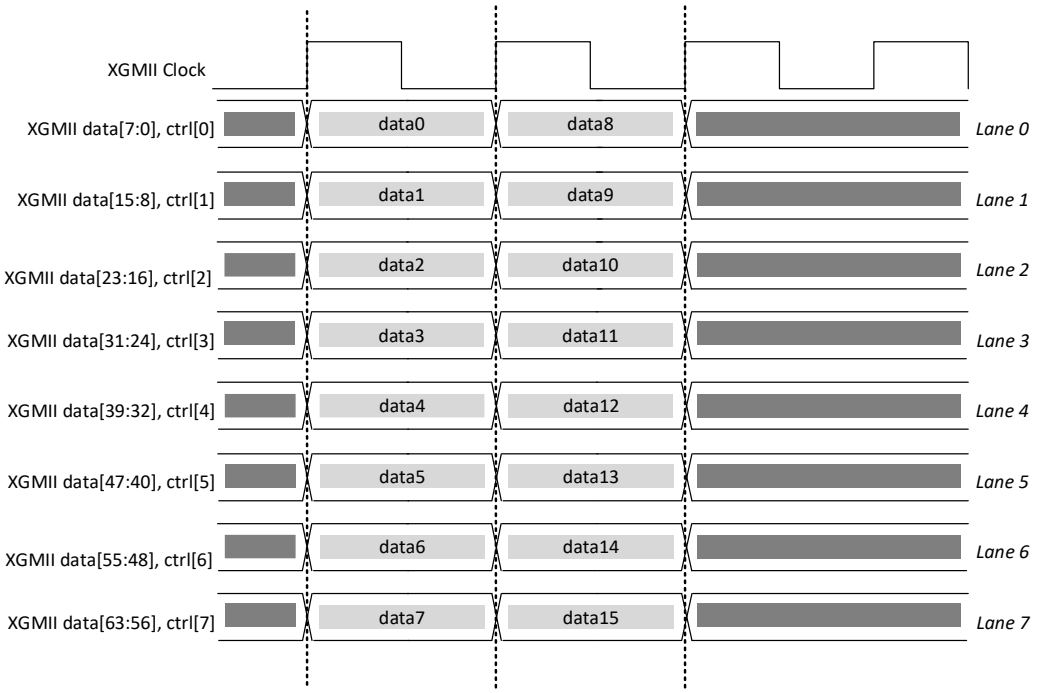

Figure 2.4. 64-bit Client-side Interface After De-multiplexing the 32-bit XGMII Data

The control signal for each lane is de-asserted when a data octet is being sent on the corresponding lane and is asserted when a control character is being sent. Supported control octet encodings are shown in Table 2.1. All data and control signals are passed directly to/from the 8b10b encoding/decoding blocks.

Table 2.1. XGMII Control Encoding

Control Signal	Data	Description	Notation
0	0x00 – 0xFF	Normal data transmission	/D/
1	0x00 – 0x06	Reserved	—
1	0x07	Idle	/I/
1	0x08 – 0x9B	Reserved	—
1	0x9C	Sequence (only valid on lane 0)	/Q/
1	0x9D – 0xFA	Reserved	—
1	0xFB	Start (only valid on lane 0)	/S/
1	0xFC	Reserved	—
1	0xFD	Terminate	/T/
1	0xFE	Error	/E/
1	0xFF	Reserved	—

2.3. XAUI-to-XGMII Translation (Receive Interface)

The block diagram for XAUI Soft IP receive path is shown in Figure 2.5. The receive interface converts the incoming XAUI stream into XGMII-compatible signals. The clock input port of clk_156_rx_i is only available for CertusPro-NX devices. Avant devices will source the clock using refclk(p/n). Refer to Table 4.1 for signals description.

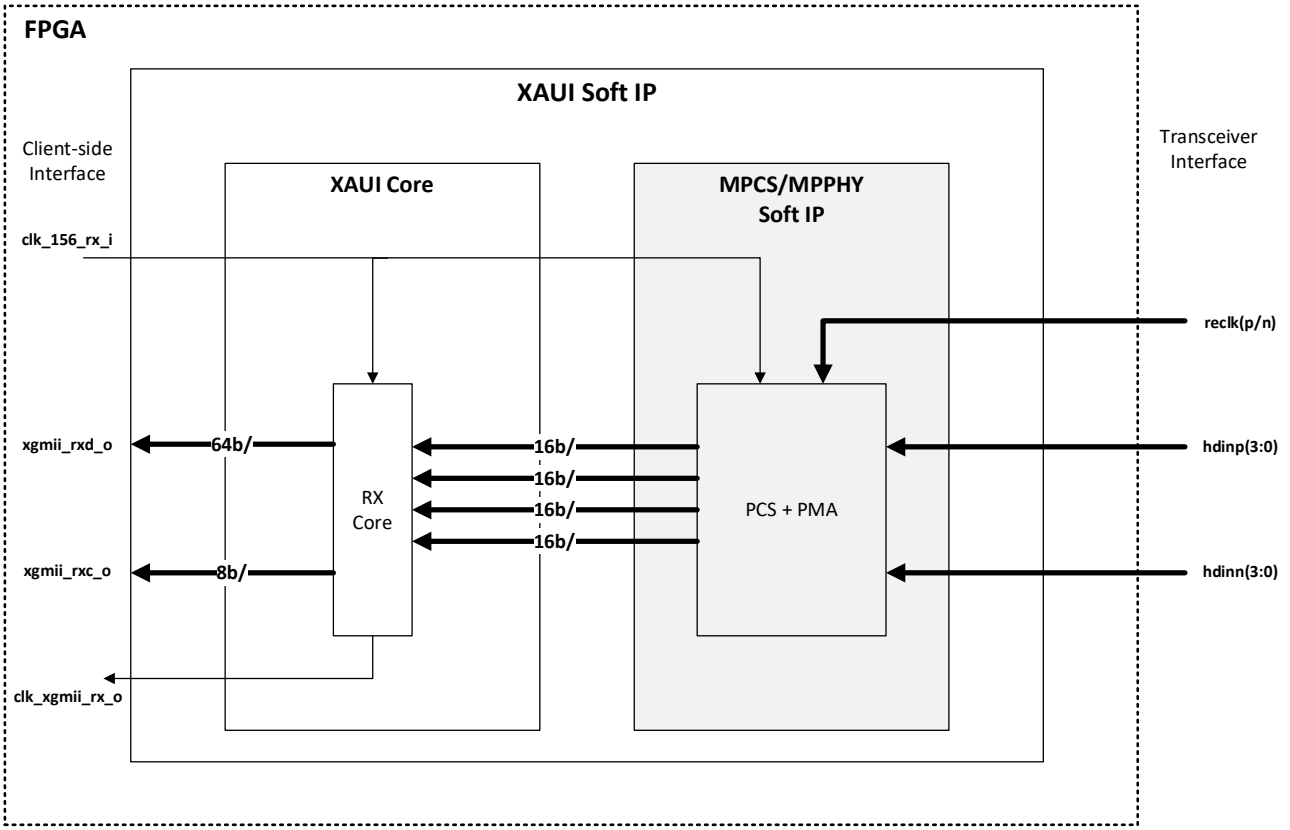

Figure 2.5. XAUI IP Receive Path

2.3.1. MPCS or MPPHY Soft IP (Receive Mode)

The MPCS soft IP is an IP that implements functionalities of PCS layer defined by SERDES protocols. The PMA and PCS settings in this IP is pre-configured for XAUI. This IP receives the serial data for each lane and performs mainly word-alignment, 8b/10b decoding, and lane-to-lane RX deskew of the incoming data.

2.3.2. XAUI RX Core

The four converted XAUI data lanes goes to the XAUI IP RX Core to de-skew, align and decode the XAUI code to its corresponding XGMII code. The XAUI IP RX core consists of the multi-channel alignment block and RX decoder block.

The multi-channel alignment block consists of four 16-byte deep FIFOs to individually buffer each XAUI lane. The multi-channel alignment block searches for the presence of the alignment character ||A|| in each XAUI lane, and begins storing the data in the FIFO when the ||A|| character is detected in a lane. When the alignment character has been detected in all four XAUI lanes, the data is retrieved from each FIFO so that all the alignment characters ||A|| are aligned across all four XAUI lanes. Once synchronization is achieved, the block does not resynchronize until a resynchronization request is issued. The data from the RX multi-channel alignment is passed to the RX decoder.

The RX decoder block converts the XAUI code to the corresponding XGMII code. Table 2.2 shows the 8b10b code points. Table 2.3 shows the code mapping between the two domains in the receive direction. XAUI ||A||, ||R||, ||K|| characters are translated into XGMII Idle (/I/) characters.

Table 2.2. XAUI 8b10b Code Points

Symbol	Name	Function	Code Group
||A||	Align	Lane alignment (XGMII idle)	K28.3
||K||	Sync	Code-Group Alignment (XGMII idle)	K28.5
||R||	Skip	Clock Tolerance Compensation (XGMII idle)	K28.0
||S||	Start	Start of Packet Delimiter (in lane 0 only)	K27.7
||T||	Terminate	End of Packet Delimiter	K29.7

Symbol	Name	Function	Code Group
||E||	Error	Error Propagation	K30.7
||Q||	Sequence	Link Status Message Indicator	K28.4
/d/	Data	Information Bytes	Dxx.x

Table 2.3. XAUI 8b10b to XGMII Code Mapping

8b10b Data	XGMII Hex	XGMII Name	XGMII Control	Usage Context /When received
K28.5 (0xBC)	0x07	/I/ (Idle)	1	Part of inter-packet gap fill, ||A||, ||R|| or ||K||
K28.3 (0x7C)	0x07	/I/ (Idle)	1	Part of inter-packet gap fill, ||A||, ||R|| or ||K||
K28.0 (0x1C)	0x07	/I/ (Idle)	1	Part of inter-packet gap fill, ||A||, ||R|| or ||K||
K27.7 (0xFB)	0xFB	/S/ (Start)	1	Begins frame parsing
K29.7 (0xFD)	0xFD	/T/ (Terminate)	1	Ends frame parsing
K30.7 (0xFE)	0xFE	/E/ (Error)	1	Flags error
K28.4 (0x9C)	0x9C	/Q/ (Order Set)	1	Interprets ordered set
Dxx.x	0x00-0xFF	/d/ (data)	0	Frame payload

2.3.3. Timing of Data Frame Transfer to Client-side Interface

The timing of a normal incoming transfer to the Client-side is shown in Figure 2.6. The frame is enclosed by a Start character (S) and by a Terminate character (T). The Start character in this implementation can occur in either lane 0 or lane 4. The Terminate character, T, can occur in any lane.

Figure 2.7 shows a frame of data propagating an error. In this example, the error is propagated in lanes 4 to 7, shown by letter E.

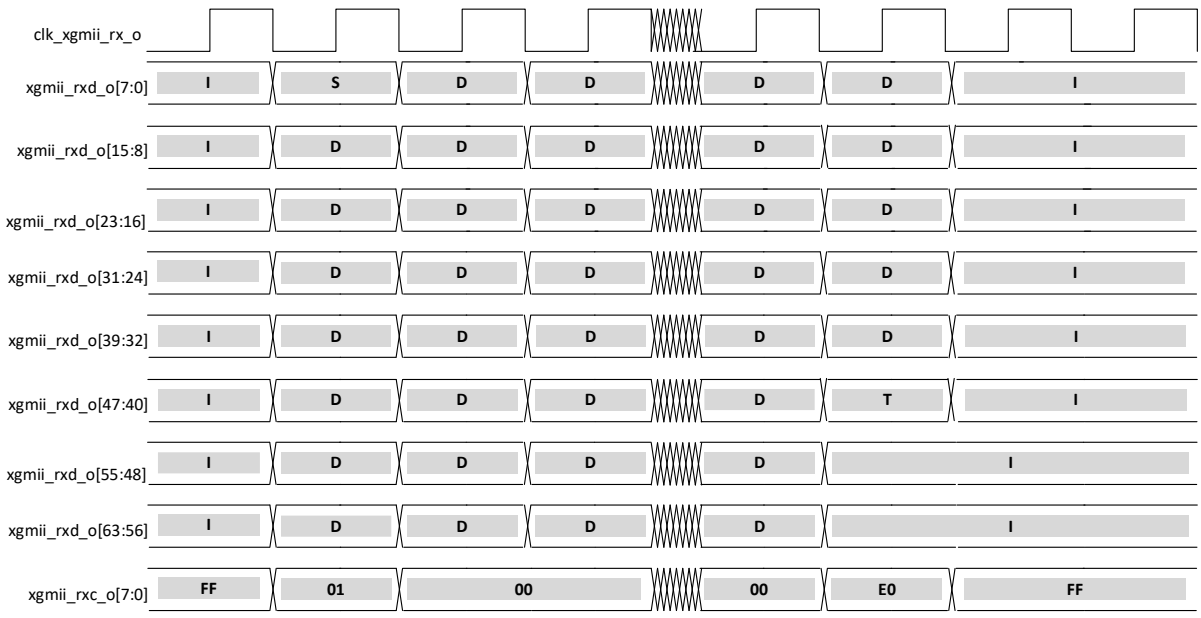

Figure 2.6. Timing of Data Frame Transfer to Client-side Interface

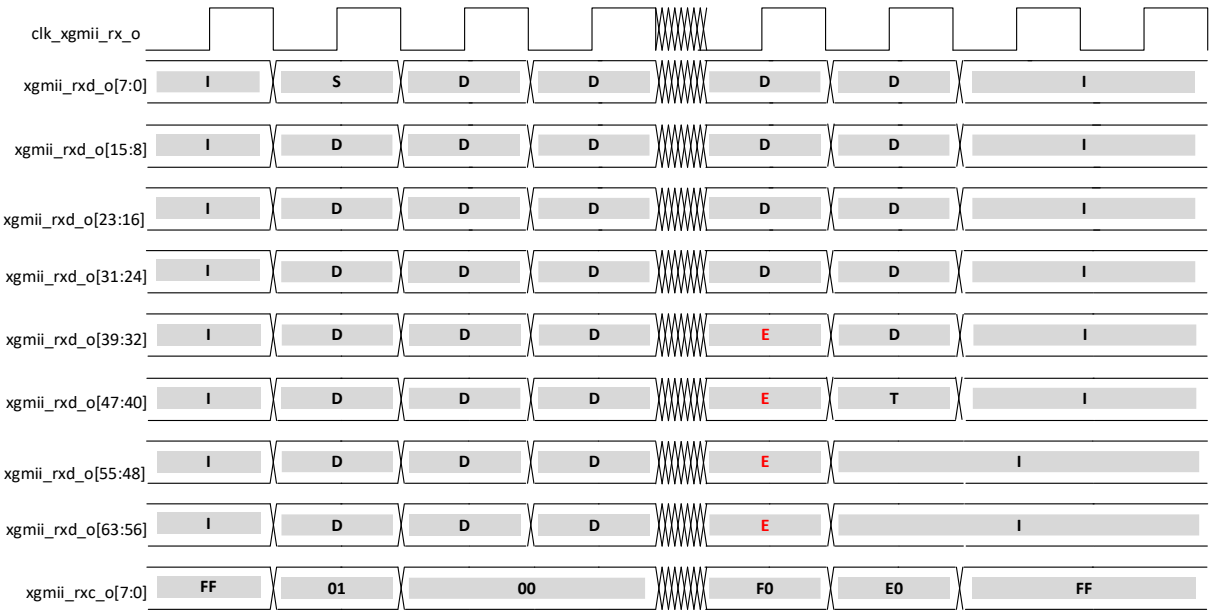

Figure 2.7. Timing of Data Frame Transfer with Error to Client-side Interface

2.4. XGMII to XAUI Translation (Transmit Interface)

The block diagram of the XAUI Soft IP transmit data path is shown in Figure 2.8. The TX interface converts the incoming XGMII data into XAUI-compatible characters. Clock input port of clk_156_tx_i is only available for CertusPro-NX devices. The Avant device sources its clock from refclk(p/n). Refer to Table 4.1 for signals description.

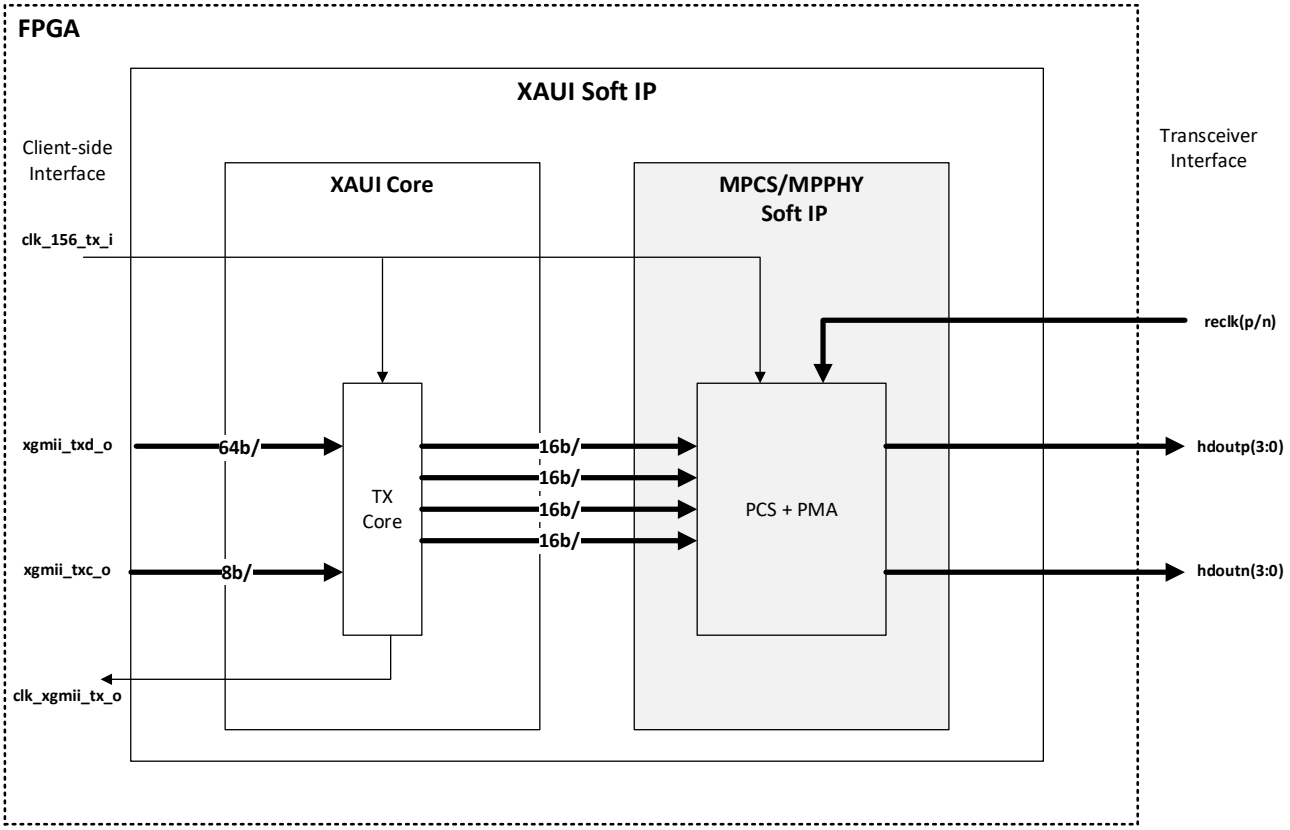

Figure 2.8. XAUI IP Transmit Path

2.4.1. XAUI TX Core

After the Slip buffer, the XGMII transmit data and control goes to the TX Encoder to encode XGMII characters into 8b10b format shown in Table 2.4. The idle generation state machine in the TX encoder converts XGMII /I/ characters to a random sequence of XAUI ||A||, ||K|| and ||R|| characters as specified in IEEE 802.3-2005. XGMII idles are mapped to a random sequence of code groups to reduce radiated emissions. The ||A|| code groups support XAUI lane alignment and have a guaranteed minimum spacing of 16 code-groups. The ||R|| code groups are used for clock compensation. The ||K|| code groups contain the 8b10b comma sequence.

Table 2.4. XGMII to XAUI Code Mapping

XGMII Control	XGMII Hex	XGMII Name	8b/10b Code on XAUI	Usage Context / When transmitted
1	0x07	/I/ (Idle)	Randomized ||A||, ||R||, ||K|| Sequence ||A|| = K28.3 = 0x7C ||R|| = K28.0 = 0x1C ||K|| = K28.5 = 0xBC (Comma)	Inter-frame gap, idle periods. Encoded as ||A||, ||R||, or ||K|| ordered sets.
1	0xFB	/S/ (Start)	||S|| = 0xFB	Marks the start of frame
1	0xFE	/E/ (Error)	||E|| = 0xFE	Signals error condition
1	0xFD	/T/ (Terminate)	||T|| = 0xFD	Marks the end of frame
1	0x9C	/Q/ (Ordered set)	||Q|| = 0x9C	Used for fault signaling
0	0x00-0xFF	(Data)	0x00-0xFF (Data)	Frame payload

The random ||A||, ||R||, ||K|| sequence is generated as specified in section 48.2.4.2 of IEEE 802.3-2005. In addition to idle generation, the state machine also forwards sequences of ||Q|| ordered sets used for link status reporting. These sets have the XGMII sequence control character on lane 0 followed by three data characters in XGMII lanes 1 through 3. Sequence ordered-sets are only sent following an ||A|| ordered set.

The random selection of ||A||, ||K||, and ||R|| characters is based on the generation of uniformly distributed random integers derived from a PRBS. Minimum ||A|| code group spacing is determined by the integer value generated by the PRBS. ||K|| and ||R|| selection is driven by the value of the least significant bit of the generated integer value. The idle generation state machine specified in IEEE 802.3-2005 transitions between states based on a 312 MHz system clock. The TX encoder implemented in the XAUI TX core runs at a system clock rate of 156.25 MHz. Therefore, the XGXS state machine implementation performs the equivalent of two state transitions each clock cycle.

2.4.2. MPCS or MPPHY Soft IP (Transmit Mode)

This block receives 20 bits of data of each XAUI lane. This block serializes the incoming XAUI data.

For more information on CertusPro-NX MPCS, refer to the MPCS Module User Guide (FPGA-IPUG-02118).

For more information on CertusPro-NX SERDES, refer to the CertusPro-NX SERDES/PCS User Guide (FPGA-TN-02245).

For more information on Avant G/X MPPHY, refer to the Lattice Avant-G/X MPPHY Module User Guide (FPGA-IPUG-02233).

For more information on Avant G/X SERDES, refer to the Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).

2.4.3. Timing of Data Frame Transfer from Client-side Interface

The timing of a normal incoming transfer from the Client-side is shown in Figure 2.9. The beginning of the data frame is indicated by the presence of the Start character followed by data characters in lanes 5, 6, and 7. Alternatively, the start of the data frame can be marked by the occurrence of a Start character in lane 0, with the data characters in lanes 1 to 7. When the frame is complete, a Terminate character completes it. The Terminate character can occur in any lane. The remaining lanes are padded by XGMII idle characters.

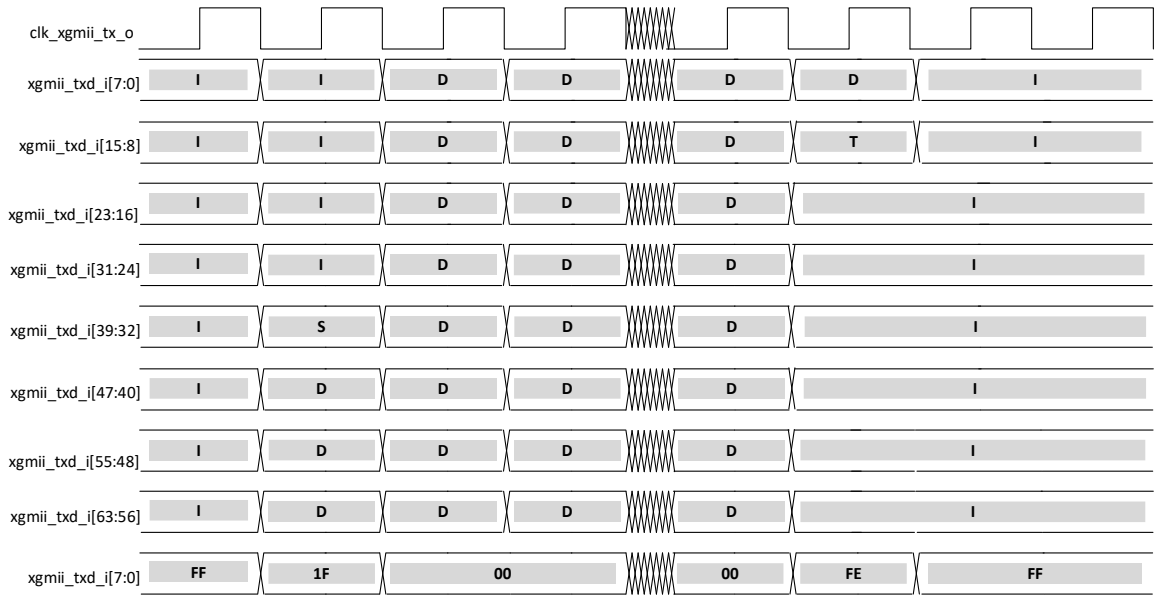

Figure 2.9. Timing of Data Frame Transmission from Client-side Interface

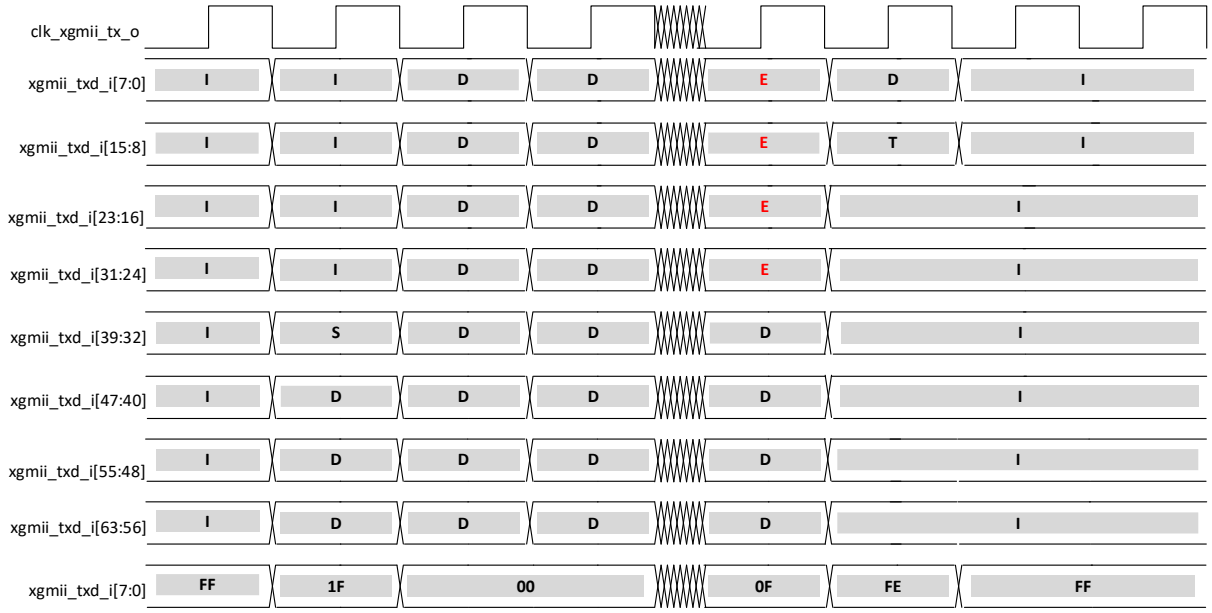

Figure 2.10. Timing of Data Frame Transmission with Error from Client-side Interface

Figure 2.10 depicts a similar frame to that of Figure 2.9 but with error in the frame. The letter E denotes the error, with the relevant control bit set.

3. IP Parameter Description

The configurable attributes of the XAUI IP core are shown in the table below. The attributes can be configured through the IP Catalog's Module/IP wizard of the Lattice Radiant software.

Table 3.1. General Attributes

Attribute	Selectable Values	Default	Description
Enable Control signals	Checked, Unchecked	Unchecked	This option allows you to control the following control signals: • mca_resync_i • mca_auto_resync_i • mca_sync_status_o

Host Interface	• For CertusPro-NX
	  • APB
	• For Avant G/X
	  • AXI4-Lite
	  • APB
		• For CertusPro-NX
		  • APB
		• Avant G/X
		  • AXI4-Lite
			System bus interface for set registers.

PCS Lane ID	• For CertusPro-NX
	  • Auto
	  • 0
	  • 4
	• For Avant G/X
	  • Auto
	  • 0
	  • 4
	  • 8
	  • 12
	  • 16
	  • 20
	  • 24
		• For CertusPro-NX
		  • Auto
		• Avant G/X
		  • Auto
			Specifies lane ID for PCS. You may select desired quad based on the device. For more information on lane ID that is mapped to the SERDES, refer to the following user guides:
			• CertusPro-NX SERDES/PCS User Guide (FPGA-TN-02245).
			• Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313).

4. Signal Description

Table 4.1. XAUI IP Core Signal Description

Port Name	I/O	Width	Description
System			
clk_xgmii_tx_o	out	1	156.25 MHz clock from XAUI IP core to the XGMII TX. Use this clock after ready_o is asserted, as there may be clock instability before the XAUI core is ready.
clk_xgmii_rx_o	out	1	156.25 MHz clock from XAUI IP core to the XGMII RX. Use this clock after ready_o is asserted, as there may be clock instability before the XAUI core is ready.
clk_156_rx_i[1]	in	1	156.25 MHz XGMII RX clock. This is also used to clock the data coming from the MPCS.
clk_156_tx_i[1]	in	1	156.25 MHz XGMII TX clock.
sysbus_clk_i	in	1	Clock for management module (APB and AXI4-Lite interface). It is recommended to use 100 MHz clock.
sysbus_clk_i	in	1	Clock for management module (APB and AXI4-Lite interface). It is recommended to use 100 MHz to 150 MHz clock.
reset_n_i	in	1	Active-low asynchronous reset.
ready_o	out	1	Indicates that XAUI core is ready. MAC IP reset must be sequential to this ready_o, and must only be asserted after XAUI core is ready.
Transceiver Interface			
refclkp_i	in	1	Differential transceiver clock pair or 156.25 MHz input clock.
refclkn_i	in	1	
hdinp0_i	in	1	Differential receive input pair lane 0.
hdinn0_i	in	1	
hdinp1_i	in	1	Differential receive input pair lane 1.
hdinn1_i	in	1	
hdinp2_i	in	1	Differential receive input pair lane 2.
hdinn2_i	in	1	
hdinp3_i	in	1	Differential receive input pair lane 3.
hdinn3_i	in	1	
hdoutp0_o	out	1	Differential transmit output pair lane 0.
hdoutn0_o	out	1	
hdoutp1_o	out	1	Differential transmit output pair lane 1.
hdoutn1_o	out	1	
hdoutp2_o	out	1	Differential transmit output pair lane 2.
hdoutn2_o	out	1	
hdoutp3_o	out	1	Differential transmit output pair lane 3.
hdoutn3_o	out	1	
Client-side Interface			
xgmii_txd_i	in	64	Transmit data, 8 bytes wide; Data is from XGMII TX going to XAUI.
xgmii_txc_i	in	8	Transmit control bits, 1-bit per transmit data byte.
xgmii_rxd_o	out	64	Receive data, 8 bytes wide; Data goes to XGMII RX from XAUI.
xgmii_rxc_o	out	8	Receive control bits, 1-bit per receive data byte.
APB Interface			
apb_psel_i	in	1	Select signal. Indicates that the completer device is selected and a data transfer is required.
apb_paddr_i	in	32	Address signal.
apb_pwdata_i	in	32	Write data signal.

Port Name	I/O	Width	Description
apb_pwrite_i	in	1	Direction signal. Write = 1, Read = 0.
apb_penable_i	in	1	Enable signal. Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	out	1	Ready signal. Indicates transfer completion. The completer uses this signal to extend an APB transfer.
apb_prdata_o	out	32	Read data signal.
apb_pslverr_o	out	1	APB error signal. Not supported and just tied to 0.
AXI4-Lite Interface (For Avant G/X only)			
axi4l_awaddr_i	in	32	Write address bus.
axi4l_awvalid_i	in	1	Write address valid.
axi4l_awready_o	out	1	Write address acknowledge.
axi4l_wdata_i	in	32	Write data bus.
axi4l_wvalid_i	in	1	Write data valid.
axi4l_wready_o	out	1	Write data acknowledge.
axi4l_wstrb_i	in	4	AXI4-Lite write strobe. This feature is not supported so tied to 0.
axi4l_bresp_o	out	2	Write transaction response. This feature is not supported.
axi4l_bvalid_o	out	1	Write response valid.
axi4l_bready_i	in	1	Write response acknowledge.
axi4l_araddr_i	in	32	Read address bug.
axi4l_arvalid_i	in	1	Read address valid.
axi4l_arready_o	out	1	Read address acknowledge.
axi4l_rdata_o	out	32	Read data output.
axi4l_rresp_o	out	2	Read data response. This feature is not supported.
axi4l_rvalid_o	out	1	Read data/response valid.
axi4l_rready_i	in	1	Read data acknowledge.
Others[2]			
mca_resync_i	in	1	Multi-channel alignment resynchronization request. This signal is in the clk_156_rx_i clock domain. Hold the input signal for at least 32 ns (5 clock cycles of clk_156_rx_i) for this to take effect. When not enabled in the GUI, this is default to 0.
mca_auto_resync_i	in	1	Multi-channel alignment auto resynchronization request. This is in the clk_156_rx_i clock domain. When not enabled in the GUI, this is default to 1.
mca_sync_status_o	out	1	XAUI multi-channel alignment status. 1 = All channels are aligned; 0 = XAUI channels are not aligned.

Notes:

1. Clock input ports for CertusPro-NX devices only. This port is not available in Avant-AT-G/X devices.
2. Available only when *Enable Control signals* is checked. This is a debug option for you to manually drive resync input to XAUI core. Auto MCA is defaulted to ON, when Enable Control signal is unchecked.

5. Register Description

5.1. APB Interface Registers

The Avant-AT-G/X and CertusPro-NX PCS Hard IP registers can be accessed directly through the APB interface. Refer to the Lattice Avant G/X MPPHY Module User Guide (FPGA-IPUG-02233) for MPPHY hard IP address mapping and the MPCS Module User Guide (FPGA-IPUG-02118) for MPCS hard IP address mapping.

5.2. AXI4-Lite Interface Registers

The Avant-AT-G/X PCS Hard IP registers can be accessed directly through AXI4-Lite interface. Refer to the Lattice Avant G/X MPPHY Module User Guide (FPGA-IPUG-02233) for MPPHY hard IP address mapping.

5.3. PHY Management Block

The PHY Management block is accessed through the AXI4-Lite or APB interface. This block is responsible for register access to the PCS registers.

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

Table 5.1. AXI4-Lite to PCS Address and Data Conversion

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

For details on how to configure PCS module registers addresses using APB and AXI4-Lite interfaces, refer to the *PHY Management Block* section of the 10G Ethernet IP User Guide (FPGA-IPUG-02245).

Note: reg 90, 91, and A4 registers for the MPCS Module are strictly for 10BASE-R usage only. You must not perform read or write operations when you are using XAUI protocol.

6. Example Design

The XAUI IP example design allows you to compile, simulate, and test the 10G Ethernet MAC together with the XAUI IP on Avant G/X 70 Versa board.

Note: You must have 10G Ethernet IP license to run this example design.

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

6.1. Example Design Configuration

This section provides information on how to use the XAUI IP along with the 10G Ethernet MAC only option.

Table 6.1. XAUI IP Configuration Supported by the Example Design

XAUI IP GUI Parameter	Example Design Selected Value
Enable Control signals	Unchecked
Host Interface	AXI4-lite
PCS Lane ID	12

6.2. Overview of XAUI IP Example Design and Features

The XAUI example design include traffic generator and checker to generate and pause traffics checking in loopback.

6.3. XAUI Example Design Components

The following figure shows the block diagram of the example design.

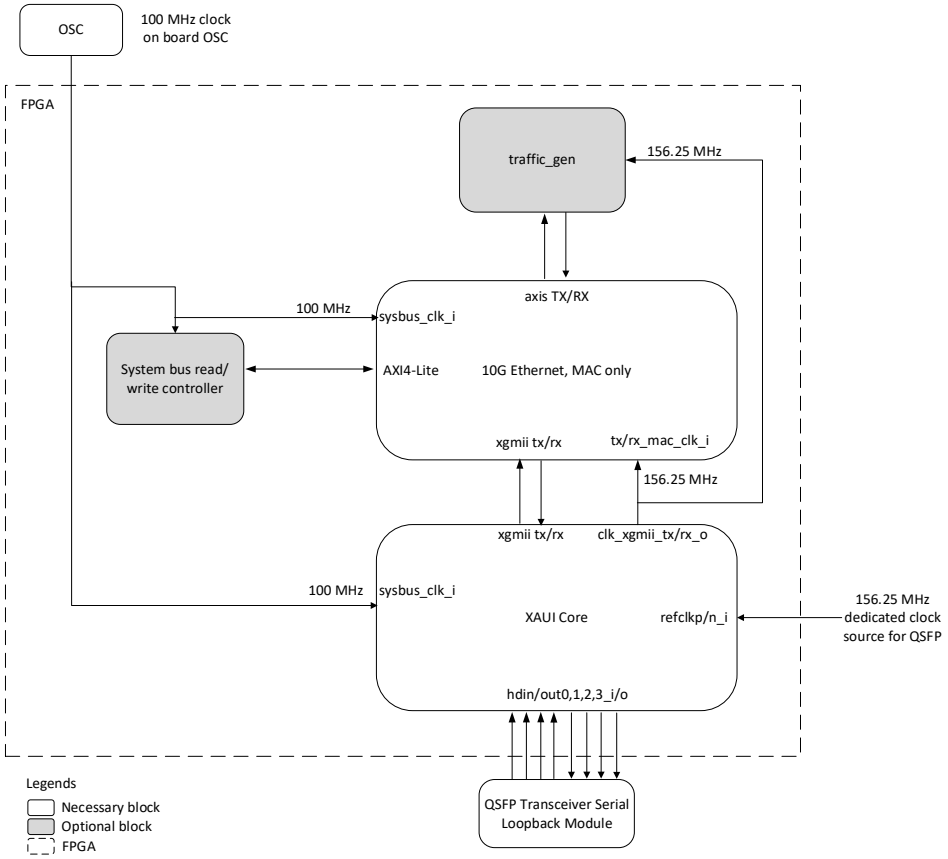

Figure 6.1. XAUI Example Design Block Diagram for Avant X70 Devices

The XAUI example design includes the following blocks:

- XAUI Core
- 10G Ethernet, MAC only
- OSC
- System bus read/write controller
- Traffic generator

6.3.1. XAUI Core

The XAUI core IP contains all the necessary logic, interfacing, and clocking infrastructure to integrate 10G Ethernet MAC and Ethernet networks.

6.3.2. 10G Ethernet—MAC Only

Industry standard 10G Ethernet MAC only interface.

6.3.3. OSC

Onboard oscillator module with 100 MHz clock output.

6.3.4. System Bus Read/Write Controller

The controller to configure 10G Ethernet MAC registers using the AXI4-Lite interface.

6.3.5. Traffic Generator

The traffic generator block module is used to generate traffic for transmission using AXI4-Stream transmit and receive interface. This module has data checker included, which performs checking on the data received against the data transmitted.

6.4. Generating Example Design

6.4.1. Creating a New Radiant Project

To create a new project, follow these steps:

1. In the Latice Radiant software, go **to File-> New -> Projec**t… or click on the **New Project** icon.

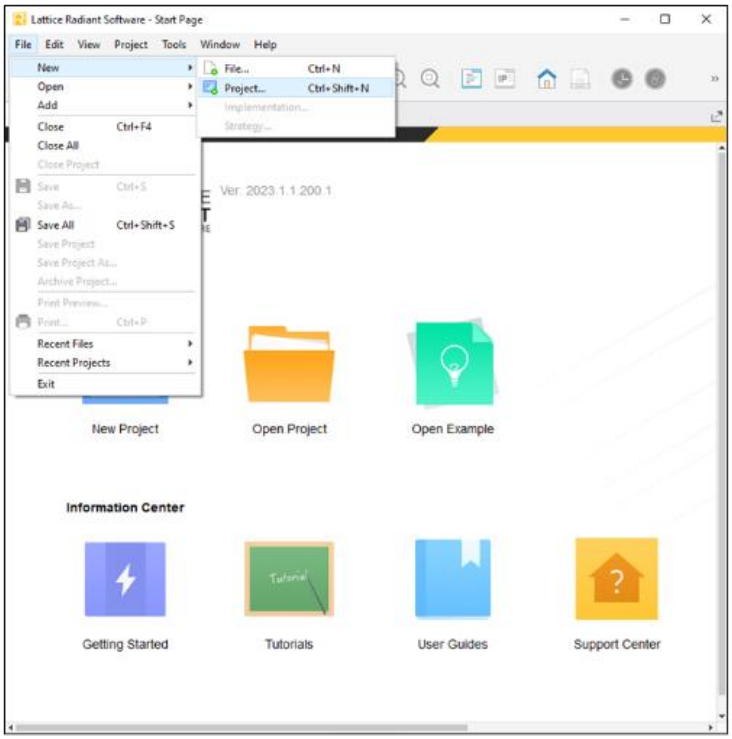

Figure 6.2. Project Creation

2. Specify **Project Name**, **Location**, **Implementation Name**, and click **Next**.

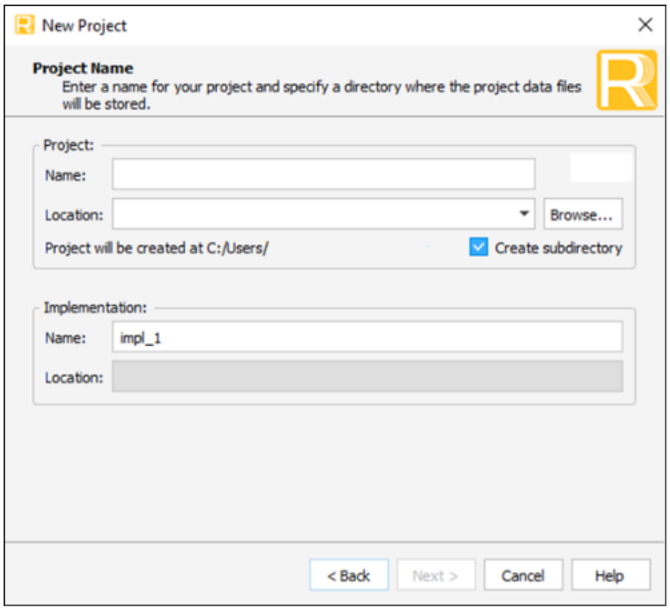

Figure 6.3. Project Name and Location

3. Specify a device for the project by selecting a device family, an operating condition, the package LFG1156, performance grade 3, and part number LAV-AT-X70-3LFG1156I.

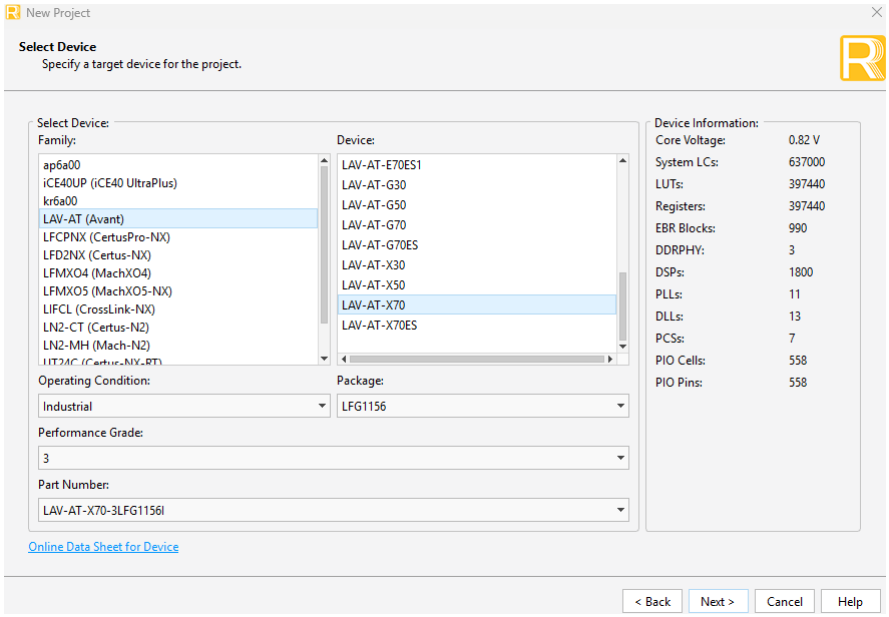

Figure 6.4. Device Selection

4. In the **Select Synthesis Tool** window, select **Synplify Pro** and click **Next**.

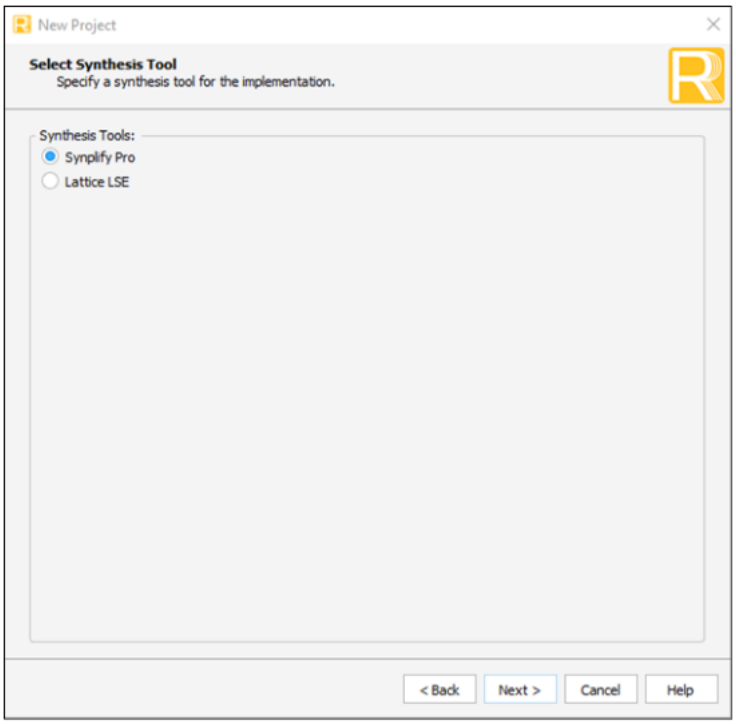

Figure 6.5. Synthesis Tool Selection

5. View and verify the project information, and click **Finish**.

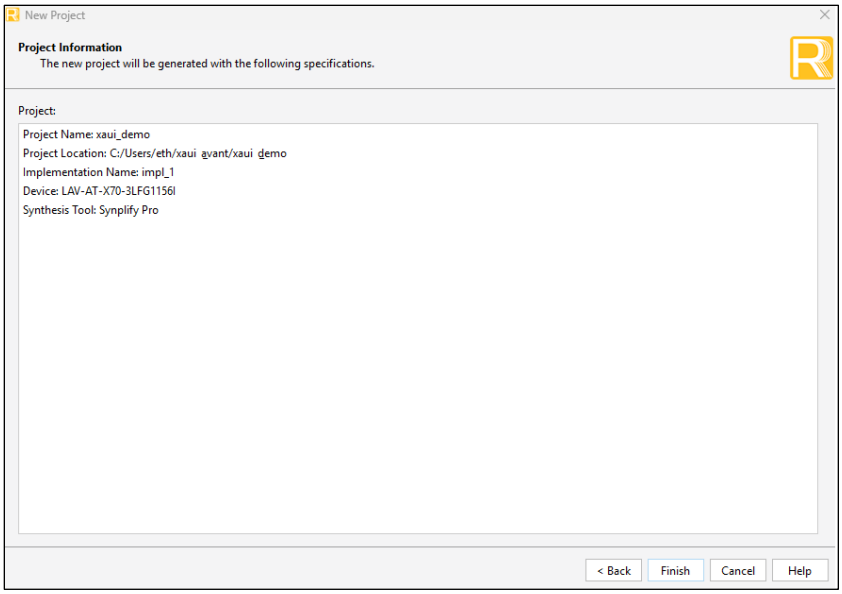

Figure 6.6. Project Information

6.4.2. IP Installation and Generation

To install and generate the IP, follow these steps:

1. Go to **IP Catalog -> IP on Server**, and download XAUI.
2. Go to the **IP on Local** tab, right-click on the XAUI IP and select **Generate…** to launch the IP GUI.

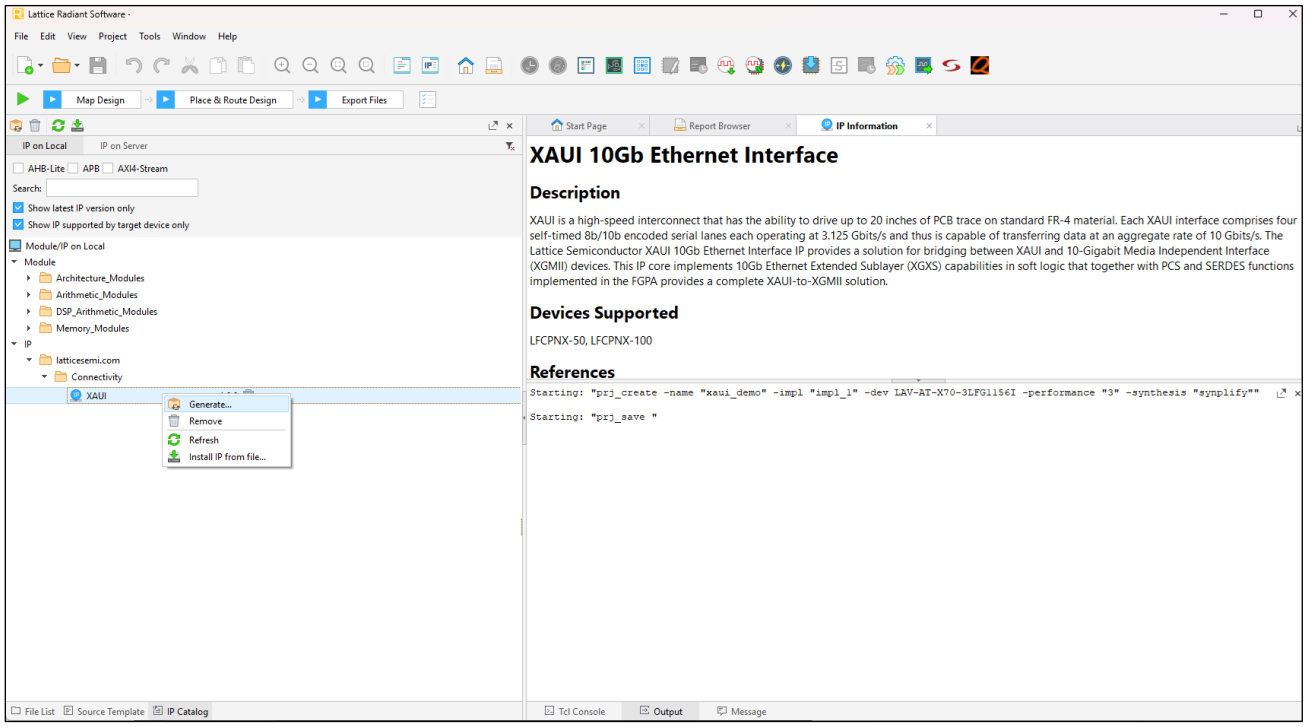

Figure 6.7. IP Generation

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

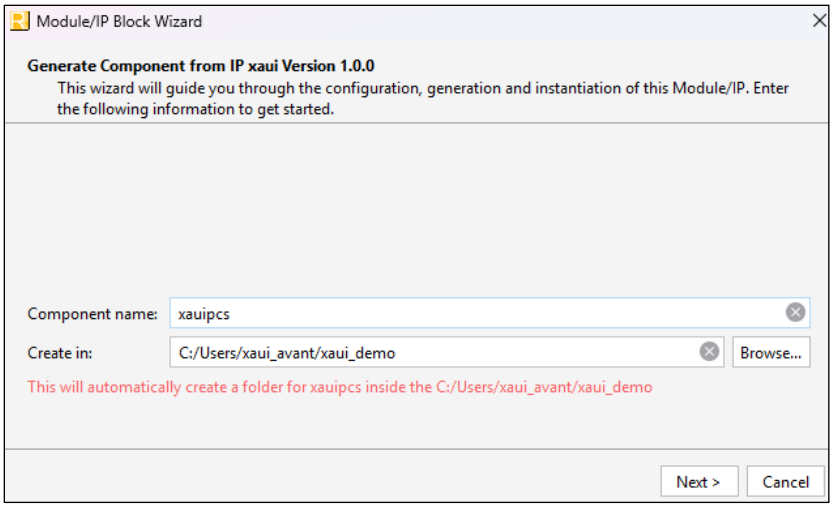

Figure 6.8. Component Name and Folder Creation

4. The following window allows you to change the IP configuration. Follow the settings shown in Table 6.1 and click **Generate**.

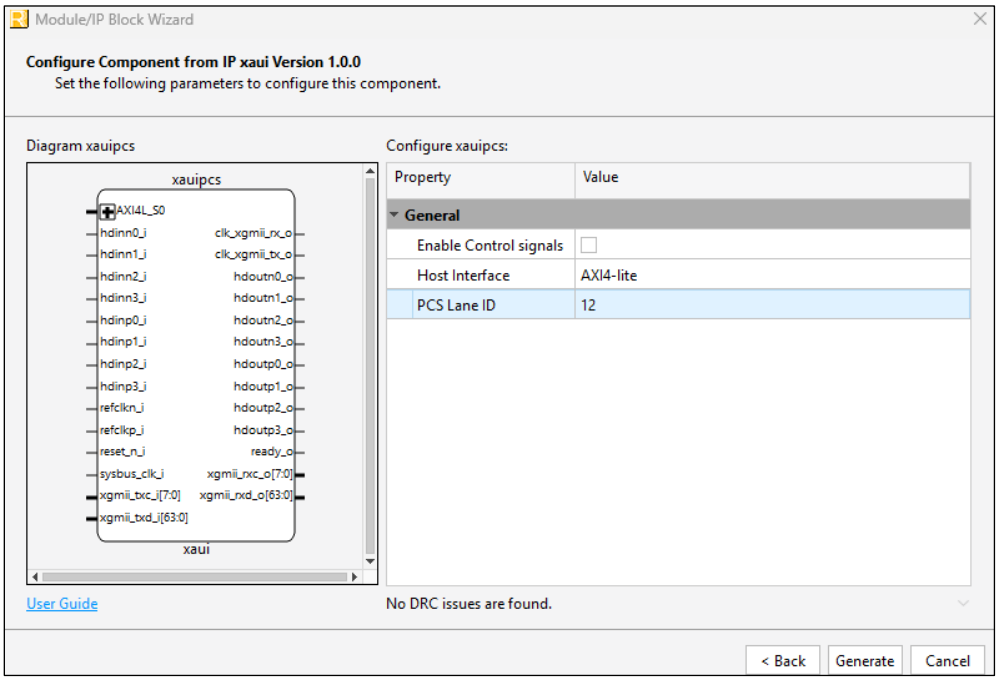

Figure 6.9. IP Parameter Settings

6.4.3. Importing Example Design Files to a Project

To add an existing example design into project, follow these steps:

1. Go to **Add->Existing File**... and select an existing example design using the Radiant software graphical user interface.
2. Import the top-level file, *top.v*, *tb_top.v*, and the *avant_xaui.pdc* files from the <IP_INSTANCE_NAME>\eval\versa_top\avant_x70_xaui folder.

   Alternatively, you may set up the example design using TCL scripts by entering the command below.
   TCL command: source {<IP_ISNTANCE_NAME>\eval\versa_top\avant_x70_xaui\ed_setup.tcl}
3. Replace <IP_INSTANCE_NAME> with the actual IP instance name.

Using the TCL script is an alternate way to set up the example design files, including setting the simulation and post-synthesis file. If TCL script is used, you do not have to perform add files using the graphical user interface.

The figure below shows how to add files using the Radiant software graphical user interface.

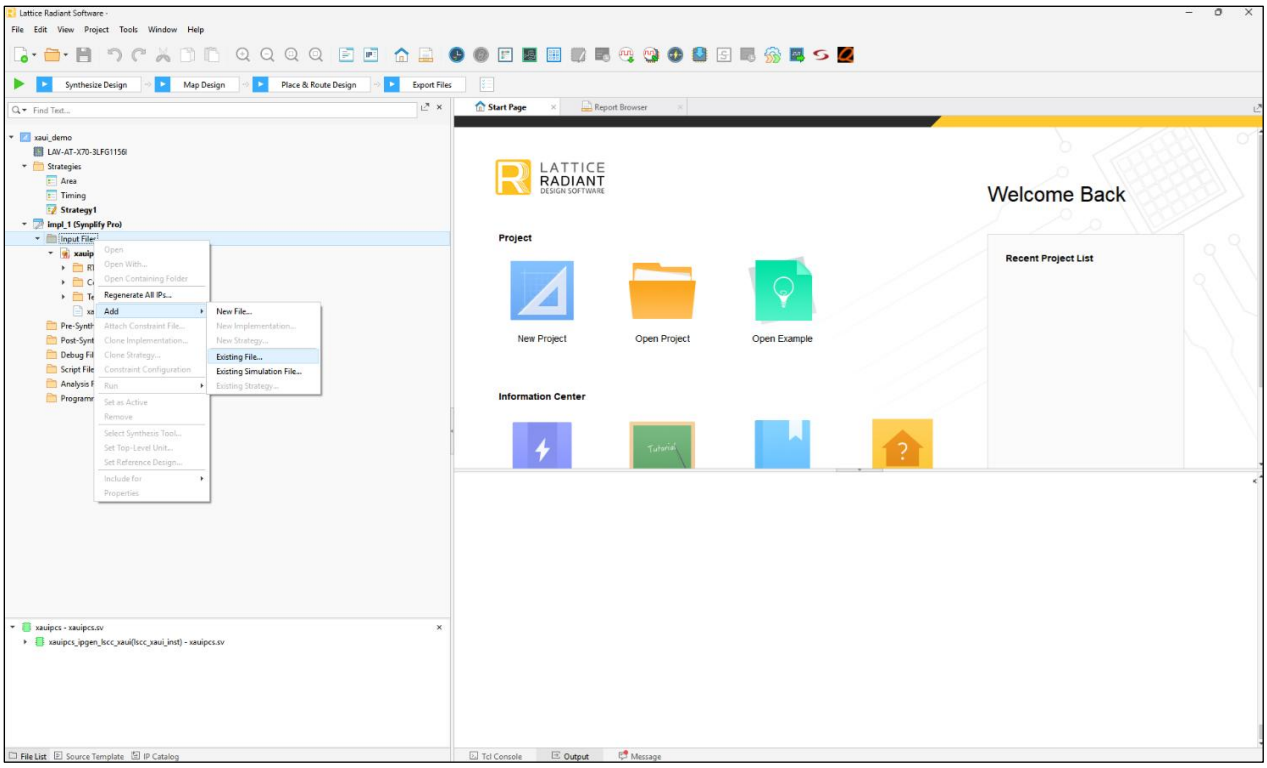

Figure 6.10. Add an Existing File Using the GUI

The figure below shows an example on how to run TCL using the TCL Console located at the bottom panel of the Radiant software with <IP_INSTANCE_NAME> of xauipcs.

You must instantiate 10G Ethernet MAC only from the IP Catalog, and pair it with the XAUI IP core hardware example design. The 10G Ethernet MAC must be instantiated with the MAC only IP option using AXI4-Lite host interface and XGMII PHY interface with instance name *mac10g*. This module name is called inside *top.v* file of the XAUI example design.

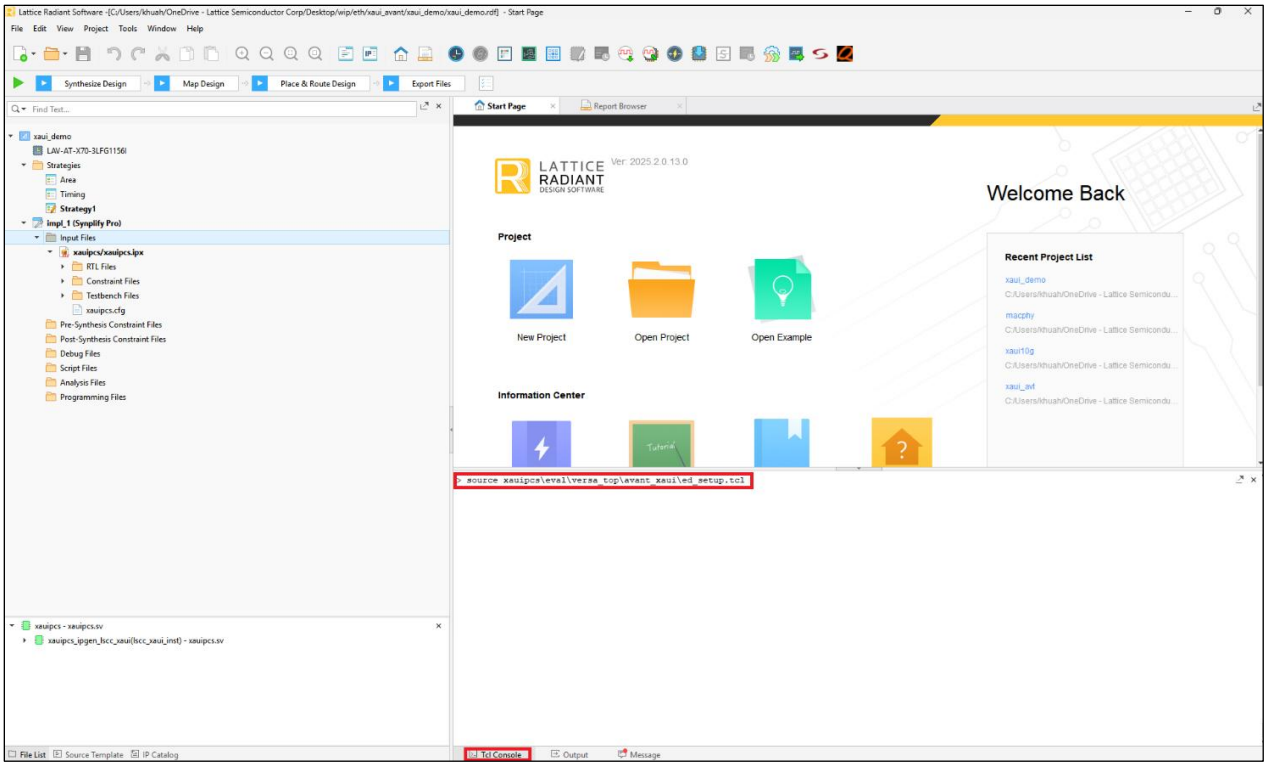

Figure 6.11. Set Up Example Design Using TCL

4. Click **Run All** to compile the bitstream file from design. Do not include any tb_* files for synthesis.
5. To program the bitstream, go to **Tools->Programmer** to launch the Radiant Programmer.

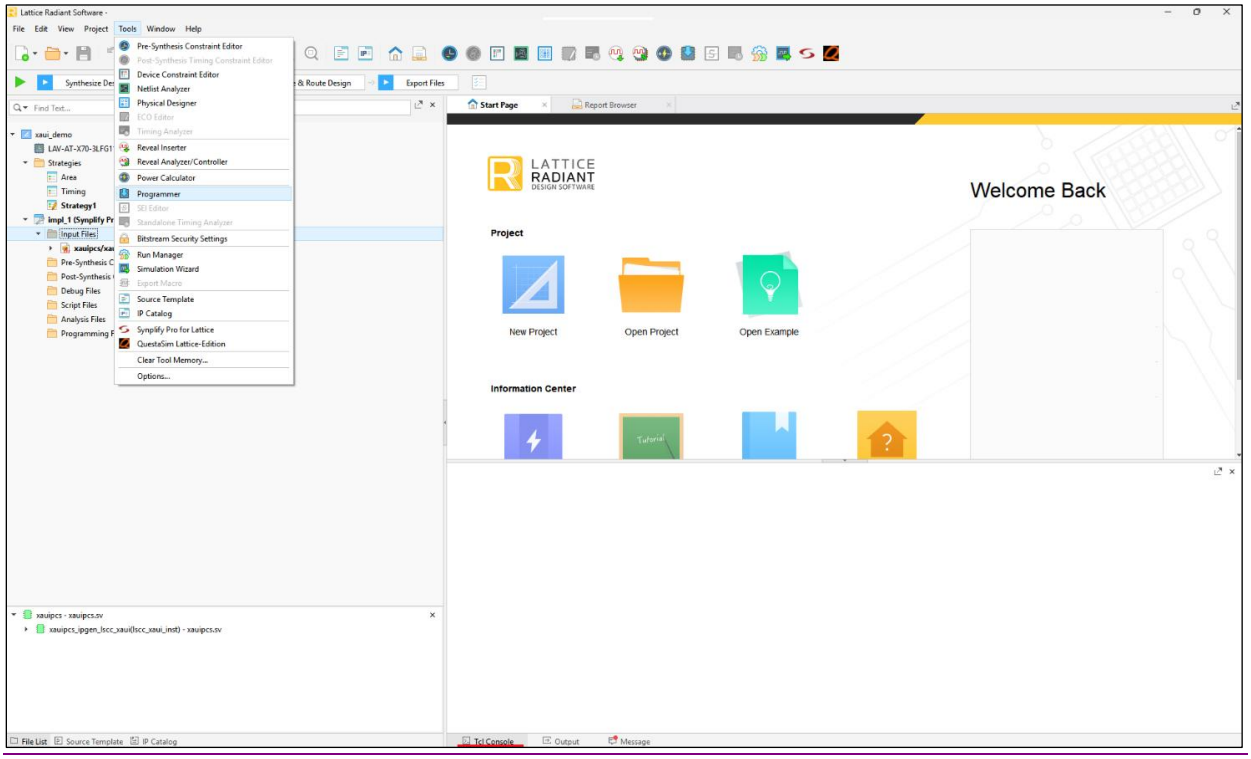

Figure 6.12. Programmer

For more information on the Radiant Programmer, refer to the Programming section of the Avant-G/X Versa Board User Guide (FPGA-EB-02063).

6.5. Hardware Testing

The XAUI IP is hardware tested on the Avant G/X70 Versa Board and QSFP Loopback Module.

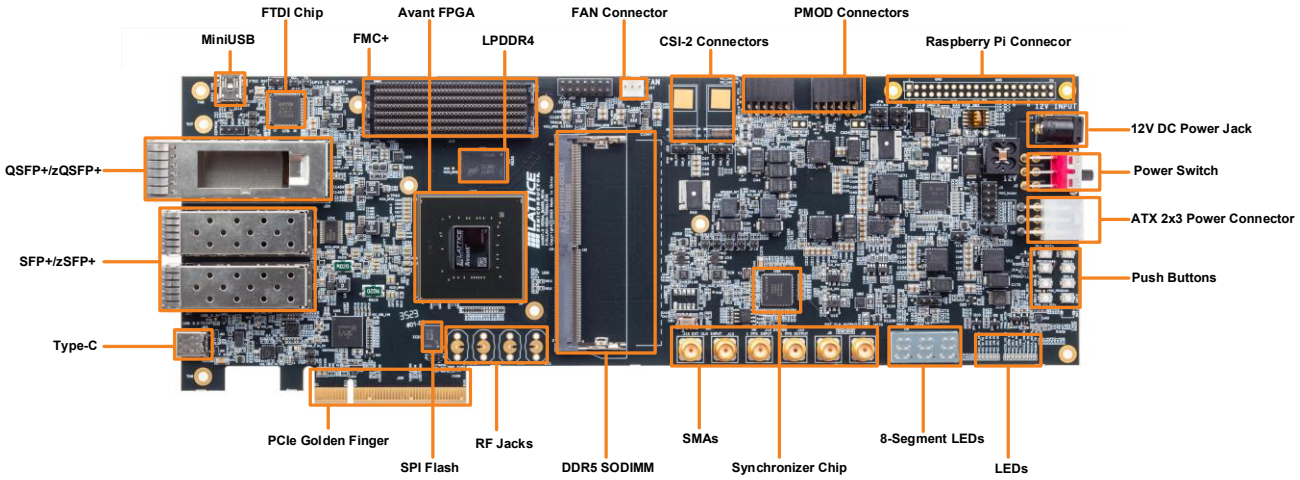

Figure 6.13. Top View of Avant G/X70 Versa Board

For more information on the Avant-G/X Versa Board, refer to the Avant-G/X Versa Board User Guide (FPGA-EB-02063).

The following figure shows the QSFP+ loopback module for external serial loopback testing.

Figure 6.14. Top View of QSFP+ Loopback Module

For more information, refer to the 40G QSFP+ Passive Loopback Testing Module for FS Switches product page on the FS website.

6.5.1. Hardware Setup for Avant X70 10G Ethernet XAUI and Validation Result

Program bitstream into the device and make sure the DIP switch is configured to the right position before triggering the RESET via pushbutton.

6.5.1.1. DIP Switch Definition

The following lists the available switches:

- DIP_SW1: (LOW) traffic generator trigger
- DIP_SW2: (LOW) RESERVED
- DIP_SW3: (LOW) RESERVED
- DIP_SW4: (LOW) RESERVED

6.5.1.2. Pushbutton

The following lists the available pushbuttons:

- SW12 RESET
- SW13 PAUSE / RESUME transmission
- SW14 pattern generator mode change (fix frame size, random frame size, or increment frame size)

To perform RESET after programming the bitstream or configuring the pattern generator mode from DIP_SW, follow these steps:

1. Confirm that the traffic generator (DIP_SW1) is LOW before triggering the RESET.
2. After the link-up is completed successfully, toggle DIP_SW1 from LOW to HIGH to start the traffic generator, which checks the IP performance.
3. Check the 7-segment LED indicator for device status, and pattern generator mode.
4. You may pause the traffic transmission using SW13 if CONTINUOUS_TRAFFIC is enabled.
5. You may change the packet length type generated from the generator by using SW 14. Perform this when the transmission is paused.

For DIG 7-segment definitions, refer to Figure 6.15.

DIG 1, Segment B, C, D, E, F must be ON, when TX and RX are actively transmitting (for CONTINUOUS_TRAFFIC enabled mode).

DIG 1, Segment A must be ON when transmission is paused or completed (for CONTINUOUS_TRAFFIC disabled mode).

The following lists the failing scenario:

- DIG 1, any of Segment B, C, D is OFF after reset.
- DIG 1, any of Segment E, F is OFF after the transmission starts.
- DIG 1, Segment G is ON.

6.5.1.3. 7-Segment LED Definition

The following figure shows the various LED segments for the Avant Versa board.

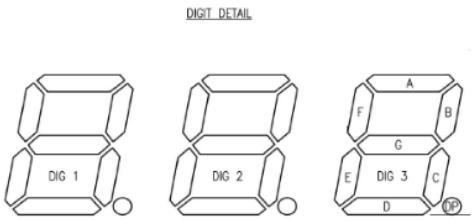

Figure 6.15. 7-Segment LED from Avant-AT-X Devices

Each segment in DIG 1 represents the status of XAUI IP. Refer to the following table for the description of each segment.

Table 6.2. LED 7-Segment Description for DIG 1

Segment	Description
A	Traffic transmission is completed or paused.
B	Reset asserted.
C	Link up is completed successfully.
D	Register configuration is completed successfully.
E	TX transmission is up or transmitting.
F	RX transmission is up or receiving.
G	Data checker detected a mismatch.

DIG 3 displays the code corresponding to different types of frame length, including static or fixed frame size, randomized frame size, and incremental frame size. To change the type of frame size, use pushbutton SW14.

Table 6.3. LED 7-Segment Description for DIG 3

Code for DIG 3	Code Description
0	Fixed frame size.
1	Randomized frame size.
2	Incremental frame size.

The maximum and minimum boundary for randomized and incremental frame size can be defined in the top file, traffic_genchk module. The value shown below is based on the limitations of the traffic_genchk module.

FRAME_LEN_MIN is the minimum value for randomized size and the starting value for incremental frame size. Supports a minimum length of 128.

FRAME_LEN_MAX is the maximum value for randomized size, and the largest value for incremental frame size, which starts over again using the minimum value. Supports a maximum length of 9,600.

FRAME_LEN_INIT is used for defining the static frame size. Supports values between 128 to 9,600.

```
traffic_genchk #(
    .CONTINUOUS_TRAFFIC (CONTINUOUS_TRAFFIC),
    .MAX_DATA_WIDTH(AXI_DATA_WIDTH),
    .MASK_DATA( (AXI_DATA_WIDTH == 128) ? 128'hEF45_4255_3A4C_5ECA_EF45_4255_3A4C_5ECA : 64'hEF45_4255_3A4C_5ECA ),
    .FRAME_LEN_MIN(16'd128),
    .FRAME_LEN_MAX(16'd9600),
    .FRAME_LEN_INIT(16'd1500)
) u_traffic_genchk (
```

Figure 6.16. Parameters for Frame Size in the Top File

Connect QSFP transceiver loopback module to the QSFP+/zQSFP+ cage. The USB-Mini B is connected to the host for the Reveal tool to check the signal from hardware, including TX, RX, and data comparison.

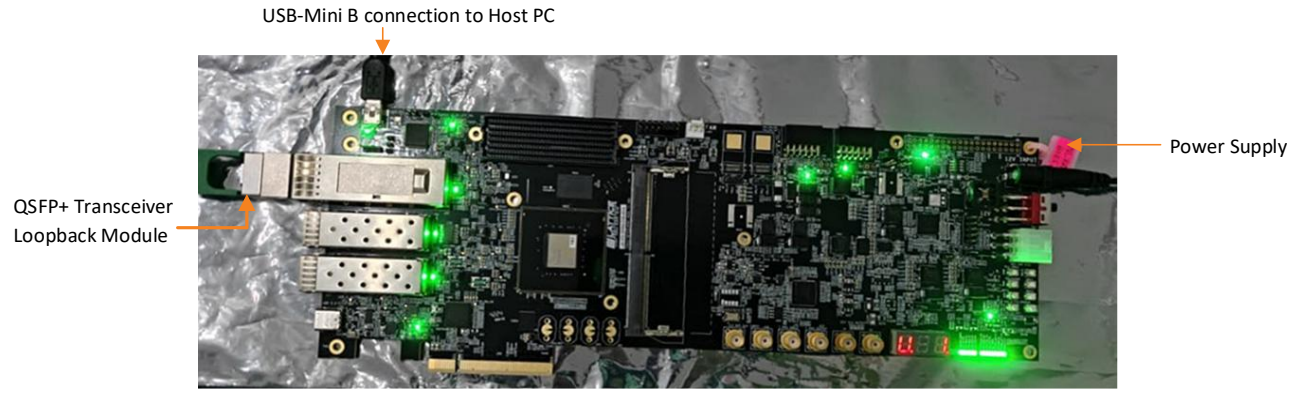

Figure 6.17. Avant-X Versa Board with QSFP+ Transceiver Loopback Module

The following figure shows the continuous transmission using the pattern generator through the QSFP loopback. Each frame is validated. The *compFail_o* signal is asserted if the frame checker detects a mismatch in the traffic checker.

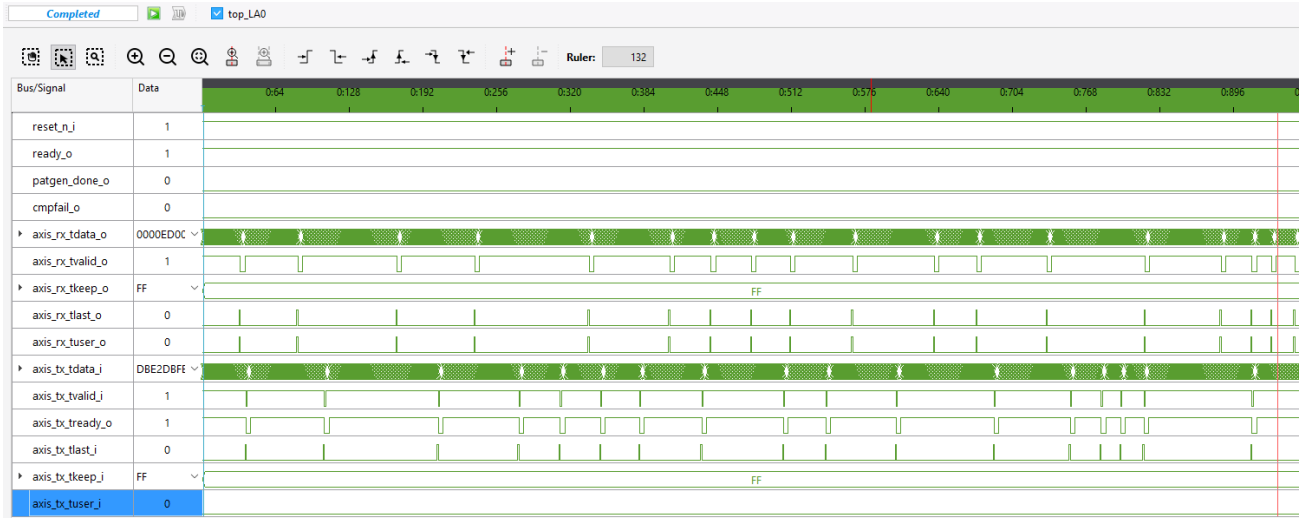

Figure 6.18. Signals shown on the Reveal Tool

The hardware example design comes with a testbench for simulating the top file. Launch the Simulation Wizard from the Radiant software and follow the on-screen instructions. Select tb_top as the simulation top module.

7. Designing with the IP

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

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

7.1. Generating and Instantiating the IP

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

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **XAUI** under the **IP, Connectivity** category. The **Module/IP Block Wizard** opens as shown in Figure 7.1.
3. Enter values in the **Instance name** and **Create in** fields. Click **Next**.

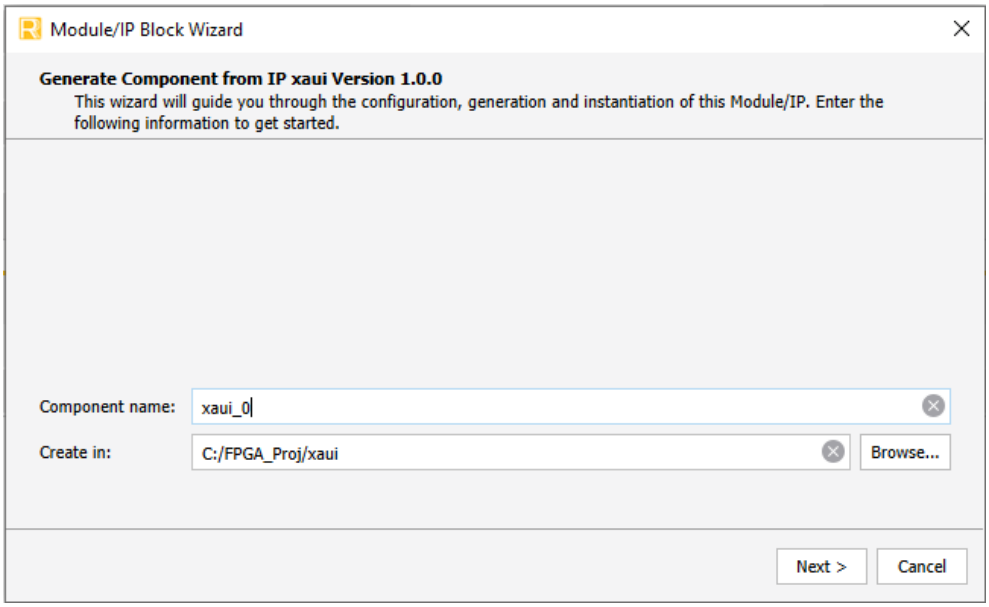

Figure 7.1. Module/IP Block Wizard

4. In the module's dialog box of the **Module/IP Block Wizard** window, customize the selected XAUI IP core using drop-down menus and check boxes. As a sample configuration, see Figure 7.2. For configuration options, see the IP Parameter Description section.

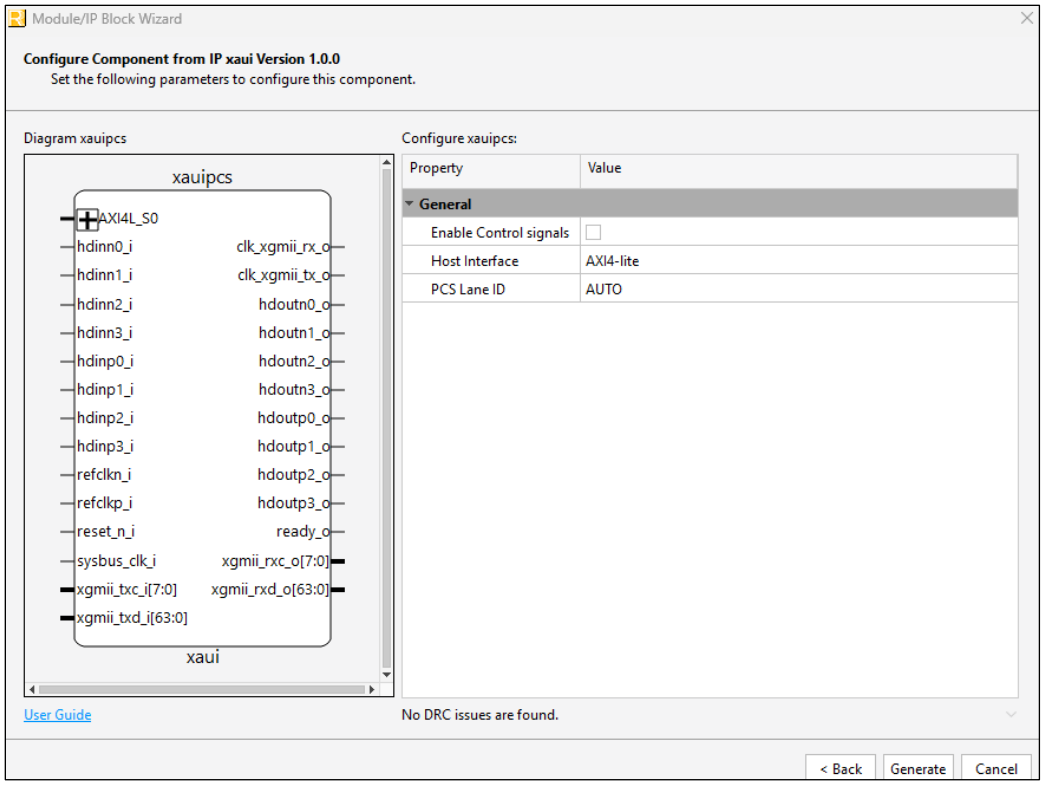

Figure 7.2. Configure User Interface of Selected XAUI IP Core

5. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in Figure 7.3.

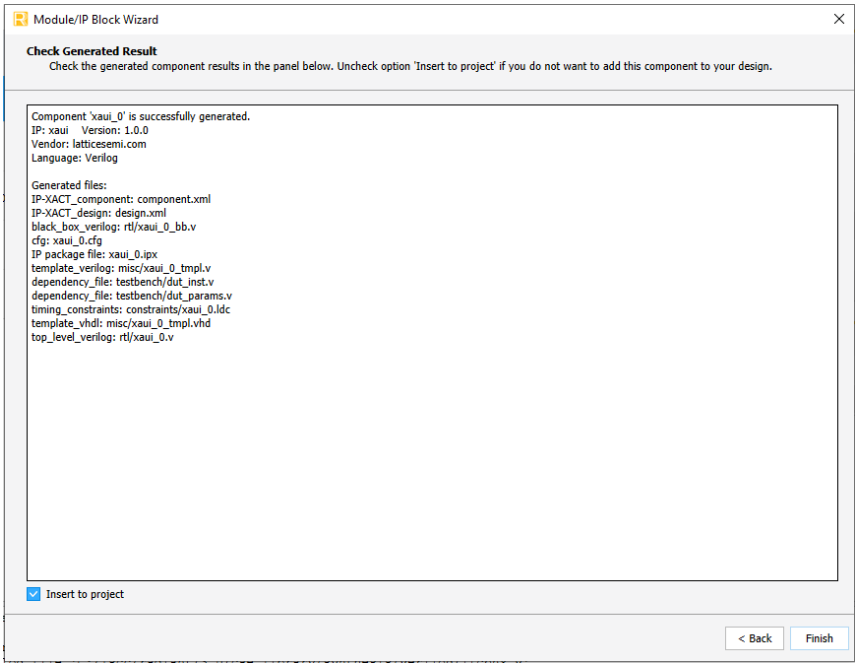

Figure 7.3. Check Generating Result

6. Click the **Finish** button. All the generated files are placed under the directory paths in the **Create in** and **Instance name** fields shown in Figure 7.1.

7.1.1. Generated Files and File Structure

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

Table 7.1. Generated File List

Attribute	Description
<Instance Name>.ipx	This file contains the information on the files associated to the generated IP.
<Instance Name>.cfg	This file contains the parameter values used in IP configuration.
component.xml	Contains the ipxact:component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Instance Name>.v	This file provides an example RTL top file that instantiates the IP core.
rtl/<Instance Name>_bb.v	This file provides the synthesis closed box.
misc/<Instance Name>_tmpl.v misc /<Instance Name>_tmpl.vhd	These files provide instance templates for the IP core.

7.2. Running Functional Simulation

You can run functional simulation after the IP is generated. To run functional simulation, follow these steps:

1. Add the top level testbench file, *tb_top.v* in the project as a simulation file. Click the **File** tab and select **Add** in the drop down menu.
2. Click **Existing Simulation File** and select *<Component name>/testbench/tb_top.v*.
3. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 7.4.

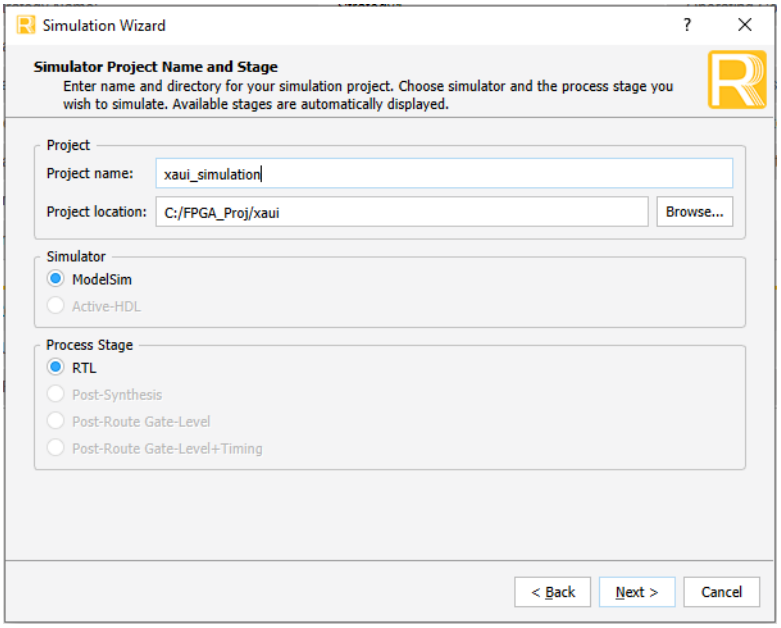

Figure 7.4. Simulation Wizard

4. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 7.5.

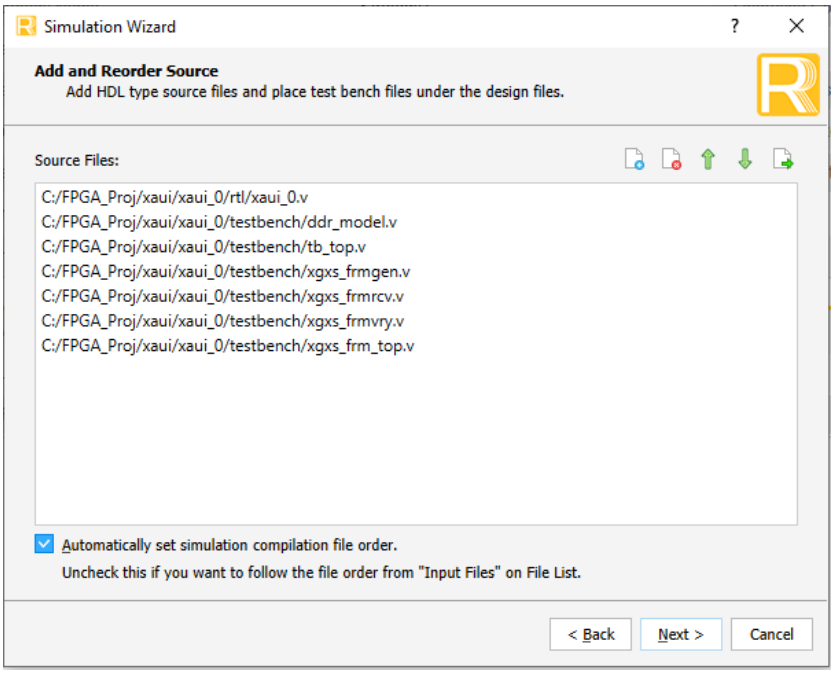

Figure 7.5. Adding and Reordering Source

5. Click **Next**. The Summary window is shown. Click **Finish** to run the simulation.

The result of the simulation in our example is provided in Figure 7.6.

The Transcript window shows the progress of the simulation. At the end of the test, it displays either a PASS or FAIL status. Note that before the ready_o signal is asserted, the XAUI core IP may exhibit clock instability. This can impact output clocks such as clk_xgmii_tx/rx_o and may cause the hdoutp/n_o serial ports to stop transmitting. The instability occurs during the XAUI core IP reset sequence and will persist until the ready_o signal becomes active.

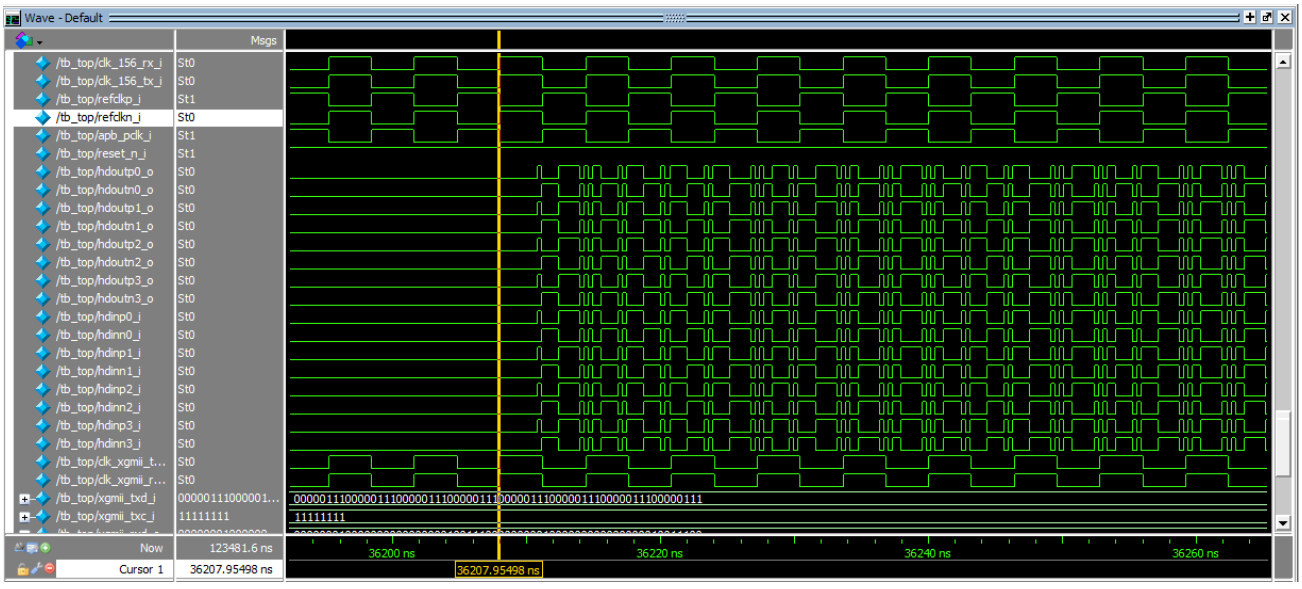

Figure 7.6. Simulation Waveform

7.3. Timing Constraints

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

You must include the following clock constraints in your design per IP configuration. Note that you might need to modify the port name according to your RTL. Alternatively, you can find the following constraints in *<IP>/constraints/constraint.sdc*.

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

7.3.1. Create Clock Constraints for CertusPro-NX and Avant G/X XAUI IP

```
#create 156.25 MHz clock for CertusPro-NX and Avant G/X SerDes refclk pin
create_clock -name {refclkp_i} -period 6.4 [get_ports refclkp_i]
create_clock -name {refclkn_i} -period 6.4 [get_ports refclkn_i]

#create 156.25 MHz clock for CertusPro-NX PMA logic. This clock is not needed for Avant G/X.
create_clock -name {clk_156_tx_i} -period 6.4 [get_ports clk_156_tx_i]
create_clock -name {clk_156_rx_i} -period 6.4 [get_ports clk_156_rx_i]

#create 100 MHz clock for sysbus_clk_i, supported maximum frequency up to 100 MHz
create_clock -name {sysbus_clk_i} -period 10 [get_ports sysbus_clk_i]
```

Appendix A. Resource Utilization

Table A.1 shows the resource utilization of the XAUI IP default configuration for the LFCPNX device using Lattice Synthesis Engine of the Lattice Radiant software.

Table A.1. Resource Utilization

Configuration	Registers	LUTs	EBRs	Target Device	Synthesis Tools
XAUI Core	1,570 (2.0%)	2,061 (2.6%)	0 (0%)	LFCPNX-100	Synplify Pro
XAUI Core	1,702 (< 1%)	2,159 (<1%)	0 (0%)	LAV-AT-X70	Synplify Pro

References

- XAUI IP Release Notes (FPGA-RN-02104)
- MPCS Module User Guide (FPGA-IPUG-02118)
- 10G Ethernet IP User Guide (FPGA-IPUG-02245)
- CertusPro-NX SERDES/PCS User Guide (FPGA-TN-02245).
- Lattice Avant-G/X MPPHY Module User Guide (FPGA-IPUG-02233)
- Lattice Avant SERDES/PCS User Guide (FPGA-TN-02313)
- Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide (FPGA-UG-02039)
- Lattice Radiant Timing Constraints Methodology Application Note (FPGA-AN-02059)
- Avant-G/X Versa Board User Guide (FPGA-EB-02063)
- CertusPro-NX FPGA web page
- Avant-G web page
- Avant-X web page
- Lattice Radiant Software web page
- Lattice Propel Design Environment web page
- Lattice Solutions IP Cores web page
- Lattice Solutions Reference Designs web page
- Lattice Solutions Boards web page
- Lattice Solutions Demonstrations web page
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

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

Revision History

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

Revision 1.1, IP v.1.1.0, December 2025

Section	Change Summary
All	Added IP version on the cover page.
Abbreviations in This Document	Updated the section title *Acronyms in This Document* to *Abbreviations in This Document*.
Introduction	• Updated Table 1.1. XAUI IP Quick Facts. • Added the IP Support Summary section. • Updated the Features section. • Moved the Licensing and Ordering Information section to this section and updated the content. • Added the Minimum Device Requirements section.
Functional Description	• Updated the IP Architecture Overview section. • Updated the XGMII Interface section. • Updated the introduction for the XAUI-to-XGMII Translation (Receive Interface) section. • Removed the Rx Slip Buffer section from the XAUI-to-XGMII Translation (Receive Interface) section. • Removed the Tx Slip Buffer section from the XGMII to XAUI Translation (Transmit Interface) section. • Updated the XAUI RX Core section. • Updated the XAUI TX Core section. • Updated the *Timing of Data Frame Transfer from Client-side Interface* sections in the XAUI-to-XGMII Translation (Receive Interface) section and the XGMII to XAUI Translation (Transmit Interface) section. • Removed the MDIO Interface section.
IP Parameter Description	• Updated the section title from *Attribute Summary* to *IP Parameter Description*. • Removed the Licensing the IP section. • Updated Table 3.1. General Attributes.
Signal Description	Updated Table 4.1. XAUI IP Core Signal Description.
Register Description	• Updated the APB Interface Registers section. • Added the PHY Management Block section.
Example Design	Added this section.
Designing with the IP	• Updated the section title from *IP Generation and Evaluation* to *Designing with the IP*. • Updated the section title for *Generation and Synthesis* to *Generating and Instantiating the IP*. • Updated Figure 7.2. Configure User Interface of Selected XAUI IP Core. • Updated the Running Functional Simulation section. • Added the Timing Constraints section.
Appendix A. Resource Utilization	Updated Table A.1. Resource Utilization.
References	Updated the references in this section.
Technical Support Assistance	Updated the content in this section.

Revision 1.0, October 2021

Section	Change Summary
All	Initial release.

www.latticesemi.com