XAUI IP Core

User's Guide

November 2009
ipug68_01.3

Introduction

The 10Gb Ethernet Attachment Unit Interface (XAUI) IP Core User's Guide for the LatticeECP2M™ and LatticeECP3™ FPGAs provides a solution for bridging between XAUI and 10-Gigabit Media Independent Interface (XGMII) devices. This user's guide implements 10Gb Ethernet Extended Sublayer (XGXS) capabilities in soft logic that together with PCS and SERDES functions implemented in the FGPA provides a complete XAUI-to-XGMII solution.

The XAUI IP core package comes with the following documentation and files:

- Protected netlist/database
- Behavioral RTL simulation model
- Source files for instantiating and evaluating the core

The XAUI IP core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase on an IP license. It may also be used to evaluate the core in hardware in user-defined designs. Details for using the hardware evaluation capability are described in the Hardware Evaluation section of this document.

Features

- XAUI compliant functionality supported by embedded SERDES PCS functionality implemented in the LatticeECP2M and LatticeECP3, including four channels of 3.125 Gbps serializer/deserializer with 8b10b encoding/decoding.
- Complete 10Gb Ethernet Extended Sublayer (XGXS) solution based on LatticeECP2M and LatticeECP3 FPGA.
- Soft IP targeted to the FPGA implements XGXS functionality conforming to IEEE 802.3ae-2002, including:
  - 10 GbE Media Independent Interface (XGMII).
  - Optional Slip buffers for clock domain transfer to/from the XGMII interface.
  - Complete translation between XGMII and XAUI PCS layers, including 8b10b encoding and decoding of Idle, Start, Terminate, Error and Sequence code groups and sequences, and randomized Idle generation in the XAUI transmit direction.
  - XAUI compliant lane-by-lane synchronization.
  - Lane deskew functionality.
  - Interface with the high-speed SERDES block embedded in the LatticeECP2M and LatticeECP3 that implements a standard XAUI.
  - Optional standard compliant MDIO/MDC interface.
- ModelSim® simulation models and test benches provided for free evaluation.

General Description

XAUI is a high-speed interconnect that offers reduced pin count and has the ability to drive up to 20 inches of PCB trace on standard FR-4 material. Each XAUI interface comprises four self-timed 8b10b encoded serial lanes each operating at 3.125 Gbps and thus is capable of transferring data at an aggregate rate of 10 Gbps.

XGMII is a 156 MHz Double Data Rate (DDR), parallel, short-reach interconnect interface (typically less than 2 inches). 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 are shown in Figure 1. A simplified block diagram of the XAUI solution is shown in Figure 2. The XGMII interface, XGXS coding and state machines and XAUI multichannel alignment capabilities are implemented in the FPGA array. The XAUI 8b10b coding and SERDES functionality are supported by the embedded SERDES_PCS block. An optional MDIO interface module is also implemented in the FPGA array.

Figure 3 shows the I/O interface view of the XAUI IP core.

Figure 1. XAUI and XGXS Locations in 10 GbE Protocol Stack

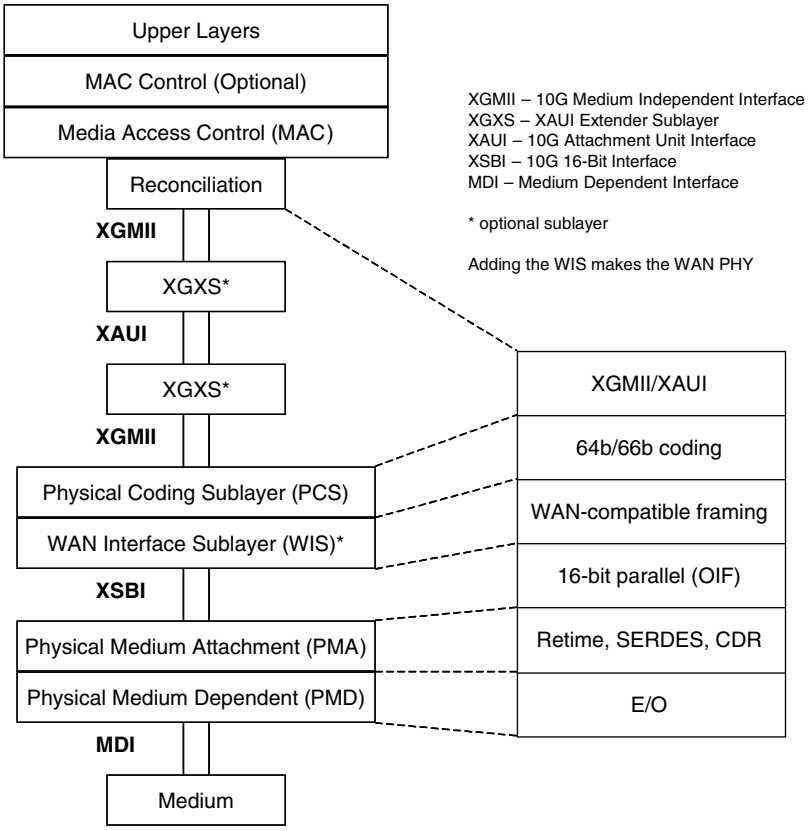


Figure 2. XAUI Solution Simplified Block Diagram

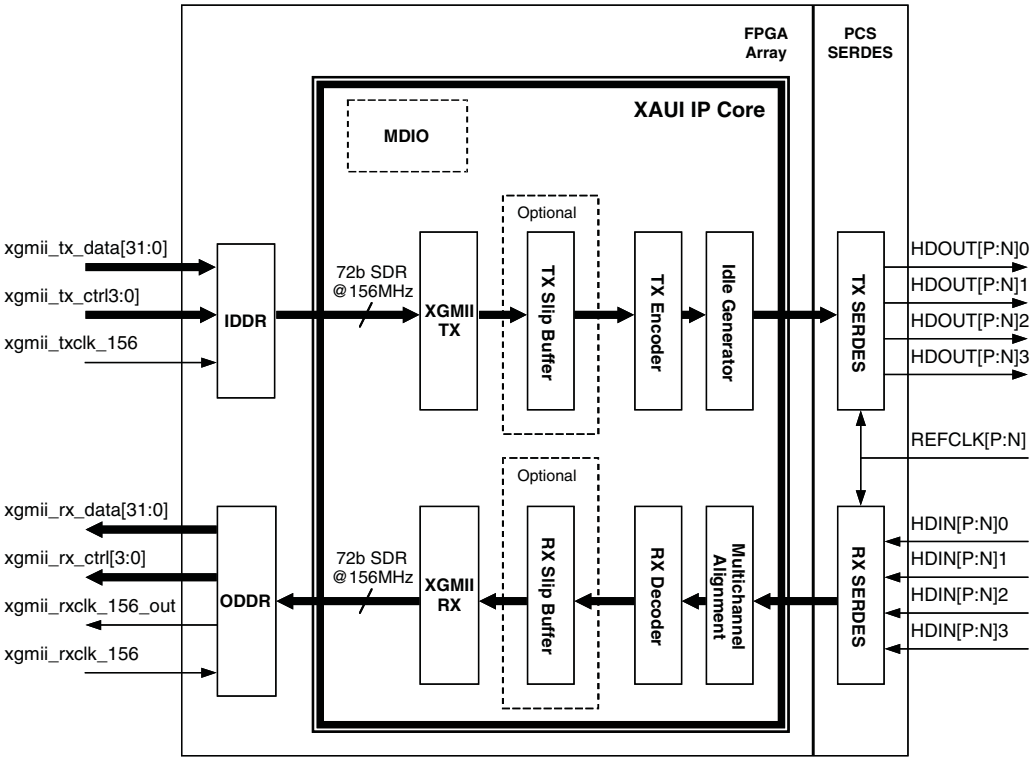
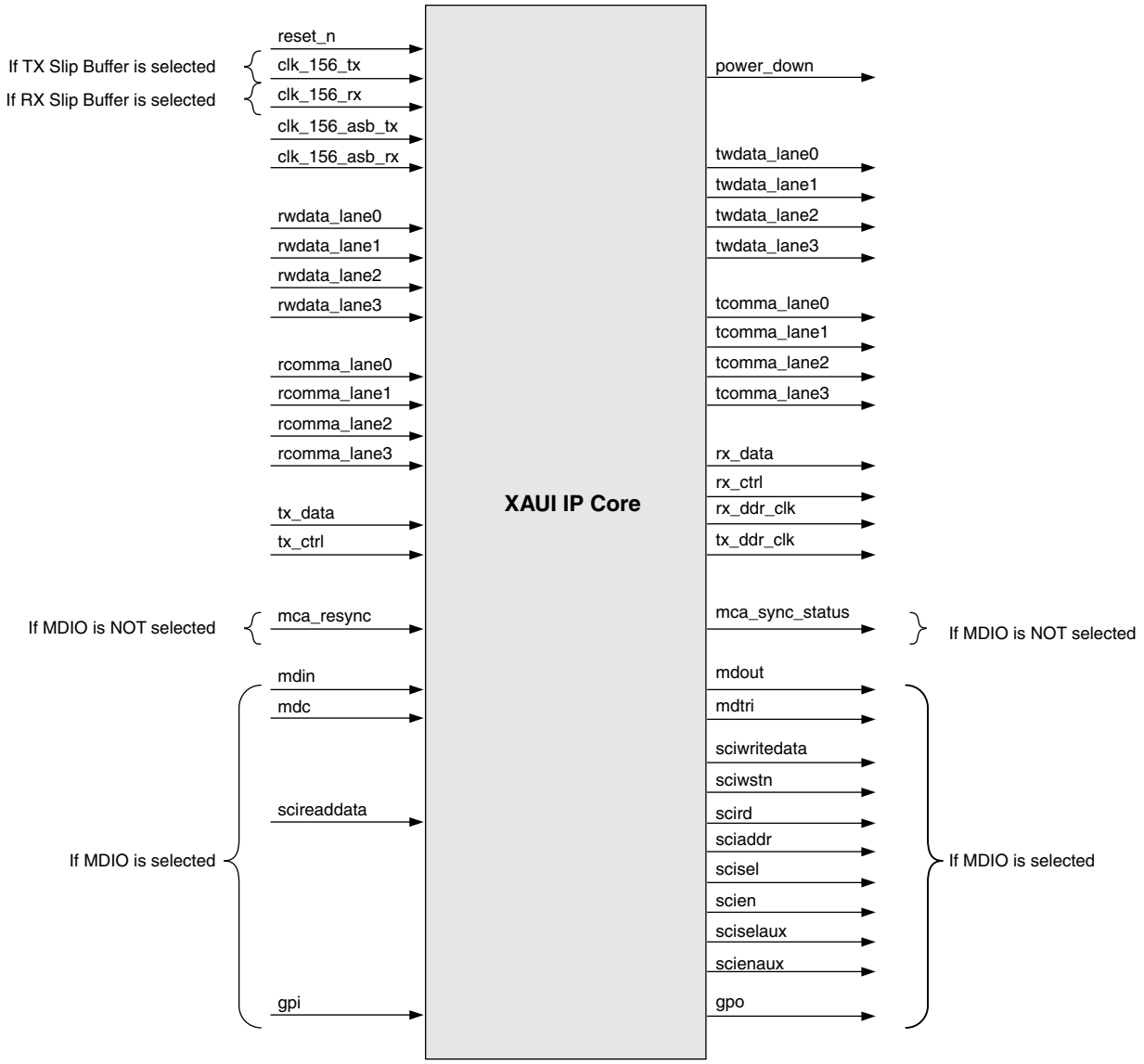


Figure 3. XAUI IP Core I/O



XAUI IP Core I/O

Table 1 defines all I/O interface ports available in this core.

Table 1. IP Core I/O and Signal Definitions

Name	Width (Bits)	Direction	Description
All Configurations			
reset_n	1	I	Active-low asynchronous reset
clk_156_asb_tx	1	I	156MHz XAUI transmit clock (from PCS)
clk_156_asb_rx	1	I	156MHz XAUI receive clock (from PCS)
rwdata_lane0	16	I	XAUI receive data channel 0 (from PCS)
rwdata_lane1	16	I	XAUI receive data channel 1 (from PCS)
rwdata_lane2	16	I	XAUI receive data channel 2 (from PCS)
rwdata_lane3	16	I	XAUI receive data channel 3 (from PCS)
rcomma_lane0	2	I	XAUI receive control channel 0 (from PCS)
rcomma_lane1	2	I	XAUI receive control channel 1 (from PCS)
rcomma_lane2	2	I	XAUI receive control channel 2 (from PCS)
rcomma_lane3	2	I	XAUI receive control channel 3 (from PCS)
twdata_lane0	16	O	XAUI transmit data channel 0 (to PCS)
twdata_lane1	16	O	XAUI transmit data channel 1 (to PCS)
twdata_lane2	16	O	XAUI transmit data channel 2 (to PCS)
twdata_lane3	16	O	XAUI transmit data channel 3 (to PCS)
tcomma_lane0	2	O	XAUI transmit control channel 0 (to PCS)
tcomma_lane1	2	O	XAUI transmit control channel 1 (to PCS)
tcomma_lane2	2	O	XAUI transmit control channel 2 (to PCS)
tcomma_lane3	2	O	XAUI transmit control channel 3 (to PCS)
rx_ddr_clk	1	O	DDR clock from IP Core to the XGMII TX direction
tx_ddr_clk	1	O	DDR clock from IP Core to the XGMII receive direction
tx_data	64	I	IP Core transmit data from XGMII RX
tx_ctrl	8	I	IP Core transmit control from XGMII RX
rx_data	64	O	Receive data from core and sent to XGMII TX
rx_ctrl	8	O	Receive control from core and sent to XGMII TX
With Optional RX Slip Buffer			
clk_156_rx	1	I	156MHz XGMII TX clock
With Optional TX Slip Buffer			
clk_156_tx	1	I	156MHz XGMII RX clock
No MDIO Option			
mca_resync	1	I	XAUI multi-channel alignment resynchronization request
mca_sync_status	1	O	XAUI multi-channel alignment status. 1 = All XAUI channels are aligned 0 = XAUI channels are not aligned
MDIO Option			
mdin	1	I	MDIO serial input data
mdc	1	I	MDIO input clock
mdout	1	O	MDIO serial output data
mdtri	1	O	Tristate control for MDIO port
scireaddata	8	I	SCI read data (from PCS)
sciwritedata	8	O	SCI write data (to PCS)

Table 1. IP Core I/O and Signal Definitions (Continued)

Name	Width (Bits)	Direction	Description
sciwstn	1	O	SCI write strobe
scird	1	O	SCI read probe
sciaddr	6	O	SCI address bus
scisel	4	O	SCI quad select
scien	4	O	SCI quad enable
sciselaux	1	O	SCI auxiliary select
scienaux	1	O	SCI auxiliary enable
gpi	4	I	General purpose input
gpo	4	O	General purpose output

Parameter Descriptions

Table 2 lists the user-configurable parameters for the XAUI IP core.

Table 2. User-Configurable Parameters

Parameter	Description	Range	Default
TX_SLIP_BUFFER	Include TX slip buffer for clock tolerance compensation	Yes/No	Yes
RX_SLIP_BUFFER	Include RX slip buffer for clock tolerance compensation	Yes/No	Yes
MDIO	Include MDIO	Yes/No	No

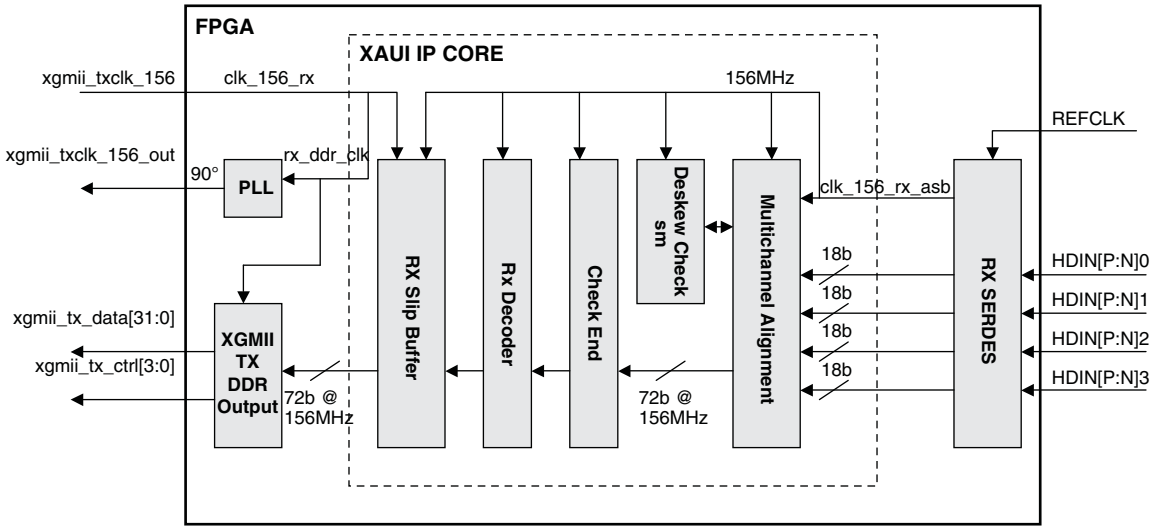
Functional Description

The XAUI receive path, shown in Figure 4, is the data path from the XAUI to the XGMII interface. In the receive direction, 8b10b encoded data received at the XAUI SERDES interface is demultiplexed and passed to the multi-channel alignment block, that compensates for lane-to-lane skew between the four SERDES channels as specified in 802.3. The aligned data streams are passed to decoder logic, where it is translated and mapped to the XGMII data format. The output of the encoder is then passed through an optional slip buffer that compensates for XAUI and XGMII timing differences and then to the XGMII 36-bit (32-bit data and four control bits) 156 MHz DDR external interface.

The receive direction data translations are shown in Figure 5. The 8b10b symbols on each XAUI lane are converted to XGMII format and passed to the corresponding XGMII lane. The XGMII comprises four lanes, labeled [0:3], and one clock in both transmit and receive directions. Each lane includes eight data signals and one control signal. Double Data Rate (DDR) transmission is utilized, with the data and control signals sampled on both the rising and falling edges of a 156.25 MHz (nom) clock for an effective data transfer rate of 2.5Gbps.

Figure 4. XAUI IP Core Receive Path

With Optional Receive Direction Slip Buffer



Without Optional Receive Direction Slip Buffer

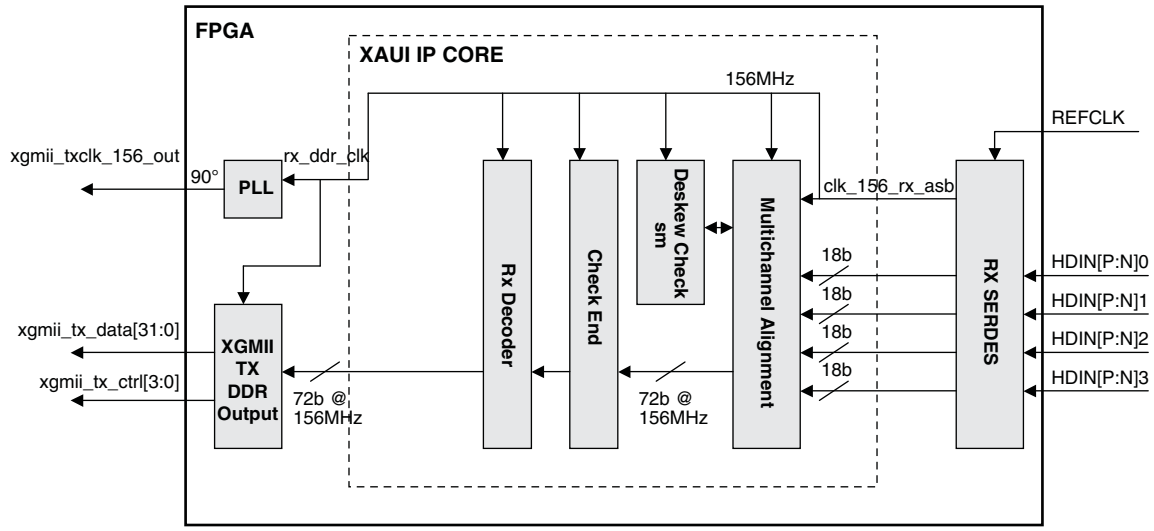
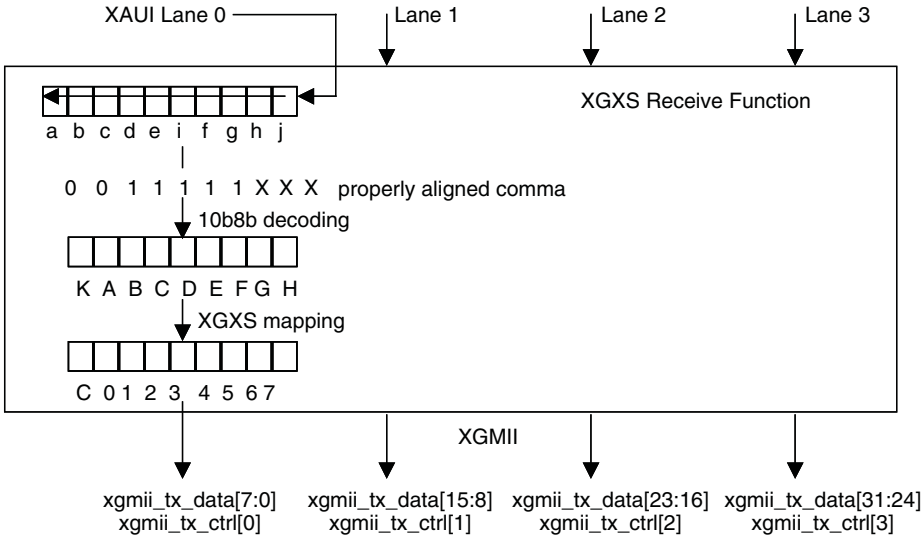


Figure 5. XAUI IP Core Receive Direction Data Translations

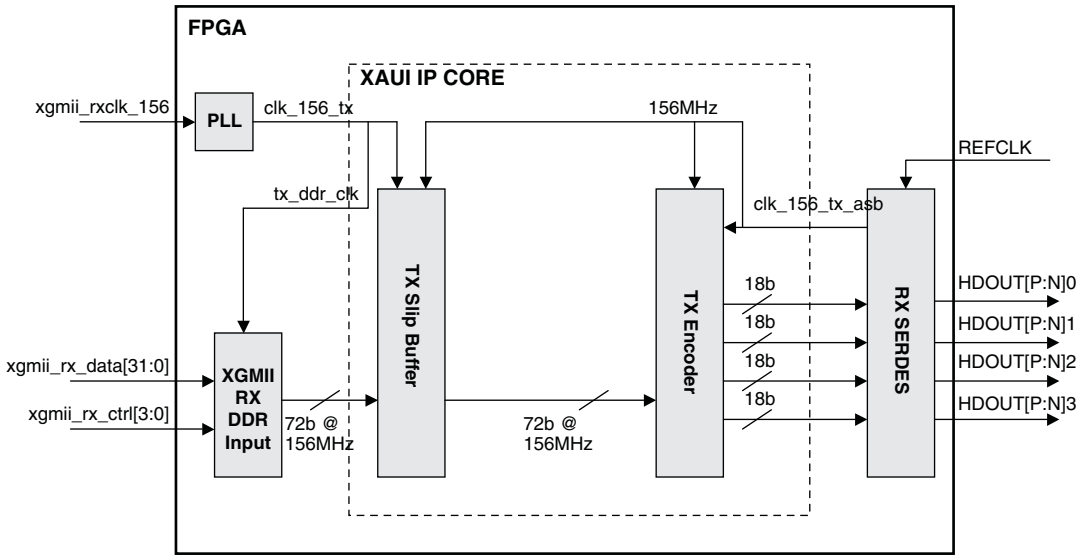


The transmit path, shown in Figure 6, is the data path from XGMII to XAUI. In the transmit direction, the 36-bit DDR data and control received at the XGMII are converted to single-edge timing and passed through an optional slip buffer that compensates for XAUI and XGMII timing differences. The XGMII data and control are then passed to the TX encoder, where they are translated and mapped to the 8b10b XAUI transmission code and then passed to the SERDES interface.

The transmit direction data translations are shown in Figure 7. Data and control from each of the four XGMII lanes are translated and mapped to the corresponding XAUI lanes. The transmit encoder includes the transmit idle generation state machine that generates a random sequence of /A/, /K/ and /R/ code groups as specified in IEEE 802.3ae.

Figure 6. XAUI IP Core Transmit Path

With Optional Transmit Direction Slip Buffer



Without Optional Transmit Direction Slip Buffer

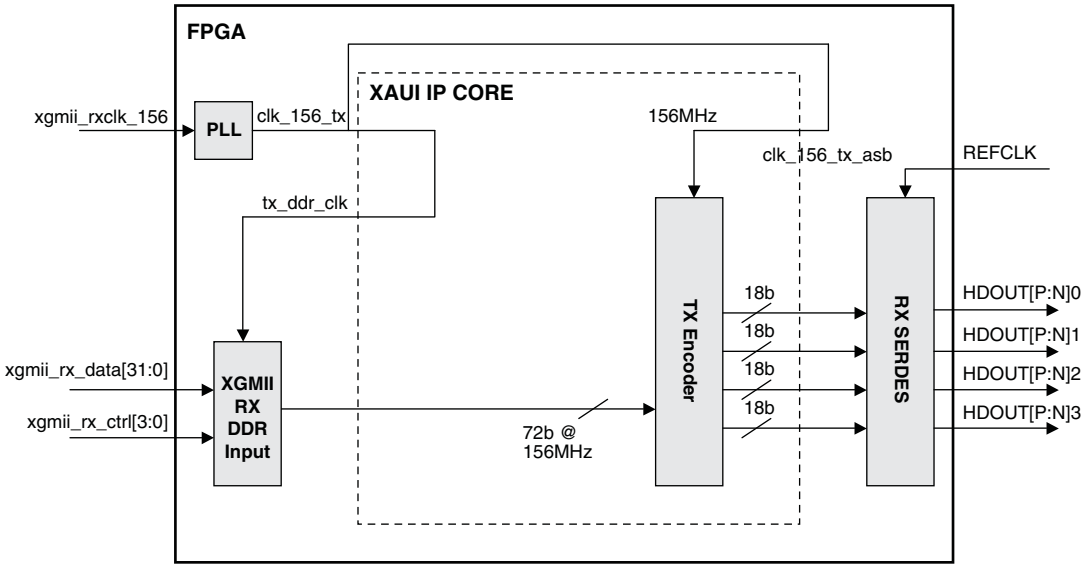
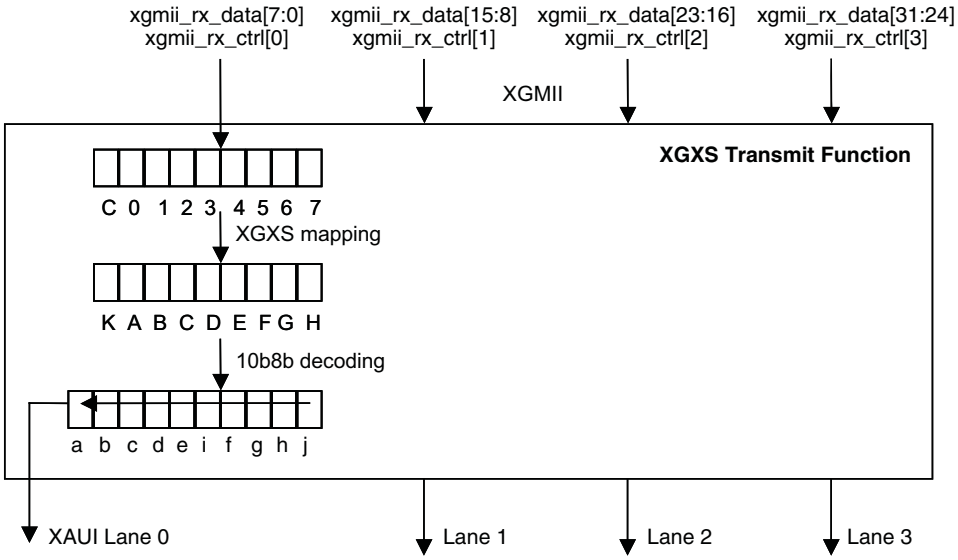


Figure 7. XAUI IP Core Transmit Direction Data Translations



XGMII and Slip Buffers

The 10Gigabit Media Independent Interface (XGMII) supported by the XAUI IP core solution conforms to Clause 46 of IEEE 802.3ae.The XGMII is composed of independent transmit and receive paths. Each direction uses 32 data signals, four control signals and a clock. The 32 data signals in each direction are organized into four lanes of eight signals each. Each lane is associated with a control signal as shown in Table 3.

Table 3. XGMII Transmit and Receive Lane Associations[1]

Tx data (xgmii_rx_data) Rx data (xgmii_tx_data)	Tx control (xgmii_rx_ctrl) Rx control (xgmii_tx_ctrl)	XAUI Lane
[7:0]	[0]	0
[15:8]	[1]	1
[23:16]	[2]	2
[31:24]	[3]	3

1. The XGMII TX signals are XGXS inputs to the transmit path (XGMII to XAUI). The XGMII RX signals are XGXS outputs of the receive path (XAUI to XGMII).

The XGMII supports Double Data Rate (DDR) transmission (i.e., the data and control input signals are sampled on both the rising and falling edges of the corresponding clock). The XGXS XGMII input (RX) data is sampled based on an input clock typically sourced from the XGMII device running at 156.25 MHz, 1/64th of the 10Gb data rate and is transmitted by the IP transmit part. The XGXS XGMII output (TX) data is referenced to a forwarded clock that is phase locked to a 156.25 MHz (typical) input reference, and the data is received from the IP receiver part.

The control signal for each lane is de-asserted when a data octet is being sent on the corresponding lane and asserted when a control character is being sent. Supported control octet encodings are shown in Table 4. All data and control signals are passed directly to/from the 8b10b encoding/decoding blocks with no further processing by the XGMII block. Note that the packet Start control word is only valid on lane 0.

Table 4. XGMII Control Encoding

Control	Data	Description
0	0x00 - 0xFF	Normal data transmission
1	0x00 - 0x06	Reserved
1	0x07	Idle
1	0x08 - 0x9B	Reserved
1	0x9C	Sequence (only valid on lane 0)
1	0x9D - 0xFA	Reserved
1	0xFB	Start (only valid on lane 0)
1	0xFC	Reserved
1	0xFD	Terminate
1	0xFE	Error
1	0xFF	Reserved

The XGMII blocks incorporate optional slip buffers that accommodate small differences between XGMII and XAUI timing by inserting or deleting idle characters. The slip buffer is implemented as a 256 x 72 FIFO. There are four flags out of the FIFO: full, empty, partially full, and partially empty. The partially empty flag is used as the watermark to start reading from the FIFO. If the difference between write and read pointers falls below the partially empty watermark and the entire packet has been transmitted, idle characters are inserted until the partially full watermark is reached. No idle is inserted during data transmission.

XAUI-to-XGMII Translation (Receive Interface)

A block diagram of the XAUI IP core receive data path was shown previously in Figure 4. The IP receive interface converts the incoming XAUI stream into XGMII-compatible signals. At the embedded core interface, the IP core receive block receives 72 bits of data at 156 MHz (64 bits of data, 8 bits of control) from four XAUI lanes. Data from the embedded core are first passed to the RX multi-channel aligner block to de-skew the four XAUI lanes.

The multi-channel alignment block consists of four 16 depth FIFO to buffer each individual XAUI lane. The block searches for the present of alignment character /A/ in each XAUI lane, and begin 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 of the alignment characters /A/ are aligned among all four XAUI lanes. Once synchronization is achieved, the block would not resynchronize until the 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 XGMII code. Table 5 shows the 8b10b code points. Table 6 shows the code mapping between the two domains in the receive direction. XAUI /A/, /R/, /K/ characters are translated into XGMII Idle (/I/) characters.

Data from the RX decoder block is written to the RX slip buffer. As mentioned previously, the slip buffers are required to compensate for differences in the write and read clocks derived from the XAUI and XGMII reference clocks, respectively. Clock compensation is achieved by deleting (not writing) idle cells into the buffer when the "almost full" threshold is reached and by inserting (writing) additional idle cells into the buffer when the "almost empty" threshold is reached. All idle insertion/deletion occurs during the Inter-Packet Gap (IPG) between data frames.

Table 5. 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
/E/	Error	Error Propagation	K30.7
/Q/	Sequence	Link Status Message Indicator	K28.4
/d/	Data	Information Bytes	Dxx.x

Table 6. XAUI 8b10b to XGMII Code Mapping

8b10b Data from Embedded Core [7:0]	XGMII Data [7:0]	XGMII Control [3:0]
Dxx.x	0x00-0xFF (Data)	0
K28.5 (0xBC)	0x07 (Idle)	1
K28.3 (0x7C)	0x07 (Idle)	1
K28.0 (0x1C)	0x07 (Idle)	1
K27.7 (0xFB)	0xFB (Start)	1
K29.7 (0xFD)	0xFD (Terminate)	1
K30.7 (0xFE)	0xFE (Error)	1
K28.4 (0x9C)	0x9C (Ordered Set)	1

XGMII-to-XAUI Translation (Transmit Interface)

A block diagram of the XAUI IP core transmit data path was shown previously in Figure 6. The TX interface converts the incoming XGMII data into XAUI-compatible characters. 36-bit XGMII DDR input data and control signals are initially converted to a 72-bit bus based on a single edge 156MHz clock. The data and control read are then passed into an optional TX slip buffer identical to the one used for the RX interface.

After the slip buffer, the XGMII formatted transmit data and control are input to the TX encoder that converts the XGMII characters into 8b10b format as shown in Table 7. 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 802.3ae. 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.

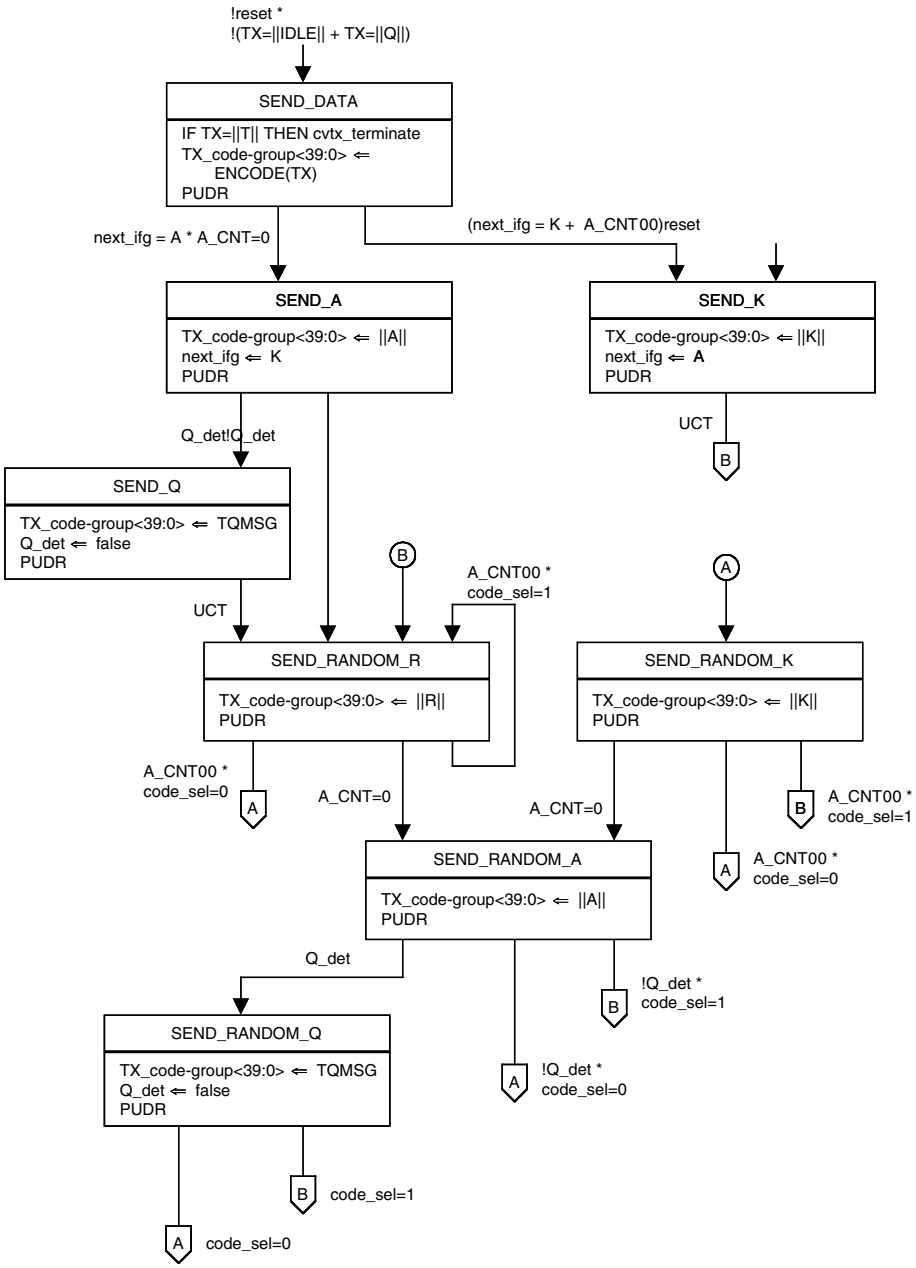
Table 7. XGMII to XAUI Code Mapping

	XGMII	XAUI
Idle	/I/ = 0x07	Randomized /A/, /R/, /K/ Sequence /A/ = K28.3 = 0x7C /R/ = K28.0 = 0x1C /K/ = K28.5 = 0xBC (Comma)
Start	/S/ = 0xFB	/S/ = 0xFB
Error	/E/ = 0xFE	/E/ = 0xFE
Terminate	/T/ = 0xFD	/T/ = 0xFD
Ordered Set	/Q/ = 0x9C	/Q/ = 0x9C
Data	Control = 0	Control = 0

The random /A/, /R/, /K/ sequence is generated as specified in section 48.2.5.2.1 of IEEE 802.3ae and shown in the state machine diagram given in Figure 8. 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 (A_cnt in Figure 8). ||K|| and ||R|| selection is driven by the value of the least significant bit of the generated integer value (code_sel in Figure 8). The idle generation state machine specified in IEEE 802.3ae and shown in Figure 7 transitions between states based on a 312 MHz system clock. The TX encoder implemented in the XAUI IP runs at a system clock rate of 156 MHz. Thus the XGXS state machine implementation performs the equivalent of two state transitions each clock cycle.

Figure 8. XGXS Idle Transmit State Diagram

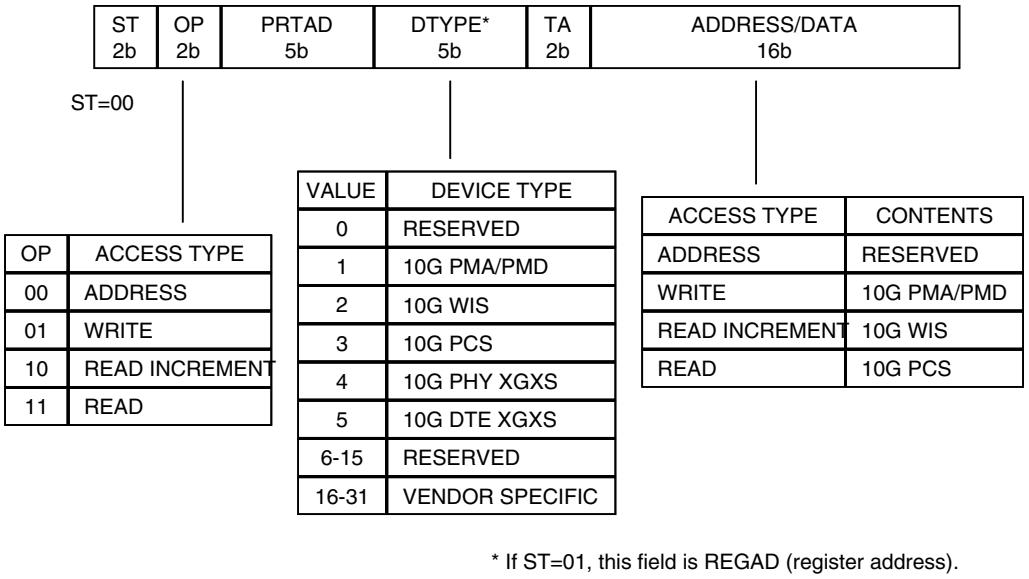


Management Data Input/Output (MDIO) Interface (Optional)

The MDIO interface provides access to the internal XAUI core registers. The register access mechanism corresponds to Clause 45 of IEEE 802.3ae. The XAUI core provides access to XGXS registers 0x0000-0x0024 as specified in IEEE 802.3ae. Additional registers in the vendor-specific address space have been allocated for implementation-specific control/status functions.

The physical interface consists of two signals: MDIO to transfer data/address/control to and from the device, and MDC, a clock up to 2.5 MHz sourced externally to provide the synchronization for MDIO. The fields of the MDIO transfer are shown in Figure 9.

Figure 9. Fields of MDIO Protocol



* If ST=01, this field is REGAD (register address).

Management Frame Structure

Each management data frame consists of 64 bits. The first 32 bits are preamble consisting of 32 contiguous 1s on the MDIO. Following the preamble is the start-of-frame field (ST) which is a 00 pattern. The next field is the operation code (OP) that is shown in Figure 9.

The next two fields are the port address (PRTAD) and device type (DTYPE). Since the physical layer function in 10 GbE is partitioned into various logical (and possibly separate physical) blocks, two fields are used to access these blocks. The PRTAD provides the overall address to the PHY function. The first port address bit transmitted and received is the MSB of the address. The DTYPE field addresses the specific block within the physical layer function.

Device address zero is reserved to ensure that there is not a long sequence of zeros. If the ST field is 01 then the DTYPE field is replaced with REGAD (register address field of the original clause 22 specification). The XAUI core does not respond to any accesses with ST = 01.

The TA field (Turn Around) is a 2-bit turnaround time spacing between the device address field and the data field to avoid contention during a read transaction. The TA bits are treated as don't cares by the XAUI core.

During a write or address operation, the address/data field transports 16 bits of write data or register address depending on the access type. The register is automatically incremented after a read increment. The address/data field is 16 bits.

For an address cycle, this field contains the address of the register to be accessed on the next cycle. For read/write/increment cycles, the field contains the data for the register. The first bit of data transmitted and received in the address/data field is the MSB (bit 15). An example access is shown in Figure 10.

Figure 10. Indirect Address Example

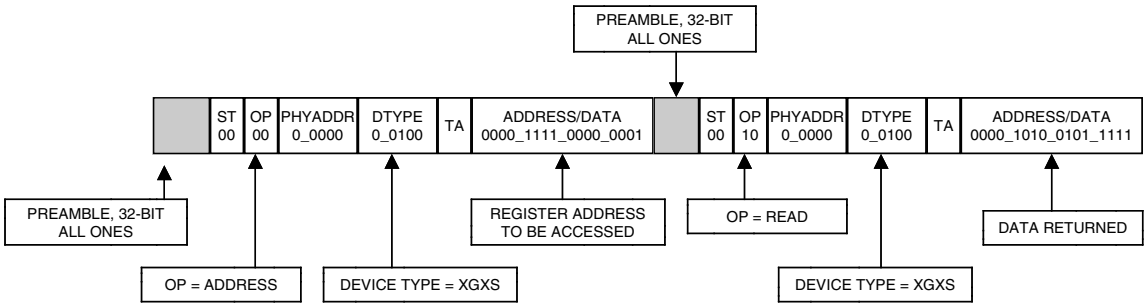


Table 8 shows PHY XGXS registers as described in IEEE Draft P802.3ae. The shaded areas are used to indicate register addresses that are specified in the draft but are not used in this implementation.

There are two vendor supported register ranges. The 4.8000h register range is used for accessing and programming the XGXS registers implemented in the programmable array. All corresponding registers are listed in Table 9. All PCS embedded core registers can be accessed thru the 4.9xxxh registers shown in Table 10, where the address is directly mapped to the PCS embedded registers.

Register Descriptions

Table 8. Register Map for XAUI IP Core (Device Address = 4)

Register Address	Register Name
0	PHY XGXS Control 1
1	PHY XGXS Status 1
2, 3	PHY XGXS Identifier
4	Reserved
5	PHY XGXS Status 2
6 - 23	Reserved
24	10G PHY XGXS Lane status
25 - 32767	Reserved
32768 - 65535	Vendor specific

Table 9. XAUI IP Core Registers

Bit(s)	Name	Description	R/W	Reset Value
Control 1 Register				
4.0.15	Reset	1 = PHY XA reset, 0 = Normal operation	R/W S/C	0
4.0.14	Loopback (not supported)	Loop back functionality is supported in the PCS core. The XAUI core does not provide loopback capability.	R	0
4.0.13	Speed Selection	Value always 0	R	0
4.0.12	Reserved	Value always 0	R	0
4.0.11	Low Power	0= Low Power Mode 1= Normal operation This bit controls the power_down signal of the XAUI core.	R/W	1
4.0.[10:7]	Reserved	Value always 0	R	0
4.0.6	Speed Selection	Value always 0	R	0
4.0.[5:2]	Speed Selection	Value always 0	R	0

Table 9. XAUI IP Core Registers (Continued)

Bit(s)	Name	Description	R/W	Reset Value
4.0.[1:0]	Reserved	Value always 0	R	0
Status 1 Register				
4.1.[15:8]	Reserved	Value always 0	R	0
4.1.7	Fault (not supported)	0 = No Fault condition	R	0
4.1.[6:3]	Reserved	Value always 0	R	0
4.1.2	PHY XS TX link status (not supported)	The Link status is available in the PCS core register.	R	0
4.1.1	Low Power Ability	1 = Low Power Mode support	R	1
4.1.0	Reserved	Value always 0	R	0
XGXS Identifier Registers				
4.2:[15:0]	PHY XS Identifier	MSB = 0x0000	R	0
4.3:[15:0]	PHY XS Identifier	LSB = 0x0004	R	0004
XGXS Reserved Register				
4.4.[15:1]	Reserved	Value always 0	R	0
4.4.0	10 G Capable	Value always 1	R	1
Status 1 Register				
4.5.[15:6]	Reserved	Value always 0	R	0
4.5.5	DTE XS Present	Value always 0	R	0
4.5.4	PHY XS Present	Value always 1	R	1
4.5.3	PCS Present	Value always 0	R	0
4.5.2	WIS Present	Value always 0	R	0
4.5.1	PMD/PMA Present	Value always 0	R	0
4.5.0	Clause 22 regs present	Value always 0	R	0
XGXS Reserved Register				
4.6.15	Vendor specific device present	Value always 0	R	0
4.6.[14:0]	Reserved	Value always 0	R	0
4.8.[15:14]	Device present	10 = Device responding to this address	R	10
4.8.[13:12]	Reserved	Value always 0	R	0
4.8.11	Transmit Fault (not supported)	0 = No fault of tx path	R	0
4.8.10	Receive Fault (not supported)	0 = No fault of tx path	R	0
4.8.9:0	Reserved	Value always 0	R	0
4.15, 4.14	Package Identifier	Value always 0	R	0
4.24.[15:13]	Reserved	Value always 0	R	0
4.24.12	PHY XGXS Lane Alignment (not supported)	TX alignment status is available in the PCS core register	R	0 0
4.24.11	Pattern Testing ability	0 = Not able to generate pattern.	R	0
4.24.10	PHY XGXS has loop back capability	1 = Has loop back capability	R	1
4.24.[9:4]	Reserved	Value always 0	R	0
4.24.3	Lane 3 Sync (not supported)	Status is available from the PCS core	R	0 0

Table 9. XAUI IP Core Registers (Continued)

Bit(s)	Name	Description	R/W	Reset Value
4.24.2	Lane 2 Sync (not supported)	Status is available from the PCS core	R	0 0
4.24.1	Lane 1 Sync (not supported)	Status is available from the PCS core	R	0 0
4.24.0	Lane 0 Sync (not supported)	Status is available from the PCS core	R	0 0
4.25.15:3	Reserved	Value always 0	R	0
4.25.2	Receive test pattern enable	0 = Receive test pattern not enabled	R	0
4.25.1:0	Test pattern select	00	R	00
4.8000.[15:0]	MCA Sync Request	Writing any value to this register triggers the MCA sync request. This register always read as 0.	R/W	0
4.8001.15	MCA Sync Status	This bit provides MCA synchronization status.	R/W	0
4.8002.[15:8]	Unused	Bit [15:8] are unused	R/W	0
4.8002.[7:0]	GPO	This field controls the General Purpose Outputs (GPO) when the MDIO is enabled. It is intended to control optional ports of the PCS core.	R/W	0
4.8003.[15:8]	Unused	Bit [15:8] are unused	R/W	0
4.8003.[7:0]	GPI	This field senses the state of the General Purpose Inputs (GPI) when the MDIO is enabled. It is intended to sense the status control of the PCS core.	R/W	0

Table 10. XAUI Vendor Specific Registers 4.9xxxh

Bit(s)	Name	Description	R/W	Reset Value
4.9xxxx.[15:8]	Unused	Bit [15:8] are unused	R/W	0
4.9xxxx.[7:0]	PCS register access	Bit [7:0] are directly mapped to sci_data port. Address [5:0] are directly mapped to sci_address port [5:0]. Address bit [8:6] are used to decode which SCI quad is being selected. [8:6] = 0 SCI0 is selected [8:6] = 1 SCI1 is selected [8:6] = 2 SCI2 is selected [8:6] = 3 SCI3 is selected [8:6] = 4 SCI AUX is selected	R/W	0

Input/Output Timing

XGMII Specifications

Clause 46 if IEEE 802.3ae specifies HSTL1 I/O with a 1.5V output buffer supply voltage for all XGMII signals. The HSTL1 specifications comply with EIA/JEDEC standard EIA/JESD8-6 using Class I output buffers with output impedance greater than 38¾ to ensure acceptable overshoot and undershoot performance in an unterminated interconnection. The parametric values for HSTL XGII signals are given in Table 11. The HSTL termination scheme is shown in Figure 11. Timing requirements for chip-to-chip XGMII signals are shown in Figure 12.

Table 11. XGMII DC and AC Specifications

Parameter	Condition	Min.	Typ.	Max.	Units
VDDIO	-	1.4	1.5	1.8	V
VREF	-	0.68	0.75	0.9	V
VTT1	-	-	0.75	-	V
VIH	-	VREF + 100mV	0.85	VDDIO + 0.3	V
VIL	-	-0.3	0.65	VREF - 100mV	V
VOH2	IOH > 8mA	1	1.1	-	V
VOL	IOL > -8mA	-	-	0.4	V

Figure 11. HSTL1 Circuit Topology

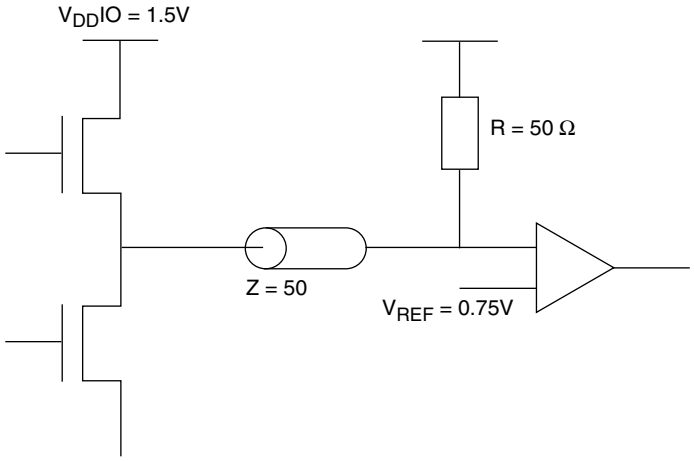
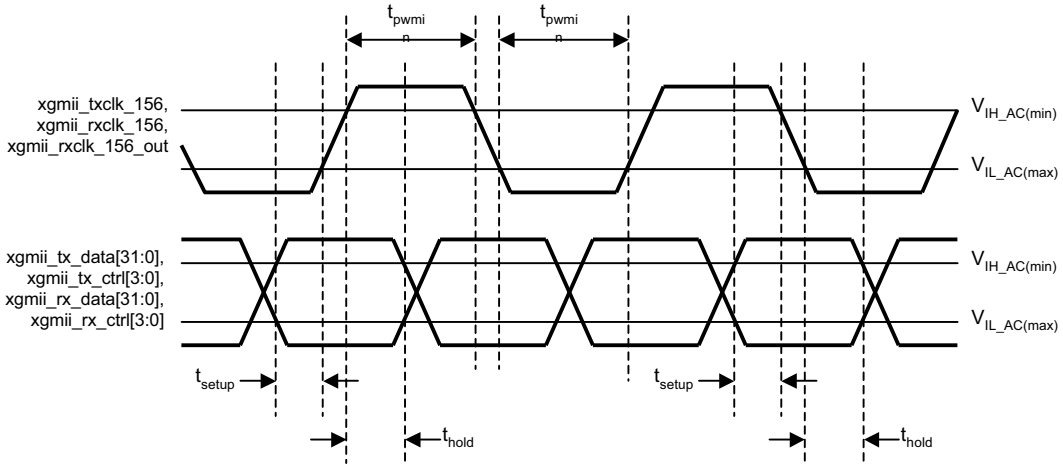


Figure 12. XGMII Timing Parameters



Symbol	Driver	Receiver	Units
t_{SETUP}	960	480	ps
t_{HOLD}	960	480	ps
t_{PWMIN}	2.5	-	ns

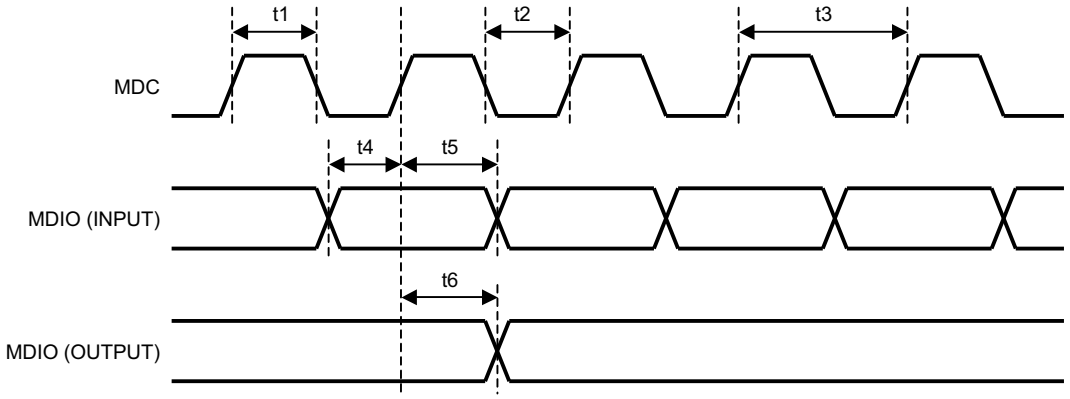
XAUI Specifications

Refer to the LatticeECP2M and LatticeECP3 PCS specifications for a complete XAUI timing requirements.

MDIO Specifications

The electrical specifications of the MDIO signals conform to Clause 45.4 of IEEE 802.3ae.

Figure 13. MDIO Timing



Symbol	Description	Min.	Max.	Units
t1	MDC high pulse width	200	—	ns
t2	MDC low pulse width	200	—	ns
t3	MDC period	400	—	ns
t4	MDIO (I) setup to MDC rising edge	10	—	ns
t5	MDIO (O) hold time from MDC rising edge	10	—	ns
t6	MDIO (O) valid from MDC rising edge	0	300	ns

Core Generation

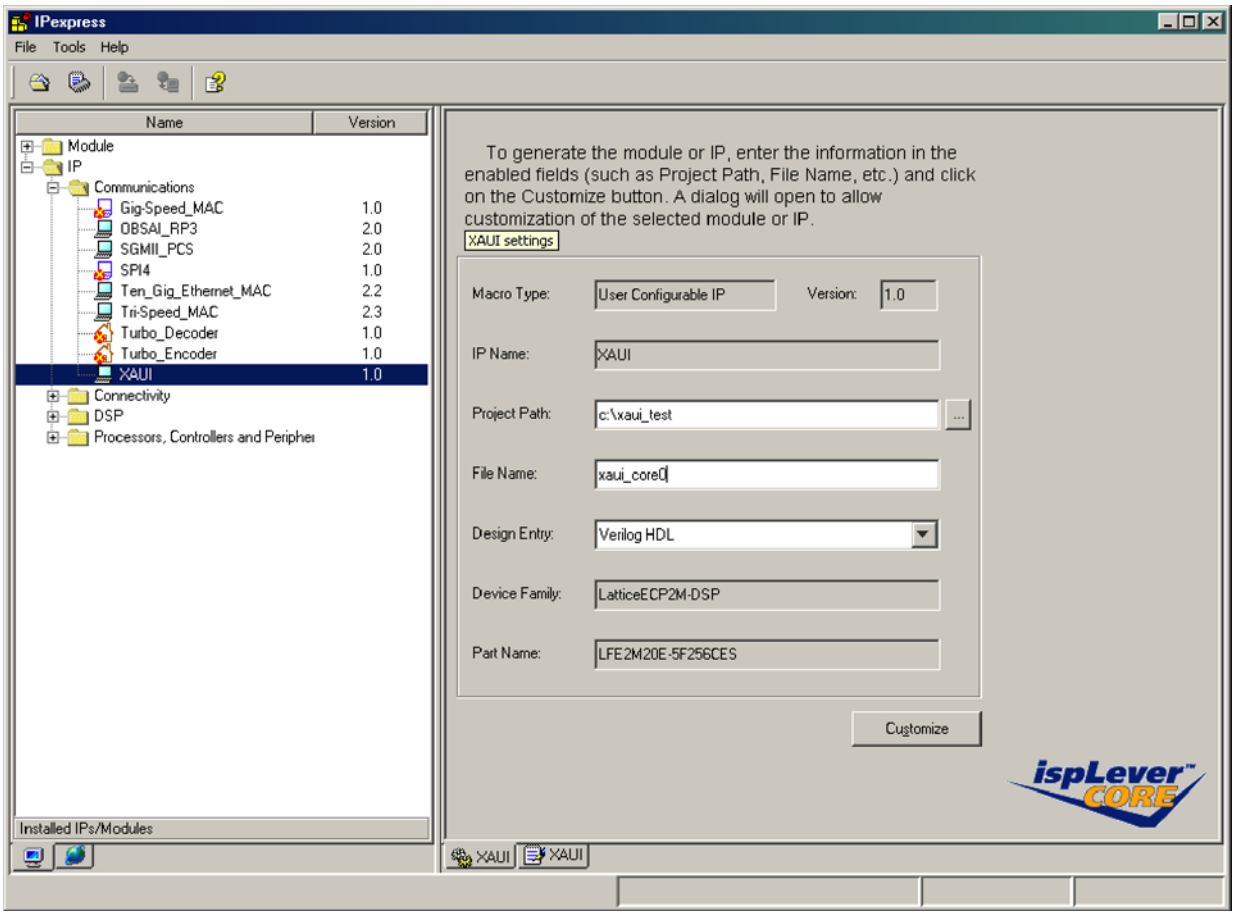
The XAUI IP is available for download from the Lattice website at www.latticesemi.com/ip. The IP files are automatically installed using ispUPDATE technology in any user-specified directory.

The ispLEVER® IPexpress™ GUI window is shown in Figure 14. To generate a specific IP core configuration the user specifies:

- Project Path - The path to directory where the generated IP files will be loaded.
- File Name - "username" designation given to the generated IP core and corresponding folders and files.
- Design Entry Type - Verilog.
- Device Family - Device family to which IP is to be targeted (e.g., LatticeECP2M). Only families that support the particular core are listed.
- Part Name - Specific targeted part within the selected device family.

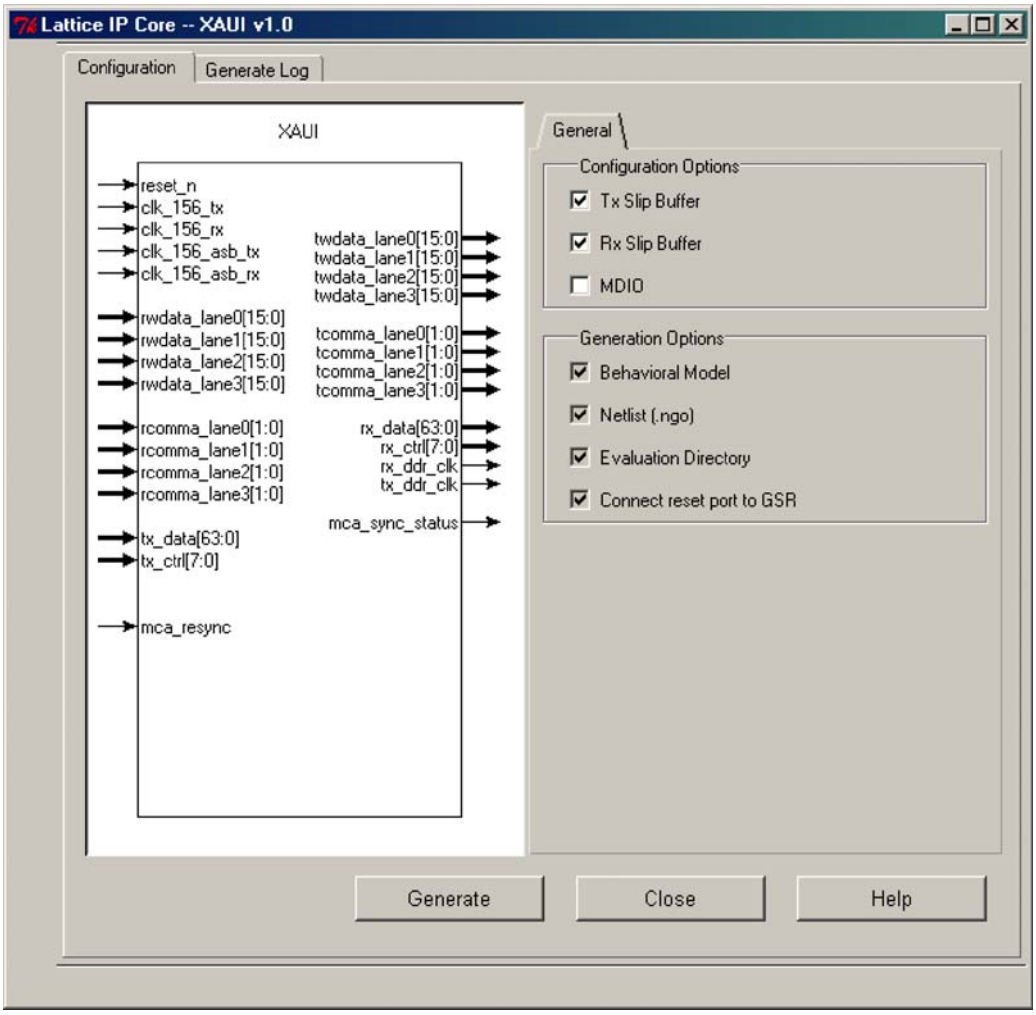
Note that if IPexpress is called from within an existing project, Project Path, Design Entry, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress on-line help for further information.

Figure 14. IPexpress GUI Window



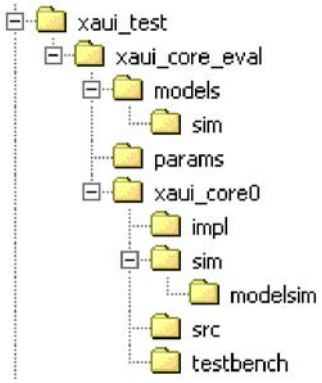
To create a custom configuration, click on the **Customize** button to display the XAUI IP core Configuration GUI, shown in Figure 15. From this window the user may select the appropriate parameters.

Figure 15. XAUI IP Configuration GUI Window



When the user clicks the **Generate** button, the configuration-specific IP core and supporting files are generated in the user's project directory. The directory structure of the generated files is shown in Figure 16.

Figure 16. XAUI IP Core Generated Directory Structure



The following files are generated in the user's project directory (\xaui_test in Figure 16):

- <username>.lpc - IP parameter file (may be directly modified by user).
- <username>.ngo - synthesized and mapped IP core.
- <username>_bb.v - black box module wrapper for synthesis.
- <username>_inst.v - example of instantiation template to be included in user's design.
- <username>_beh.v - behavioral simulation model for IP core configuration username.

These are all of the files needed by the user to implement and verify the XAUI IP core in their top-level design. The following additional files providing IP core generation status information and command line generation capability are generated in the user's project directory:

- <username>_generate.tcl - created when GUI "Generate" button is pushed, invokes generation, may be run from command line.
- <username>_generate.log - ispLEVER synthesis and map log file.
- <username>_gen.log - IPexpress IP generation log file.

The \xaui_core_eval and subtending directories provide files supporting XAUI core evaluation. The \xaui_core_eval directory shown in Figure 16 contains files/folders with content that are constant for all configurations of the XAUI core. The \<username> subfolder (\xaui_core0 in this example) contains files/folders with content specific to the username configuration.

The \xaui_core_eval directory is created by IPexpress the first time the core is generated and updated each time the core is regenerated. A \<username> directory is created by IPexpress each time the core is generated and regenerated each time the core with the same file name is regenerated. A separate \<username> directory is generated for cores with different names, e.g. \core1, \core2, etc.

Instantiating the Core

The generated XAUI IP core is provided in Lattice's proprietary .ngo format, which is independent of the HDL used to capture the rest of the user's design. The generated XAUI IP core package includes black-box (<username>_bb.v) and instance (<username>_inst.v) templates that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file is provided in \<project_dir>\xaui_core_eval\<username>\src. Customers may use this top-level reference as the starting template for the top-level for their complete design.

Running Functional Simulation

The functional simulation includes a configuration-specific behavioral model of the XAUI core (<username>_beh.v) that is instantiated in an FPGA top-level along with DDR IO logic for the XGMII interface. This FPGA top is instantiated in an eval testbench that configures the XAUI core registers. The testbench test pattern generator sources data to the XAUI core via the XGMII interface. The output of the PCS core is then serially loopback to the XAUI core. The receive data at the XGMII interface is then checked against the expected data.

Users may run the eval simulation by doing the following:

1. Open ModelSim®.

2. Under the **File** tab, select **Change Directory** and choose folder

   \<project_dir>\xaui_core_eval\<username>\sim\modelsim.

3. Under the **Tools** tab, select **Execute Macro** and execute one of the ModelSim "do" scripts shown.

The simulation waveform results are displayed in the ModelSim Wave window.

Synthesizing and Implementing the Core in a Top-Level Design

As mentioned previously, the XAUI IP core itself is synthesized and is provided in NGO format when the core is generated. Users may include the core in their own top-level design by instantiating the core in their top-level and then synthesizing the entire design with either Synplify® or Precision® RTL Synthesis.

The following steps are used in the implementation phase of the design process:

- Copy and paste the instance template (<username>_inst.v) into the user netlist where the XAUI IP core resides (not limited to user's top level).
- Edit the connection list as necessary to connect the XAUI IP core to the rest of the user design.
- Run the synthesis tool, making sure the component definition file (<username>_bb.v) is included in the list of files to be compiled. The resulting netlist will contain a black box instantiation of the XAUI IP core.
- When running map, place, and route, make sure the <username>.ngo and associated pmi_*.ngo memory files can be found by the ispLEVER software. This is accomplished either by copying the <username>.ngo file to the place-and-route working directory, or pointing to the directory where it resides.

Implementation Evaluation

An example RTL top-level reference source file supporting XAUI core top-level synthesis and implementation is provided with the IP core in \<project_dir>\xaui_core_eval\<username>\src\rtl\top.

Push-button implementation of this design is supported via an ispLEVER project file <username>_eval.syn located in \<project_dir>\xaui_core_eval\<username>\impl. To use this project file:

1. Select **Open Project** under the **File** tab in ispLEVER.
2. Browse to \<project_dir>\xaui_core_eval\<username>\impl in the Open Project dialog box.
3. Select and open <username>_eval.syn. A this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the device top-level entry in the left-hand GUI window.
5. Implement the complete design via the standard ispLEVER GUI flow.

Hardware Evaluation

Lattice's IP hardware evaluation capability makes it possible to create versions of IP cores that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase on an IP license. The hardware evaluation capability is turned on by enabling the Hardware Evaluation option in the properties of the Build Database process in ispLEVER. When the Hardware Evaluation option is enabled it is possible to generate a programming file that may be downloaded into the device. After initialization, the IP core will be operational for approximately four hours. After four hours, the IP core will stop working and it will be necessary to reprogram the device to re-enable operation. This hardware evaluation capability is only enabled if the core has not been licensed. If a license is detected, core generation is completed with no restrictions.

References

- FPGA Design with ispLEVER Tutorial
- ispLeverCORE IP Module Evaluation Tutorial

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
June 2007	01.0	Initial release.
June 2008	01.1	Title changed from "10Gb Ethernet Attachment Unit Interface (XAUI) IP Core User's Guide" to "XAUI IP Core User's Guide".
		Updated Appendix for LatticeECP2M FPGAs.
June 2009	01.2	Added support for LatticeECP3 FPGA family.
November 2009	01.3	Updated utilization tables with ispLEVER 8.0.

Appendix for LatticeECP2M/S FPGAs

Table 12. Performance and Resource Utilization[1]

Configuration				Utilization			
Device	TX Slip Buffer	RX Slip Buffer	MDIO	SLICEs	LUTs	Registers	EBRs
LFE2M20E-6F256CES	No	No	No	1276	1693	1509	0
LFE2M20E-6F256CES	No	No	Yes	1418	1895	1677	0
LFE2M35E-6F484CES	No	Yes	No	1679	2151	2057	2
LFE2M50E-6F672CES	Yes	No	No	1671	2129	2052	2
LFE2M70E-6F900CES	Yes	Yes	Yes	2207	2773	2770	4

1. Performance and utilization characteristics are in Lattice ispLEVER® 8.0 software. When using this IP core in a different density, speed, or grade within the LatticeECP2M/S family or in a different software version,þperformance and utilization may vary.

Ordering Part Number

The Ordering Part Number (OPN) for all configurations of the XAUI IP core targeting LatticeECP2M/S devices is XAUI-PM-U1.

IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and ispLEVER help system. For more information on the ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

Jitter and XAUI Compliance

The Lattice XAUI IP core for the LatticeECP2M device family offers compatibility with XAUI solutions and is functionally compliant to XAUI standard specification IEEE 802.3AE. XAUI specification limits total jitter to 112ps (0.35UI at 320 picoseconds/Unit Interval). LatticeECP2M exhibits transmit jitter within XAUI specification under typical conditions. Worst case transmit jitter considered over all temperature, voltage and process corners may fall outside XAUI specification. However, the LatticeECP2M based XAUI solution has been found to be compatible with many systems. See TN1084, LatticeSC SERDES Jitter, for applicable jitter specification.

XAUI specification requires the receive side to have the ability to operate with 208ps (0.65UI) of jitter. The Lattice solution for XAUI receive operates well within that specification.

Appendix for LatticeECP3 FPGAs

Table 13. Performance and Resource Utilization[1]

Configuration				Utilization			
Device	TX Slip Buffer	RX Slip Buffer	MDIO	SLICEs	LUTs	Registers	EBRs
LFE3-35E-7FN484CES	No	No	No	1187	1694	1497	0
LFE3-70E-7FN672CES	No	Yes	No	1552	2086	2037	2
LFE3-150E-7 FN1156CES	Yes	Yes	No	1936	2523	2581	4

1. Resource utilization characteristics are in Lattice ispLEVER 8.0 software with Synplify synthesis. When using this IP core in a different density, speed, or grade within the LatticeECP3 family or in a different software version, resource utilization may vary.

Ordering Part Number

The Ordering Part Number (OPN) for the XAUI IP core targeting LatticeECP3 devices is XAUI-E3-U1.

IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and ispLEVER help system. For more information on the ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.