



LatticeECP3 XAUI Demo Design

User's Guide

Introduction

This document provides technical information and instructions on using the LatticeECP3™ XAUI Demo Design. The demo has been designed to demonstrate the performance of the LatticeECP3 PCS in XAUI mode at 3.125 Gbps. The document provides a circuit description as well as instructions for running the demo on the LatticeECP3 Serial Protocol Board.

In addition to this user's guide the XAUI Demo comes with the following:

- Verilog source code for the FPGA design
- ispLEVER® Project Navigator implementation Project file and Aldec® Active-HDL® simulation script
- Bitstream (in format of *.bit)
- ORCAstra Plug-in GUI

Hardware requirement for the XAUI Demo Design:

- LatticeECP3 Serial Protocol Board (Revision D or newer) with LatticeECP3-95, 1156-pin device
- 12V DC power supply
- PC with ORCAstra - PC not provided
- Optional Clock Generator instrument to provide differential 156.25 MHz external reference clock to the PCS/SERDES QUAD – Instrument not provided
- 8 SMA cables for PCS loopback
- ispVM™ JTAG to USB download cable

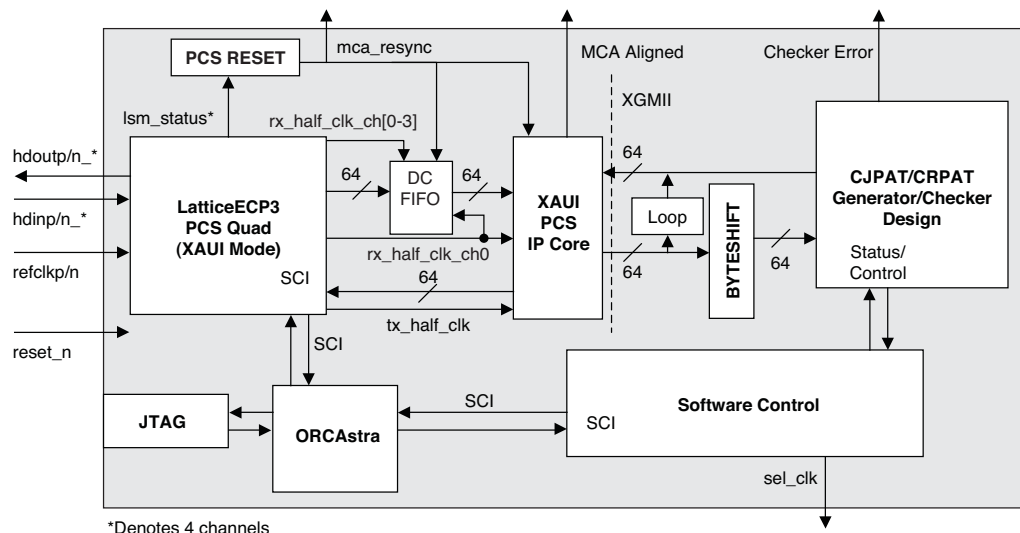
Software application and driver requirements include:

- ispVM System software (version 17.4 or later) for FPGA bitstream download
- ispLEVER design software version 8.1 or later
- ORCAstra software for user control interface – Included with ispLEVER 8.1 or later

XAUI Demo Design Overview

A block diagram of the XAUI Demo design is provided in Figure 1.

Figure 1. XAUI Demo Design



The XAUI Demo Design is a quad-based XAUI generator/checker that transmits/checks XGMII data (CJPAT or CRPAT) to/from a PCS quad. In turn, the PCS quad serializes the data in the transmit direction, and de-serializes it in the receive direction. For this demo the encoded serial data stream is looped-back externally via cables using the on-board SMA connectors. In this case, a backplane of variable length can be included in the serial path, or the Quad outputs can be connected to its inputs.

For this demo, the PCS QUAD B location on the chip is used. It has been generated in XAUI mode, at 3.125 Gbps per channel. The following paragraphs describe the main blocks used in the demo. In addition to these blocks, the reference design implements the LatticeECP3 PCS TX and RX Reset State machines described under the SERDES/PCS RESET section of TN1176, [LatticeECP3 SERDES/PCS Usage Guide](#).

LatticeECP3 XAUI PCS

PCS quad B is clocked by its designated differential external reference clock input, `refclkp/n`. The reference clock's frequency is 156.25 MHz and has two potential sources: The on-board Y1 differential oscillator or SMA differential inputs J29 (P) and J33 (N) to the board. The clock selection is controlled by an output signal (`sel_clk`) which is controlled by an internal user register bit. Internally to the PCS, the reference clock is multiplied by a factor of 20 to generate the 3.125 Gbps per channel data rate. At the PCS/FPGA interface, a 16-bit data interface is used. This requires 156.25 MHz receive and transmit clocks. The PCS-generated `rx_half_clk` and `tx_half_clk` (see Figure 1) clock the PCS interface RX and TX data respectively into the PCS XAUI IP.

PCS RESET Logic

The PCS RESET logic monitors low pulses on `lsm_status*` and issues a reset to the DC FIFO as well as an `MCA_RESYNC` to the XAUI soft IP when any of these low pulses occur. This helps reset the DC FIFO pointers as well as the XAUI PCS IP multi-channel alignment logic upon any disturbance of the SERDES serial input data.

DC FIFO

Each of the individual recovered `rx_half_clk_ch[0-3]` clocks the corresponding PCS channel's RX parallel data into the DC FIFO. On the output side of the DC FIFO, the data from all four channels are combined and clocked out by a single recovered clock (`rx_half_clk_ch0`) into the PCS XAUI IP core. The PCS RESET block resets the read and write pointers of the DC FIFO once all clocks have stabilized.

XAUI PCS Soft IP

The 10Gb Ethernet Attachment Unit Interface (XAUI) IP provides a solution for bridging between XAUI and 10-Gigabit Media Independent Interface (XGMII) devices. This IP implements 10 GB Ethernet Extended Sublayer (XGXS) capabilities in soft logic that together with PCS and SERDES functions implemented in the LatticeECP3 FPGA provides a complete XAUI-to-XGMII solution.

The LatticeECP3 XAUI soft IP implements:

- Multi-channel alignment, Clock Tolerance Compensation (CTC) and 8b10b->XGMII data translation in RX direction. Indicates proper MCA alignment via `MCA_ALIGNED`.
- XGMII-> 8b10b data translation in the TX direction.

For more information on the XAUI PCS IP, please refer to IPUG68, [XAUI IP Core User's Guide](#).

XGMII Loopback Logic

This logic is controlled by an internal user register bit. The logic allows the XGMII RX data to be looped-back in the TX direction. This allows the XAUI PCS reference design to act as a DUT type #2 device as described by the IEEE 802.3ae-2002 specification.

CJPAT/CRPAT Generator/Checker Quad

The XAUI Generator/Checker Quad block in the demo design has the following characteristics:

- 4 channels form an XGMII bus
- Data bus is 64-bit (+ 8 bits of control)
- 1 XAUI Generator (CJPAT or CRPAT) transmitting to all 4 TXD channels
- 1 XAUI Checker (CJPAT or CRPAT) checking all 4 RXD channels
- Control/Status interface to user registers
- 1 Error Counter (CJPAT or CRPAT) connected to user registers for monitoring
- 1 latched Checker error signal output to indicate that one or more errors have occurred within a single frame. The latched checker error signal goes low upon the reception of a new frame.

The ORCAstra block controls all user registers as well as the LatticeECP3 PCS Quad via the SCI interface. ORCAstra is in turn controlled via the JTAG interface. See Figure 1.

BYTESHIFT Logic

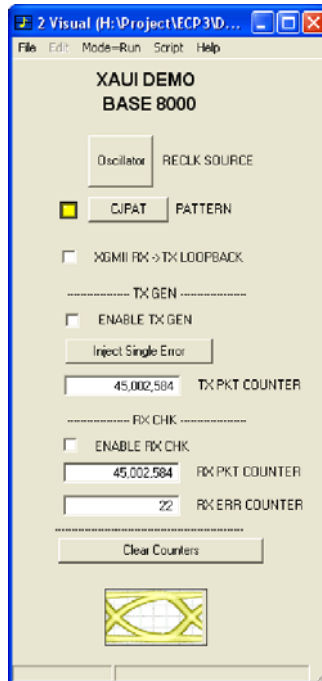
The CJPAT/CRPAT TX Generator will transmit data with the SOP (F) character at the lowest byte position. The CJPAT/CRPAT RX checker also expects the same character to appear at the lowest byte position. Due to the fact that the LatticeECP3 XAUI PCS is configured in 16-bit bytes, the 16-bit word alignment out of the PCS RX FPGA interface may not be such that SOP is in the lowest byte position. The byte shift logic corrects for this by monitoring the occurrence of the SOP (FB) character on lane 0 and shifting data on all 4 lanes if SOP is in the upper byte position of the 16-bit RX data on lane 0 of the LatticeECP3 PCS. For more information on 16-bit word alignment, please refer to the “16/20-bit Word Alignment” section of TN1176, [LatticeECP3 SERDES/PCS Usage Guide](#).

XAUI Generator/Checker ORCAstra GUI

This demo utilizes a visual window as a plug-in to the base ORCAstra installation. The visual window is associated with logical LatticeECP3 PCS QUAD 0. To open the visual windows, complete the following steps.

1. Open ORCAstra and select **CustomProgrammability > Visual Window**. An empty Custom Visual Interface window will appear.
2. In the Custom Visual Interface window, select **File > Open**. Browse to the directory where you have installed the XAUI Demo folder, then to ORCAstra Plug-ins. Select **8000\EyeDemo.vis**, and click **Open**. This will open the visual window plug-in for XAUI Demo. Figure 2 provides a screen capture of this window.

At this point, make sure that the Continuous Polling check box is selected in the main ORCAstra window.

Figure 2. XAUI Generator/Checker ORCAstra Plug-in Visual Window

The following is a description of the control and status elements for the visual window.

REFCLK SOURCE: This button allows the user to select either the internal differential oscillator or the external SMA differential inputs as the source of the refclkp/n to the PCS SERDES.

XGMII RX->TX LOOPBACK: The logic allows the XGMII RX data at the XAUI PCS soft IP interface to be looped-back in the TX direction. This allows the XAUI PCS reference design to act as a DUT type #2 device as described by the IEEE 802.3ae-2002 specification.

PATTERN: This button allows the user to select either CRPAT or CJPAT. There is also a yellow LED to the left of the button that glows when CJPAT is selected.

ENABLE TX: When checked, it enables the transmission of XAUI (CJPAT or CRPAT depending on what the PATTERN button is set to) packets from the generator.

ENABLE RX: When checked, it enables the detection of errors (CJPAT or CRPAT depending on how the PATTERN button is set) by the checker of the XAUI quad.

TX PKT COUNTER: The TX PACKET counter keeps count of the number of packets (CJPAT or CRPAT depending on how the PATTERN button is set) transmitted in the TX XGMII direction. The counter will count up as long as ENABLE TX is checked. Clicking on the Clear Counters button resets this counter. The counter is software-based. ORCAstra periodically loads a snapshot of the TX PKT counter (tx_pkt_counter_snap) into a software accumulator. The software accumulator value is displayed in the counter field.

RX PKT COUNTER: The RX PACKET counter keeps count of the number of packets (CJPAT or CRPAT depending on what the PATTERN button is set to) received by the checker in the RX XGMII direction. The counter will count up as long as ENABLE RX is checked and valid packets (with proper SOP and EOP) are received. Clicking on the Clear Counters button resets this counter. The counter is software-based. ORCAstra periodically loads a snapshot of the RX PKT counter (rx_pkt_counter_snap) into a software accumulator. The software accumulator value is displayed in the counter field.

RX ERR COUNTER: The RX ERR counter keeps count of the number of packets (CJPAT or CRPAT depending on how the PATTERN button is set) with one or more incorrect bytes received by the checker in the RX XGMII direction. The counter will count up as long as ENABLE RX is checked and valid packets (with proper SOP and EOP) are received. Clicking on the Clear Counters button resets this counter. The counter is software-based. ORCAstra periodically loads a snapshot of the RX ERROR counter (rx_error_counter_snap) into a software accumulator. The software accumulator value is displayed in the counter field.

Clear Counters: When checked, it asynchronously clears the content of the TX PKT, RX PKT, and RX ERR counters.

Inject Single Error: The TX generator inserts a single 64-byte data error when the Inject Single Error button is pushed. This will happen as long as ENABLE TX is checked. The incorrect DATA byte is inserted on the next CJPAT/CRPAT frame after the Inject Single Error is pushed.

XAUI Generator/Checker User Register Map

The user registers for the XAUI generator/checker quad are defined in Table 1. All register addresses are in hexadecimal. Also note that register address h00800 (not shown in Table 1) is a read-only register that contains the version number of the design.

Table 1. User Registers Map

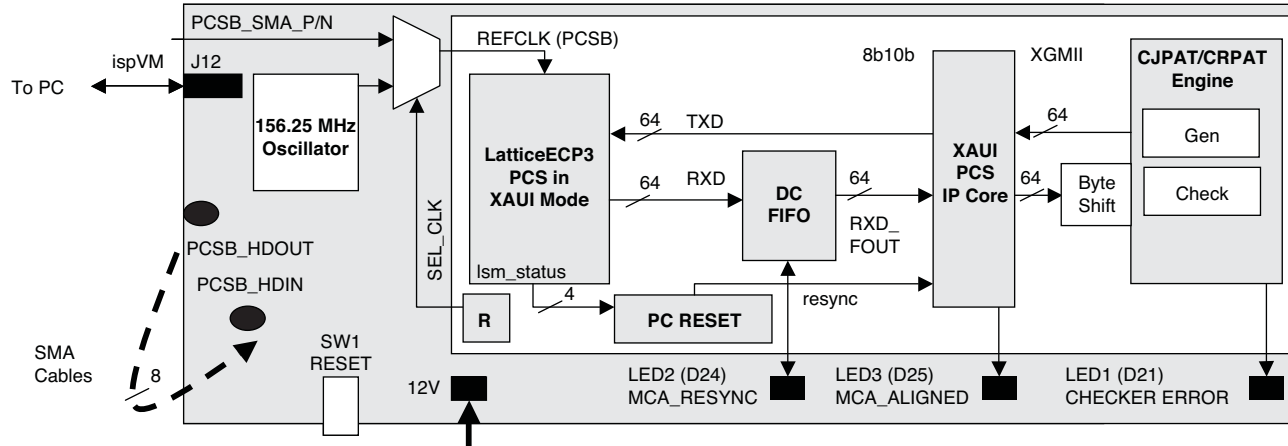
GUI Option	FPGA Registers	Description
Patterns	0x08000, bit 0	0 = CRPAT 1 = CJPAT
Enable TX	0x08000, bit 1	0 = Disable 1 = Enable
Inject Single Error	0x08000, bit 2	0 = No Action 0 to 1 = Inject 1 Error
Clear Counters	0x08000, bit 3	0 = No Action 1 = Clear
Enable RX	0x08000, bit 4	0 = Disable 1 = Enable
REFLCK SOURCE	0x08000, bit 5	0 = Oscillator 1 = External
XGMII Loopback	0x08000, bit 6	0 = No loopback 1 = Loopback
TX Packet Counter	tx_pkt_counter_snap[63:0]={0x08004[7:0], 0x08005[7:0], 0x08006[7:0], 0x08007[7:0], 0x08008[7:0], 0x08009[7:0], 0x0800A[7:0], 0x0800B[7:0]}	A snapshot of TX PKTCOUNT is taken into this counter when 8001 is read. This counter rolls over.
RX Packet Counter	rx_pkt_counter_snap[63:0]={0x0800C[7:0], 0x0800D[7:0], 0x0800E[7:0], 0x0800F[7:0], 0x08010[7:0], 0x08011[7:0], 0x08012[7:0], 0x08013[7:0]}	A snapshot of TX PKTCOUNT is taken into this counter when 8002 is read. This counter rolls over.
RX Error Counter	rx_error_counter_snap[63:0]={0x08014[7:0], 0x08015[7:0], 0x08016[7:0], 0x08017[7:0], 0x08018[7:0], 0x08019[7:0], 0x0801A[7:0], 0x0801B[7:0]}	A snapshot of TX PKTCOUNT is taken into this counter when 8003 is read. This counter rolls over.

LatticeECP3 Serial Protocol Board (Version D or Newer) Setup

The XAUI Demo design transmits XAUI data in the TX direction, and loops the serial high-speed data back to the LatticeECP3 and checks the data. Figure 3 shows the demo setup. The setup assumes the following:

1. ispVM is installed on a PC.
2. ORCAstra is installed on a PC.
3. Optional 156,25 MHz external CML differential clock source through SMA inputs

- ispVM download cable connected to the USB port of the PC and to the ispVM JTAG Connector on the board.
- Power is applied to the board via the provided power supply.

Figure 3. LatticeECP3 Serial Protocol Board Setup

XAUI Demo Design Signal Descriptions

Table 2 lists all the XAUI Demo Design signals that are connected on the LatticeECP3 Serial Protocol Board Version D or newer.

Table 2. Signal Descriptions

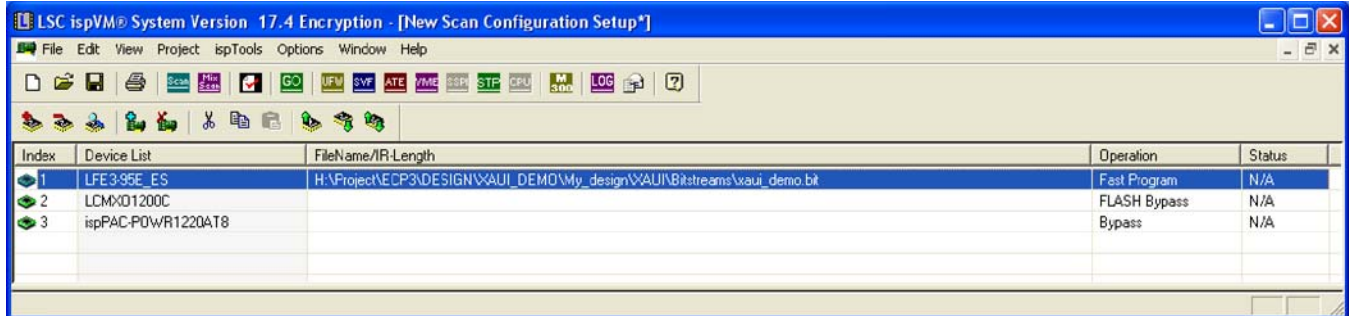
Signal Name	Signal Type	Board Connection	Description
General Signals			
reset_n	I	SW1 Push-button	FPGA global active low reset
Reference Design Signals			
mca_resync	O	D24 yellow LED	MCA_RESYNC indication
mca_aligned	O	D25 green LED	MCA ALIGNED status from XAUI IIP
RX checker error	O	D21 red LED	Latched checker error signal for current frame.
JTAG Signals			
tck	I	To J12 JTAG header	JTAG pins
tdi	I		
tdo	O		
tms	I		
PCS Quad			
PCSB_SMA_P/N	I	Sourced from J29 (P) and J33 (N)	Optional SMA differential input reference clock
hdinp_0/ hdinn_0	I	SMA J13 and J14	Channel 0 differential high-speed SERDES inputs
hdinp_1/ hdinn_1	I	SMA J15 and J16	Channel 1 differential high-speed SERDES inputs
hdinp_2/ hdinn_2	I	SMA J21 and J22	Channel 2 differential high-speed SERDES inputs
hdinp_3/ hdinn_3	I	SMA J23 and J24	Channel 3 differential high-speed SERDES inputs
hdoutp_0/ hdoutn_0	O	SMA J17 and J18	Channel 0 differential high-speed SERDES outputs
hdoutp_1/ hdoutn_1	O	SMA J19 and J20	Channel 1 differential high-speed SERDES outputs
hdoutp_2/ hdoutn_2	O	SMA J25 and J26	Channel 2 differential high-speed SERDES outputs
hdoutp_3/ hdoutn_3	O	SMA J27 and J38	Channel 3 differential high-speed SERDES outputs

Loading the LatticeECP3 XAUI Demo Bitstream with ispVM

Follow the instructions below to load the XAUI bitstream.

1. If ORCAstra is already loaded, make sure **Interface** is set to **None**.
2. Start ispVM version 17.4 (or later).
3. Click on the green **Scan** icon (see Figure 4).
4. Make sure devices 2 and 3 are in bypass as per Figure 4.
5. Make sure device 1 selects LFE3-95_ES or equivalent -95 device on the board.
6. Make sure the device 1 bitstream file name points to <your_project_path>\XAUI\Bitstreams\xaui_demo.bit and that **Operation** is set to **Fast Program**.
7. Click **Go**.

Figure 4. ispVM Setup



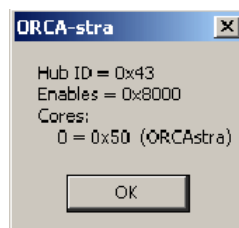
Running a Demo with the ORCAstra PCS View

This section describes the use of the ORCAstra GUI to interactively change/monitor the LatticeECP3 PCS and user registers.

Starting LatticeECP3 ORCAstra

Ensure the LatticeECP3 XAUI Demo bitstream has been loaded, then:

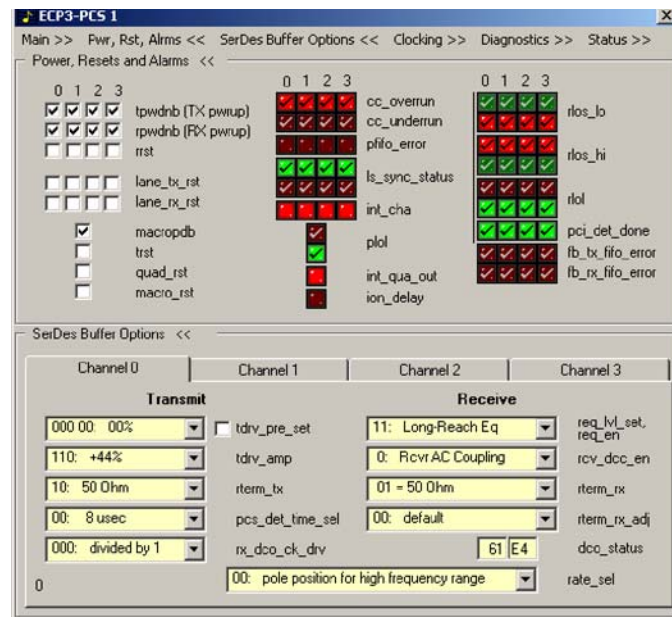
1. Start ORCAstra version 2.2.34 (or later).
2. Select **Interface > 1. ispVM JTAG Hub USB Interface**. If the **Select Target JTAG Device** window is displayed, select the first device and click **OK**.
3. You will see the following window. Click **OK**.



4. Select **Device > Lattice ECP3**. Also select **Options** and un-check the check box next to **Display Data in [7:0] Order in Data Box**.

- Click on the **ECP3 PCS1** option. You will see the ECP3-PCS1 Visual Window. Close the Main tab, and open the **Pwr, Rst, Alrms**, and **SerDes Buffer Options** tab. The resulting window is shown in Figure 5.
- From the main ORCAstra Visual Window, select **CustomProgrammability-> Visual Window**.
- In the new window, select **File > Open > <your_project_path>XAUI\ORCAstra Plug-ins\8000\Eyedemo.vis**. You will see the **XAUI Gen./Check Visual Window** shown in Figure 2.
- Make sure Continuous Polling is checked in the main ORCAstra window. The resulting GUIs are shown in Figure 5 for the ECP3-PCS1 Visual Window and in Figure 2 for the CJPAT/CRPAT generator/checker Visual Window.

Figure 5. LatticeECP3 PCS 1 ORCAstra View



Configuring PCS 1 XAUI Options in ORCAstra

Power, Reset and Alarms

The default Pwr, Resets and Alarms section contains the following important information:

- Green/red LEDs (one per channel) to indicate that the receive link state machines are synchronized (ls_sync_status). Green indicates successful synchronization.
- Green/red LEDs (one for whole quad) to indicate that the SERDES transmit PLL (plol) is locking. Green indicates successful lock.
- Green/red LEDs (one per channel) to indicate Receive CDR lock (rlos). Green indicates successful lock.

This view also allows the user to identify which channels (or the entire quad), are powered down or reset. This view also allows users to reset PCS digital logic (lane_tx_rst and lane_rx_rst), as well as SERDES logic (macro_rst) and the whole quad (quad_rst).

SerDes Buffer Options View

This view allows controlling the characteristics of output, input, and reference clock buffers: TX pre-emphasis, TX amplitude, RX equalization, TX and RX buffer termination and coupling.

Typical XAUI Demo Application

This section describes the demo application. Note that the smaller blocks (BYTESHIFT, RESYNC) operation is not described. In this setup:

- The FPGA XAUI Generator is used to transmit CJPAT/CRPAT data from the FPGA to the PCS XAUI IP.
- The PCS XAUI IP converts XGMII data to 8b10b based TXD data to the LatticeECP3 PCS.
- The PCS SERDES HDOUT* pins are connected to the SERDES HDIN* pins through SMA cables.
- The PCS RXD ports then feed the recovered 8b10b data to the XAUI PCS IP.
- The XAUI PCS IP performs multi-channel alignment, clock tolerance compensation and 8b10b to XGMII data conversion and hands the data over to the FPGA XAUI QUAD checker.

This application makes use of the XAUI Generator/Checker ORCAstra Plug-in Visual Window from Figure 2 and the SerDes Buffer Options section of the XAUI ORCAstra View from Figure 5. While looping XAUI data in external near end mode, the first window is used to verify error-free XAUI data is received, while the second window is used to tweak output and input buffer options. The following steps describe the demo sequence:

1. Make sure power is supplied to the evaluation board.
2. Make sure HDOUT* are looped back to HDIN*.
3. Load the LatticeECP3 XAUI Demo bitstream as previously described.
4. Start an ORCAstra session and load the LatticeECP3 PCS1 visual basic window and the XAUI Generator/Checker ORCAstra Plug-in Visual Window as previously described.
5. Make sure the Continuous Polling box is checked in the main window.
6. In the LatticeECP3 PCS1 visual basic window, in the Power, Resets and Alarms view, ls_sync_status and rlol LEDs should be solid green. Also verify that plol is solid green. If any of these indicators are red, then proceed to debugging these indicators as indicated in step 7F.
7. In the XAUI Generator/Checker QUAD Visual Window
 - A. Select **CJPAT** or **CRPAT** for **PATTERN**.
 - B. Check the **ENABLE RX** box. The checker is now ready to check CJPAT or CRPAT packets.
 - C. Clear all counters by pressing the **Clear Counters** button.
 - D. Check the **ENABLE TX** box. The generator is now transmitting CJPAT or CRPAT packets.
 - E. If the TX PKT and RX PKT counters start incrementing while the RX ERR COUNTER remains at zero, the XAUI checker is receiving error-free packets. Check the basic XAUI view for proper link state machine, x4 alignment and CTC behavior.
 - F. If the RX ERR COUNTER increments, then the XAUI checker is receiving packets with errors.
 - a. Verify that the Power, Reset and Alarm section of PCS 1 XAUI View does not show any PLL loss of lock (rlol), CDR loss of lock (rlol), or internal link state machines loss of sync (ls_sync_status).
 - b. A red plol indicates that the reference clock source to the TX PLL is not stable
 - c. A red rlol indicates incorrect activity on the HDIN* inputs. The input signals may be too attenuated by the medium. Also, if the data was transmitted from a different device, the reference clock transmitting the HDIN* input data may not be within the required +/-100 ppm of the local reference clock. The latter issue, though rarely the case, needs to be addressed.

- d. Try modifying some of the input and output buffer settings in the SerDes Buffer Options section of PCS 1 XAUI View (TX pre-emphasis, TX amplitude, RX equalization...) to address the rol issue.
- e. If the rol is green but the ls_sync_status* is red, this is an indication that the patterns received do not contain valid K28.5 commas or may contain 8b10b code violations.
- f. If rol, ls_sync_status are green, but the on-board D25 green LED is not lit, then this indicates that the patterns either do not contain valid /A/ alignment characters, or that the alignment characters are not occurring on all 4 lanes within the required interval of time defined by the IEEE 802.3ae specification.

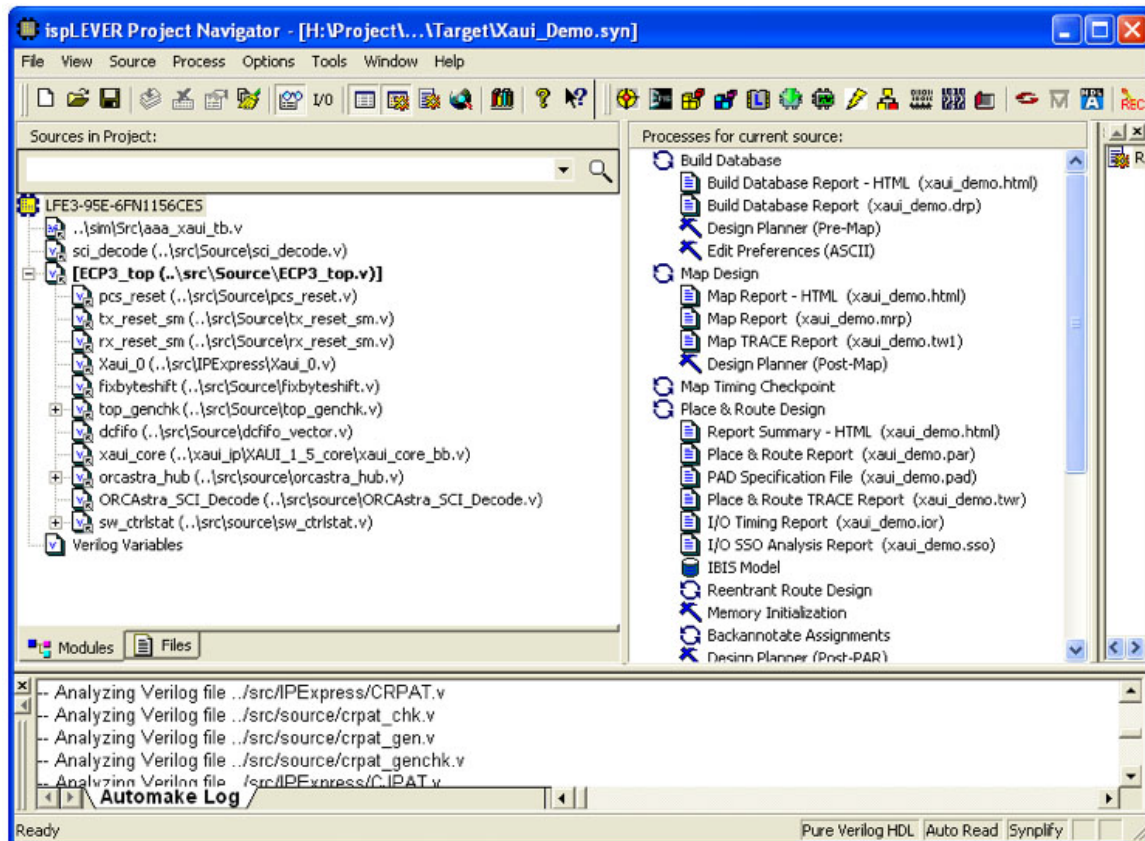
Implementing and Simulating Reference Design

The steps below explain how to run the XAUI reference design source code through ispLEVER Project Navigator MAP, place and route and bitstream generation, and simulation.

Both implementation and simulation start with the same steps:

1. Start ispLEVER Project Navigator
2. Open <your_project_path>XAUI\Target\Xaui_Demo.syn. This will load the Verilog-based project as shown in Figure 6.

Figure 6. Project Navigator



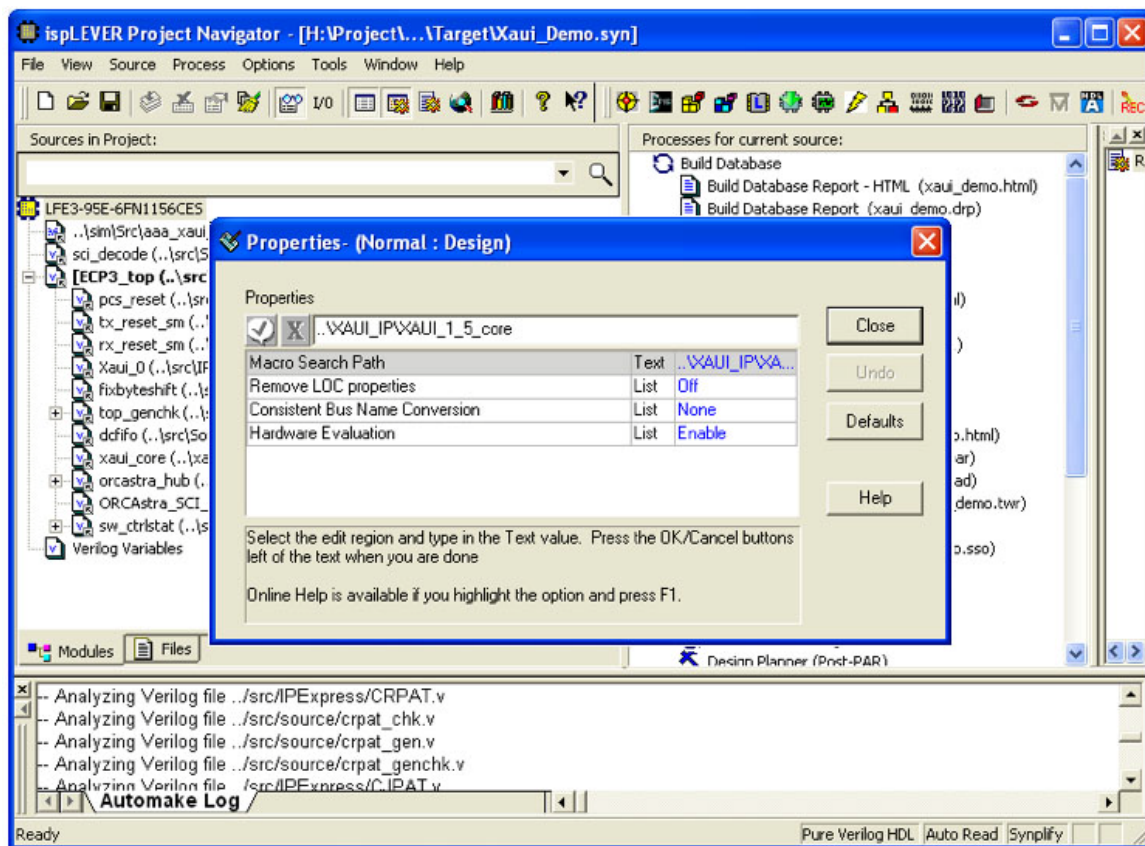
Implementation

The reference design uses an NGO file for the LatticeECP3 XAUI IP.

To ensure this NGO is used, follow the steps below as illustrated in Figure 7.

1. Select **Build Database** in the **Processes** window, then right-click and select **Properties**.
2. In the **Properties** windows, enter `..\XAUI_IP\XAUI_1_5_core`. This path points to the location of the XAUI IP core NGO.
3. Set the **Hardware Evaluation** value to **ENABLE (DISABLE)** to enable (disable) Lattice's IP hardware evaluation capability. 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 of 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.
4. Click on **Close**.

Figure 7. Build Database Properties



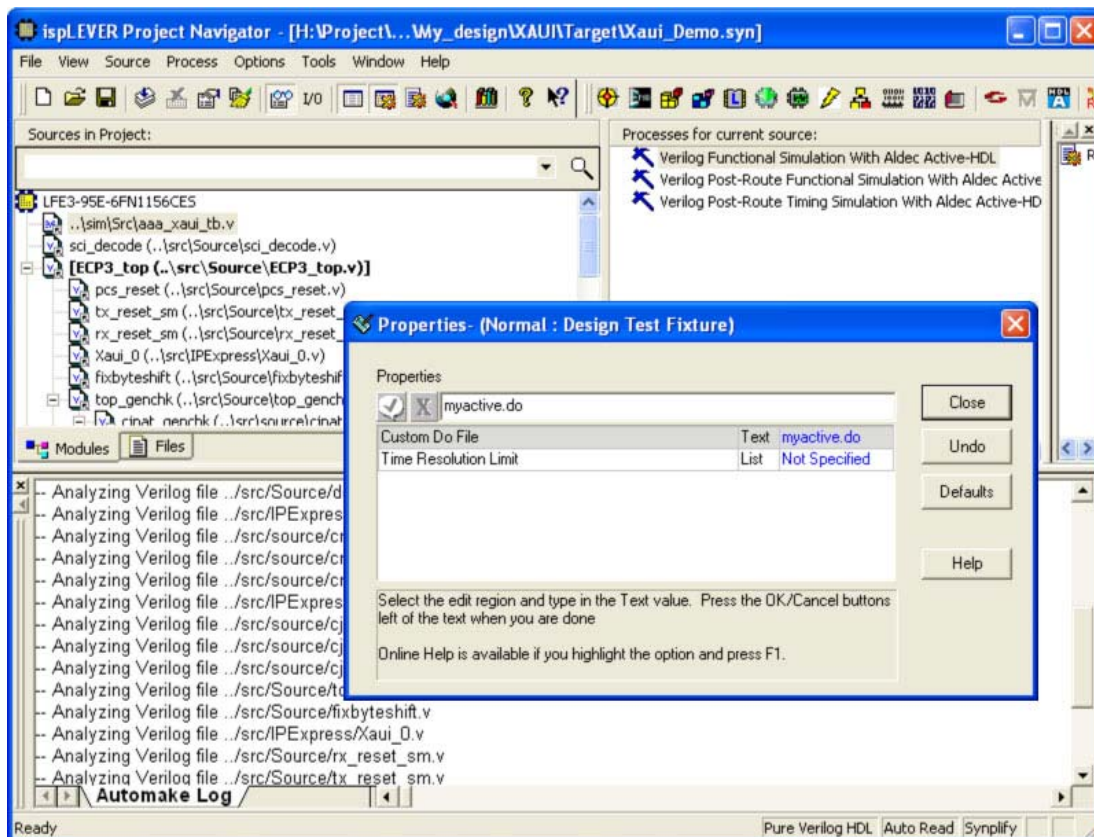
To implement the design and generate a bitstream, double-click the **Generate Bitstream Data** target in the **Processes** window. This will run through the full synthesis, place and route flow and generate a new `xaui_demo.bit` file. Note that the supplied version of the reference design meets timing with **Placement Iteration Start Pt** set to **2** in the **Place and Route Design** process properties in ispLEVER 8.1 SP1. Before using the bitstream, please verify that the Place and Route Timing Score is 0. If this is not the case, you will need to run multiple Place and Route iterations by changing the **Placement Iterations** value from the **Place and Route Design** process properties. Also set the **Auto Hold-Time Correction** to **ON** to correct hold violations during the Place and Route process.

Simulation

The simulation process uses the **myactive.do** Aldec Active-HDL simulation script located under the **Target** directory. To ensure this script is used, follow the steps below, as illustrated in Figure 8.

1. Select the **aaa_xaui_tb.v** testbench file in the **Source** window.
2. Select **Verilog Functional Simulation with Aldec Active-HDL** in the **Processes** window, then right-click and select **Properties**.
3. In the **Properties** window, enter **myactive.do** for the **Custom Do File** entry. Then click **Close**.
4. Double-click the **Verilog Functional Simulation with Aldec Active-HDL** target to start the Aldec-HDL simulation process, which will execute the **myactive.do** script. The script compiles all necessary design and testbench files and runs the simulation into the Aldec Waveform window shown in Figure 9.

Figure 8. Configuring Aldec Simulation



[illegible]

The following documents provide more information:

- ## Technical Support Assistance

Revision History

© 2011 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.