LatticeCORE

CPRI IP Core User's Guide

April 2014
IPUG56_02.4

Table of Contents

Introduction

This document provides technical information about the Lattice Common Public Radio Interface (CPRI) IP core. This IP core together with SERDES and Physical Coding Sublayer (PCS) functionality integrated in the LatticeECP3™ and ECP5™ FPGAs implements the physical layer of the CPRI specification and interleaves IQ data with synchronization, control and management information. It can be used to connect Radio Equipment Control (REC) and Radio Equipment (RE) modules.

Two CPRI core configurations are supported. The “basic” core configuration implements all of the capabilities required to support the physical layer of the CPRI specification, except specific requirements related to link delay accuracy. The “low latency” core configuration is equivalent to the basic configuration, but includes a modified SERDES/PCS interface that supports the ability to manage the variability in the absolute latency for data transmission through the core to meet the stringent CPRI link delay accuracy requirements.

The remainder of this document focuses on the detailed specifications associated with implementing and using the basic CPRI IP configuration. Areas of difference between the basic and low latency configurations are highlighted. Complete details on the implementation and use of the low latency configuration are included in IPUG74, *CPRI IP Core Low Latency Variation Design Considerations User’s Guide*.

The CPRI soft-core comes with the following documentation and files:

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

The CPRI 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 of 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.

In the following text, transmit refers to data flow from the user application logic to the CPRI link. Receive refers to data flow from the CPRI link to the user application logic. Downlink refers to the direction of data flow from REC to RE, and uplink refers to the direction of data flow from RE to REC.

The Lattice CPRI core is compliant with the version 3.0 CPRI specification. Note however that the core does not directly support requirement R-31 (line-rate autonegotiation). Lattice supports dynamic switching between full and half rate line settings (i.e., 614M/1.2G or 1.2/2.4G) but switching dynamically between all line rates is not supported since some PCS/SERDES bit settings need to be re-programmed through the SCI to support reliable data transfer. It is anticipated that in most network applications, line rate negotiation will be established/managed at the system level and there is nothing in the IP core that precludes supporting such capability.

Quick Facts

Table 1-1 gives quick facts about the CPRI IP core.

Table 1-1. CPRI IP Core Quick Facts

		CPRI IP Core	
		Across All IP Configurations	
Core Requirements	FPGA Families Supported	Lattice ECP3, ECP5	
	Minimal Device Supported	LFE3-35E-6FTN256C	LFE5UM-85F-7MG381C
Resource Utilization	Data Path Width	8-40 bits	
	LUTs	1400-1600	1600-2000
	sysMEM EBRs	2-6	
	Registers	1500-1700	1500-1700
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.2	
	Synthesis	Synopsys® Synplify Pro® for I-2013.09L-SP1 beta	
	Simulation	Aldec® Active-HDL™ 9.2 Lattice Edition	
		Mentor Graphics® ModelSim® 6.6E SE	

Features

The following features apply to the basic CPRI core configuration:

- Supports the physical link layer (Layer 1) of the CPRI specification
- Supports four standard bit rates of the CPRI specification
  - 614.4 Mbps
  - 1228.8 Mbps
  - 2457.6 Mbps
  - 3072 Mbps
- Supports 8b/10b encoding/decoding performed in the PCS/SERDES
- Supports code-violation detection performed in the PCS/SERDES
- Performs CPRI Hyperframe Framing
  - Performs interleaving of IQ data, sync, C&M data, and vendor specific information
  - Provides an 8-, 16-, or 32-bit parallel interface for IQ data
- Performs subchannel mapping:
  - Supports a slow C&M channel based on a serial HDLC interface at standard bit rates (240 Kbps, 480 Kbps, 960 Kbps, and 1920 Kbps). The HDLC framer, if needed, must be provided as a separate IP core.
  - Supports a fast C&M channel based on a serial Ethernet interface (84.48 Mbps max.) to the user logic, a non-standard rate MII Ethernet interface to a MAC, or a 100 Mbps MII interface to a PHY device. Accepts a user-selected pointer to the CPRI subchannel where the Ethernet link starts. The Ethernet MAC function is provided as a separate IP core.
- Performs synchronization and timing as defined in section 4.2.8 of the CPRI Specification
- Supports the L1 Inband Protocol
- Provides a parallel interface for merging vendor specific data into the CPRI frame
- Provides a start-up sequence state machine in hardware for both REC and RE nodes which performs:
  - Synchronization and Rate Negotiation
  - C&M Plane setup

- Performs Link Maintenance as defined in section 4.2.10 of the CPRI Specification:
  - LOS detection
  - LOF detection
  - RAI indication
- Optional top-level template that implements user registers for control and status management
- Optional 8-bit register interface through the JTAG port

The low latency CPRI core configuration supports all of the features specified for the basic core configuration with the following key exceptions/modifications:

- Supported for LatticeECP3 and ECP5 FPGA families only
- Supports 1228.8 Mbps, 2457.6 Mbps, and 3072 Mbps line bit rates only.
- FPGA bridge FIFOs in the SERDES/PCS (DCU in ECP5) block are bypassed in both the receive and transmit directions
- Logic blocks supporting receive direction 10b word alignment, 10b/8b decoding and core-violation detection in the SERDES/PCS (DCU in ECP5) block are bypassed and the corresponding functions are implemented in FPGA gates

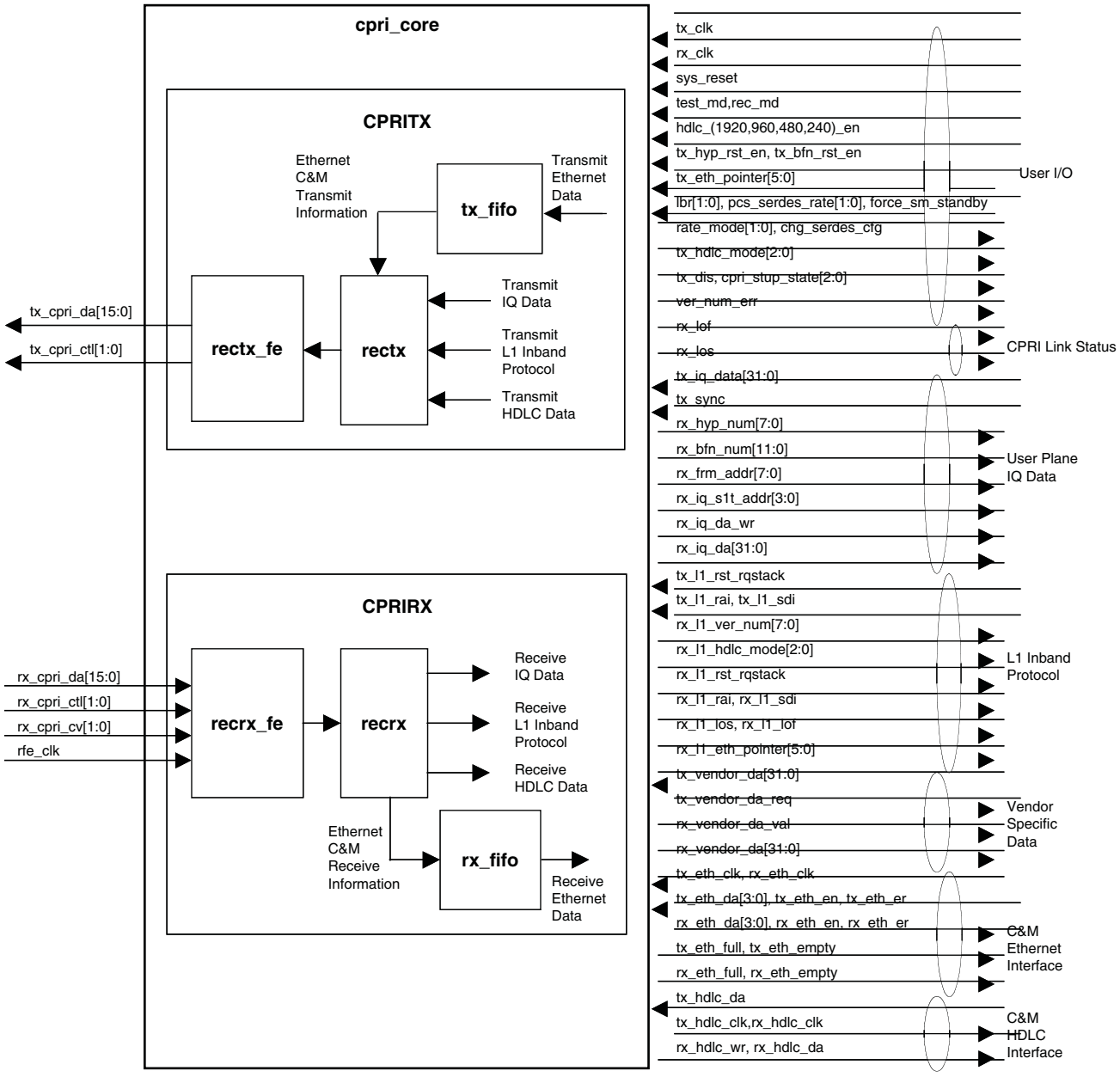
This chapter provides a functional description of the CPRI IP core.

Block Diagram

The complete CPRI IP core includes two key components, the CPRI IP logic core and separate logic blocks that support the interface between the logic core and the integrated PCS/SERDES block.

A block diagram of the CPRI IP logic core is shown in Figure 2-1. The CPRI IP logic core is identical for both the basic and low latency core configurations.

Figure 2-1. CPRI IP Logic Core Block Diagram



General Description

The complete CPRI IP core includes two key components, the CPRI IP logic core and separate logic blocks that support the interface between the logic core and the SERDES and PCS (DCU for ECP5) functions integrated in the FPGA. Figure 2-1 shows a block diagram of the CPRI IP logic core. The CPRI IP logic core is identical for both the basic and low latency core configurations. Control and status parameters specifying core functionality are managed via bit-mapped I/O that may be hard-wired or interfaced to programmable registers, providing users with optimal flexibility in defining static and/or dynamic management of the various functional parameters needed for their particular applications.

The CPRI IP Core supports a parallel user IQ data interface, a serial HDLC interface, a serial Ethernet interface, a parallel interface for vendor specific information, and provides individual signals which allow the user to insert/receive L1 Inband Protocol information to/from the CPRI link. The interfaces to the user application are by design very simple to allow the CPRI IP Core to be as flexible as possible. All higher-level functions such as Ethernet MAC functions, HDLC framing, etc. are all intended to be done outside of the CPRI IP Core in user logic or in other Lattice IP Modules.

Figure 2-2 shows a system level block diagram of CPRI IP core instantiated in a ECP5 series FPGA. As indicated in Figure 2-2, additional IP cores may be instantiated to support multiple RP3 data links. Also shown in Figure 2-2 are RXFE and TXFE logic blocks that support the interface between IP logic core and the integrated SERDES/PCS (DCU for ECP5) block.

Figure 2-2. CPRI IP Core System Block Diagram in LatticeECP3

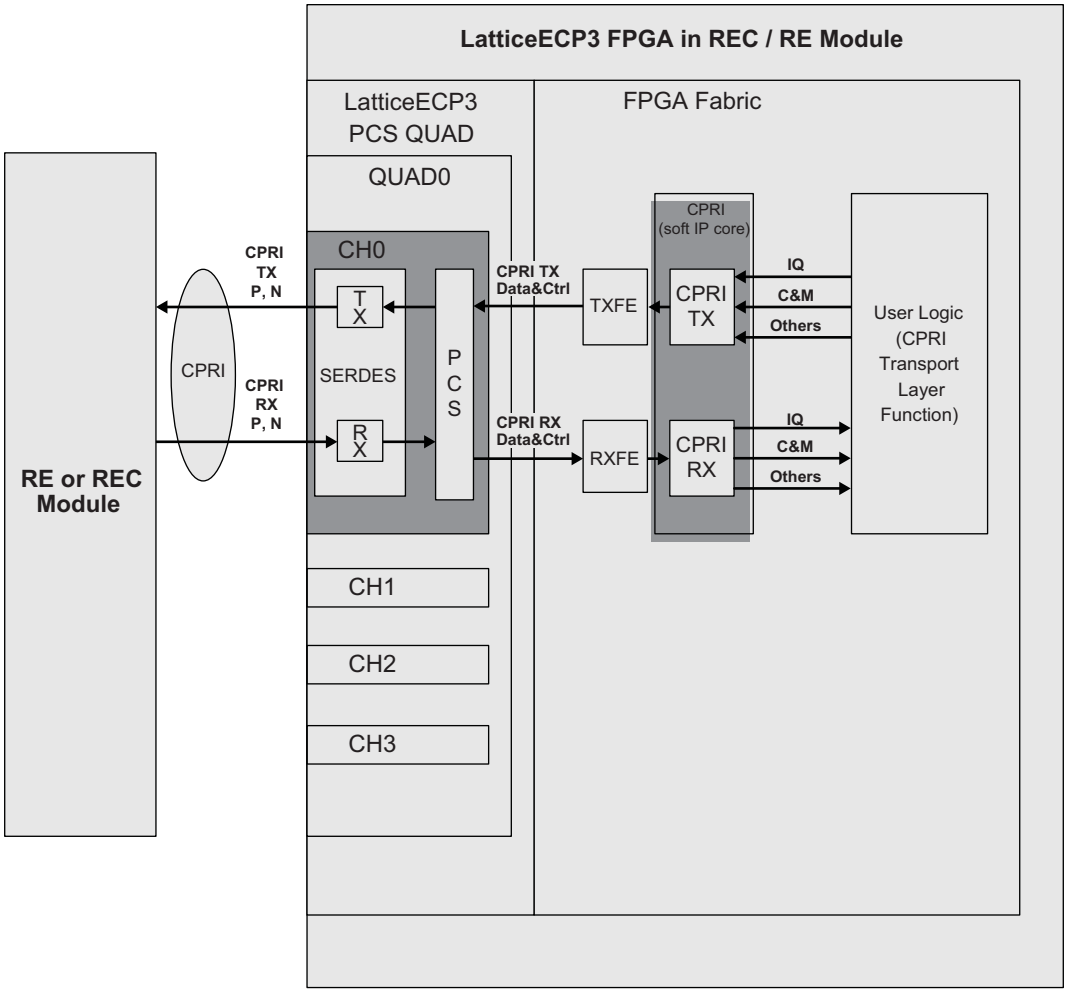
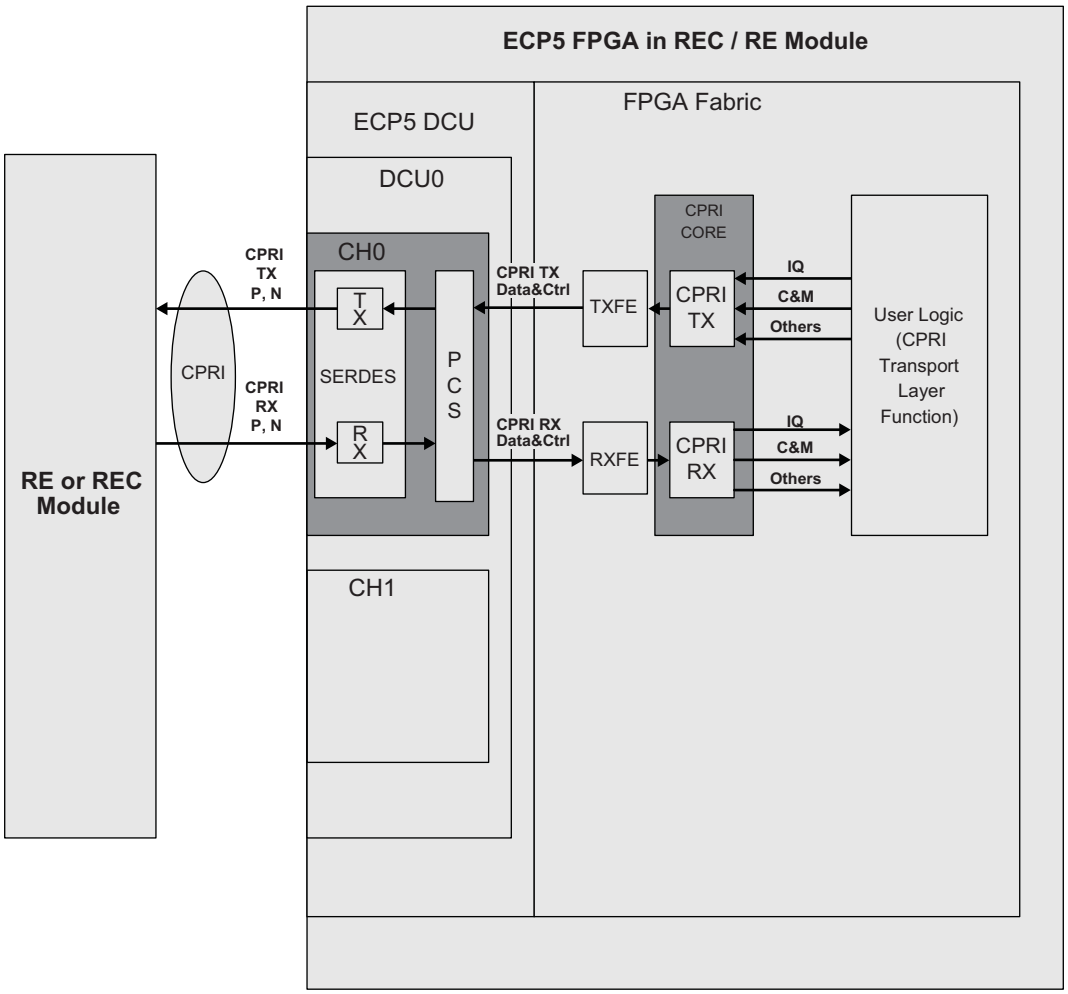
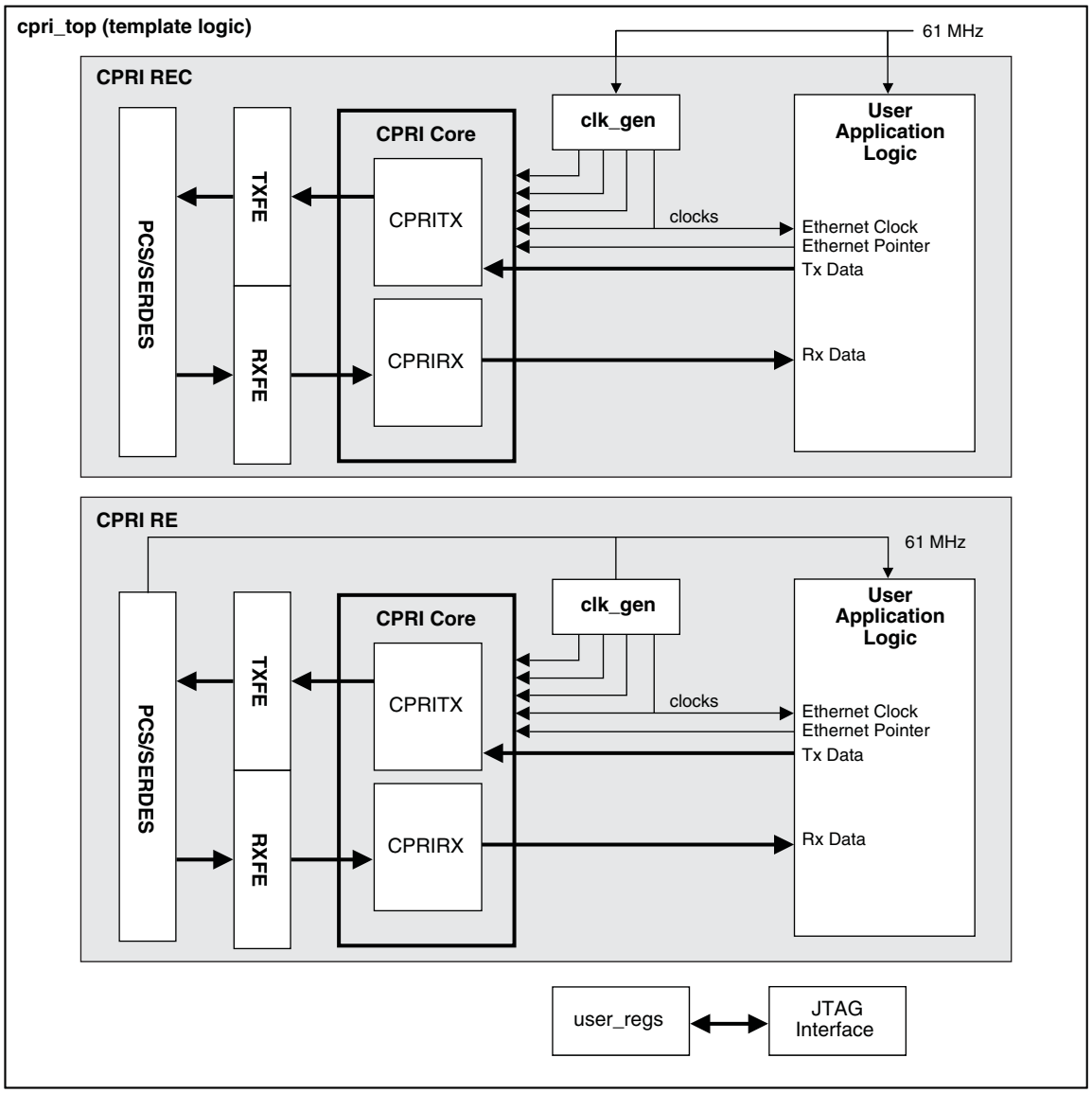


Figure 2-3. CPRI IP Core System Block Diagram in ECP5



Included in the CPRI IP core evaluation package is a reference module that provides an example of how the IP core is instantiated at the top level, as shown in Figure 2-4. This top-level template is provided in RTL format and provides a good starting point from which the user can begin to add custom logic to a design.

Figure 2-4. Top-Level Template Included with CPRI IP Core (Includes Example Connections for using IP Core in REC and RE Modes)



The CPRI IP logic core is provided in NGO format. For the basic CPRI IP core configuration, the rxfe and txfe logic blocks are provided in RTL format and may be used as-is or modified as necessary. For the low latency CPRI IP core configuration, the rxfe block also includes soft PCS logic in NGO format as described in detail in IPUG74, *CPRI IP Core Low Latency Variation Design Considerations User's Guide.*

Also included in the top-level netlist is a user side driver/monitor module and register implementation module for optional use. These included modules are used in the evaluation simulation capability. The driver/monitor module is used to provide data in the transmit direction and verify data in the receive direction. The register implementation module is used to control the IP core. The overall top-level design can be used without modification in the debugging phase on physical hardware needing only a CPRI source/sink capability to verify IP core operation.

Signal Descriptions

Table 2-1. CPRI I/O Signal Descriptions

Signal Name	Direction Input/Output	Width (Bits)	Description
System Clock and Reset			
tx_clk	I	1	61 MHz system clock input
rx_clk	I	1	61 MHz receive side clock input
sys_reset	I	1	Active low reset
txrst	I	1	Active high Ethernet FIFO reset (in TX CPRI)
rxrst	I	1	Active high Ethernet FIFO reset (in RX CPRI)
User Interface			
rec_md	I	1	Select between REC or RE mode, REC = 1
test_md	I	1	test_mod = 1, speed up the timer for simulation
hdlc_240_en	I	1	240 KHz HDLC frequency enable
hdlc_480_en	I	1	480 KHz HDLC frequency enable
hdlc_960_en	I	1	960 KHz HDLC frequency enable
hdlc_1920_en	I	1	1920 KHz HDLC frequency enable
hdlc_2400_en	I	1	2400 KHz HDLC frequency enable (only available in 3G)
auto_cnt	I	1	For master CPRI TX only, update Z64, Z128, Z192 with hyp_cnt_init and bfn_cnt_init when low. When it is high, control words are updated by internal counter.
hyn_cnt_init	I	8	For master CPRI TX only, used for two purposes: Auto_cnt = 0: update Z64 Auto_cnt = 1: update internal hfn counter when tx_sync = 1 and tx_hyp_rst_en = 1
bfn_cnt_init	I	12	For master CPRI TX only, used for two purposes: Auto_cnt = 0: update Z128, Z192 Auto_cnt = 1: update internal bfn counter when tx_sync = 1 and tx_bfn_rst_en = 1
tx_hyp_rst_en	I	1	Transmit side hype frame counter reset enable during tx_sync active
tx_bfn_rst_en	I	1	Transmit side bfn counter reset enable during tx_sync active
tx_eth_pointer	I	6	Transmit Ethernet pointer value
lbr_en	I	2	[1:0] – Maximum CPRI Line bit rate enabled by user
force_sm_standby	I	1	Force startup state machine to standby
pcs_serdes_rate	I	2	CPRI line rate supported by PCS/SERDES
rate_mode	O	2	Final negotiated CPRI Line bit rate after start-up sequence completes
chg_serdes_cfg	O	1	Indicator to program the SERDES for different rate
tx_hdlc_mode	O	3	Transmit side HDLC negotiated bit rate
tx_dis	O	1	For RE mode disable transmit side active high
cpri_stup_stat	O	3	Start up sequence state state 1 = synchronization state 2 = protocol setup state 3 = c/m plane setup state 4 = interface and vendor negotiation state 5 = operation
ver_num_err	O	1	Version number error
CPRI Link Status			
rx_lof	O	1	Receive side Loss of Frame
rx_los	O	1	Receive side Loss of Signal

Table 2-1. CPRI I/O Signal Descriptions (Continued)

Signal Name	Direction Input/Output	Width (Bits)	Description
User Plane IQ Data Signals			
tx_iq_da	I	32 (40 for 3G)	Transmit side IQ data
tx_sync	I	1	Transmit side IQ data sync
rx_hyp_num	O	8	Receive side hyper frame number
rx_bfn_num	O	12	Receive side basic frame number
rx_frm_addr	O	8	Receive side frame address
rx_iq_slt_addr	O	4	Receive side IQ data slot address
rx_iq_da_wr	O	1	Receive side IQ data write enable
rx_iq_da	O	32 (40 for 3G)	Receive side IQ data
L1 Inband Protocol Signals			
tx_l1_rst_rqstack	I	1	Transmit reset request or acknowledge
tx_l1_rai	I	1	Transmit Remote Alarm indication
tx_l1_sdi	I	1	Transmit SAP Defect Indication
rx_l1_ver_num	O	8	Received version number
rx_l1_hdlc_mode	O	3	Receive side HDLC negotiated bit rate
rx_l1_rst_rqstack	O	1	Receive reset request or acknowledge
rx_l1_rai	O	1	Receive Remote Action Indication
rx_l1_sdi	O	1	Receive SAP Defect Indication
rx_l1_los	O	1	Receive Loss of Signal
rx_li_lof	O	1	Receive Loss of Frame
rx_l1_eth_pointer	O	6	Receive side Ethernet pointer value
Vendor-Specific Data Interface Signals			
tx_vendor_da	I	32 (40 for 3G)	Vendor-specific transmit data
tx_vendor_da_req	O	1	Vendor-specific transmit data request
rx_vendor_da_val	O	1	Vendor-specific receive data valid
rx_vendor_da	O	32 (40 for 3G)	Vendor-specific receive data
CPRI to SERDES Signals			
tx_cpri_ctl	O	2	Transmit side CPRI control to SERDES
tx_cpri_da	O	16 (40 for 3G applies to tx-cpri_da & rx_cpri_da) (5 for 3G applies to tx_cpri_ctl, rx_cpri_ctl & rx_cpri_cv)	Transmit side CPRI data to SERDES
rx_cpri_ctl	I	2	Receive side CPRI control from SERDES
rx_cpri_da	I	16	Receive side CPRI data from SERDES
rx_cpri_cv	I	2	Receive side CPRI code violation from SERDES
slip_rxfe	O	1	Receive side slip to get “BC” in low byte

Table 2-1. CPRI I/O Signal Descriptions (Continued)

Signal Name	Direction Input/Output	Width (Bits)	Description
C&M Ethernet Interface Signals			
tx_eth_clk, rx_eth_clk	I	1	Transmit Ethernet clock, Receive Ethernet clock
tx_eth_da	I	4	Transmit Ethernet data (1 bit wide in Serial mode)
tx_eth_en	I	1	Transmit Ethernet data enable (not equipped in Serial mode)
tx_eth_er	I	1	Transmit Ethernet error (not equipped in Serial mode)
rx_eth_da	O	4	Receive Ethernet data (1 bit wide in Serial mode)
rx_eth_en	I	1	Receive Ethernet data enable (not equipped in Serial mode)
rx_eth_er	I	1	Receive Ethernet error (not equipped in Serial mode)
tx_eth_almost_full	O	1	Transmit Ethernet FIFO almost full. This is for user flow control.
tx_eth_empty	O	1	Transmit Ethernet FIFO empty
rx_eth_full	O	1	Receive Ethernet FIFO full
rx_eth_empty	O	1	Receive Ethernet FIFO empty
rx_eth_fifo_err	O	2	Receive Ethernet FIFO error flag (only for Fixed mode)
C&M HDLC Interface Signals			
tx_hdlc_da	I	1	Transmit HDLC data
tx_hdlc_clk	O	1	Transmit HDLC clock
rx_hdlc_clk	O	1	Receive HDLC clock
rx_hdlc_wr	O	1	Receive HDLC write
rx_hdlc_da	O	1	Receive HDLC data

Timing Specifications

Refer to DS1021, *LatticeECP3 Family Data Sheet* and DS1044, *ECP5 Family Data Sheet* for detailed SERDES and FPGA interface timing and electrical specifications.

CPRI Overview

Figure 2-5 shows the CPRI layer 1 and layer 2 protocols. The CPRI IP core interleaves user IQ (in-phase and quadrature) data with control and management (C&M) data into the frame and hyperframe formats specified by the CPRI Specification and shown in Figure 2-6 and Figure 2-7, respectively.

Figure 2-5. CPRI Protocol Overview

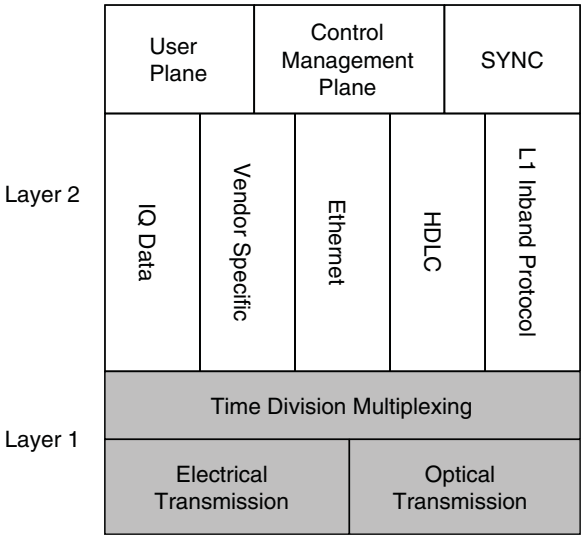


Figure 2-6. CPRI Frame (Shown for 614.4 Mbps, 1228.8 Mbps, and 2457.6 Mbps Line Rates)

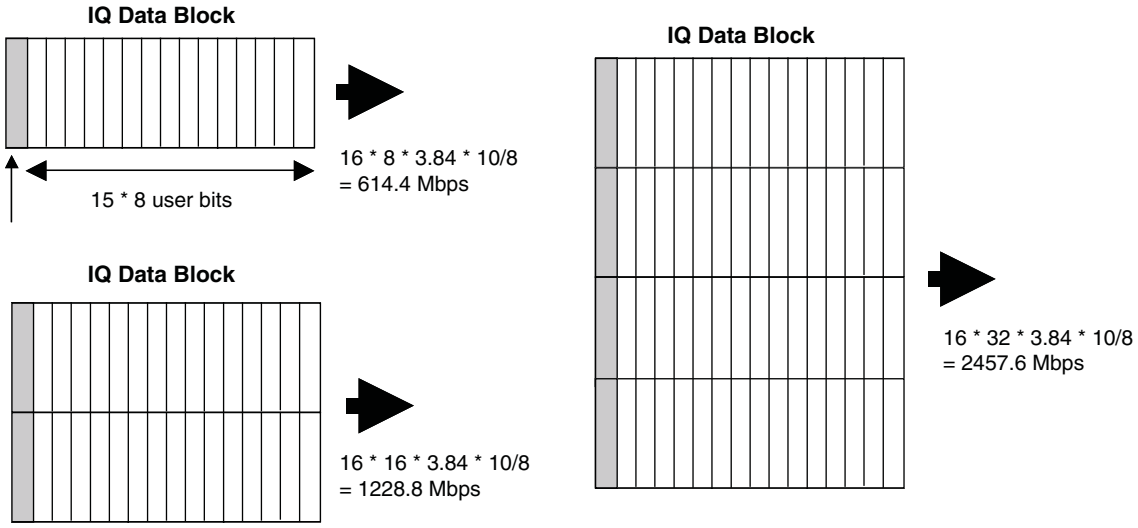
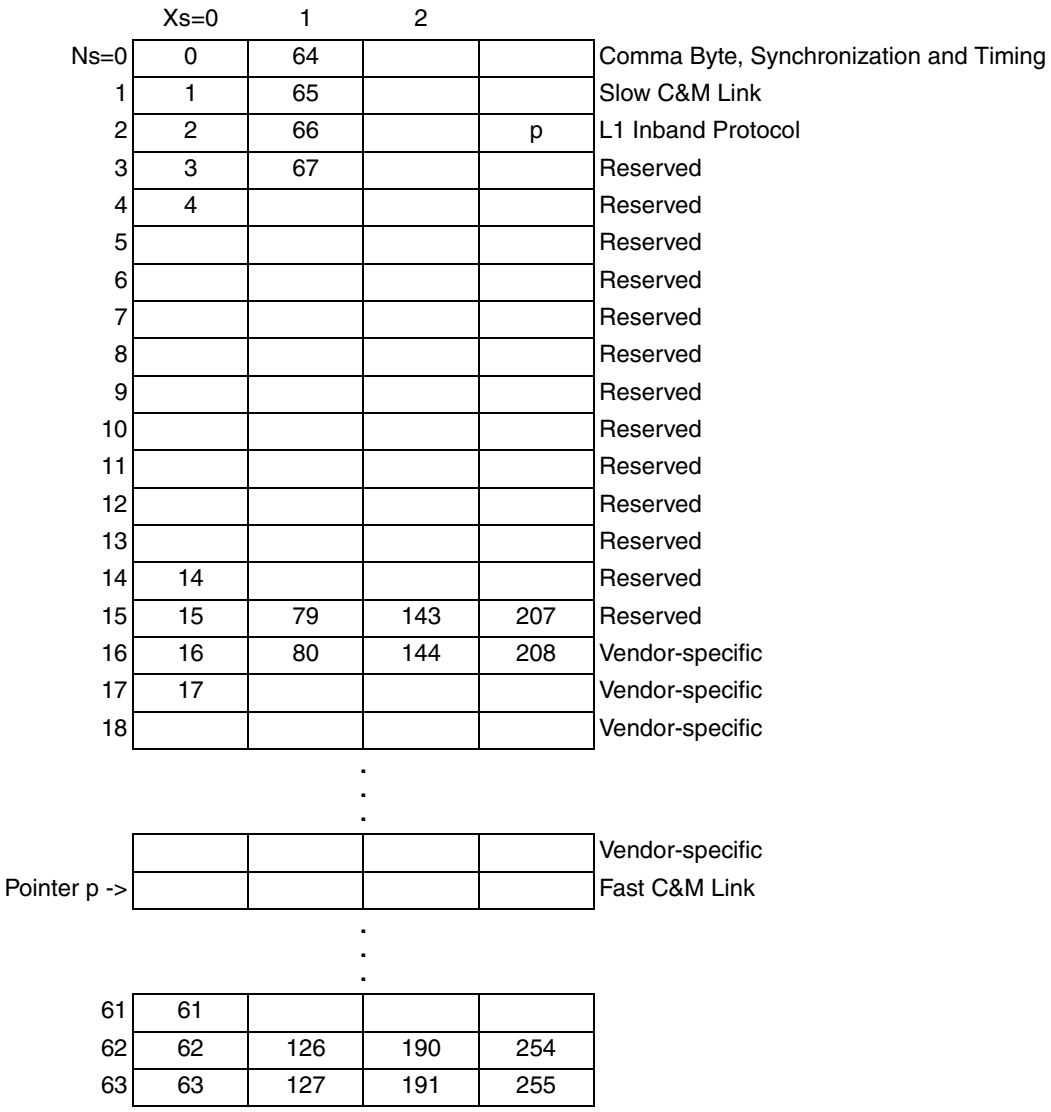


Figure 2-7. CPRI Hyperframe Format



Four possible line rates for the CPRI frame are supported, as shown in Table 2-2. The maximum line bit rate that the core must support is selected by setting the lbr_en[1:0] input signals. Once the maximum line bit rate has been selected, all lower line bit rates are automatically enabled. The bit rate between the REC and RE is then negotiated between the transmitting and receiving ends by the start-up sequence state machine within the CPRI IP core.

Table 2-2. CPRI Line Rates

CPRI Line Bit Rate	Rate (Mbps)	lbr_en[1:0]
Option 1	614.4	00
Option 2	1228.8	01
Option 3	2457.6	1X or 10 (LatticeECP3 only)
Option 4	3072	11 (LatticeECP3 only)

Functional Overview

A block diagram of a typical LatticeECP3 CPRI application is shown in Figure 2-2 and a block diagram of a typical ECP5 CPRI application is shown in Figure 2-3. These examples show four CPRI link transceivers implemented in a LatticeECP3 and an ECP5 device. The major functional blocks in the example are the CPRI Soft IP core, the PCS/SERDES block, and the user application logic. The CPRI IP Core (cpri_core), shown in Figure 2-1, consists of two major functional blocks, the transmitter (CPRITX) and the receiver (CPRIRX). One side of each of these two blocks interfaces to the PCS/SERDES module, and the other side interfaces to the user logic that implements the data link and higher layers of the CPRI protocol.

The CPRI IP Core multiplexes user IQ data with synchronization and C&M data. The IP core is being designed with very simple and versatile interfaces for transferring data to the user application logic.

IQ data is transferred between the IP core and the user logic using a parallel data interface. In the transmit direction, the user application provides a sync pulse which determines the start of a CPRI BFN (every 150 hyperframes) frame. In the receive direction, the IP core transfers IQ data to the user application along with framing information recovered from the CPRI link.

In the transmit direction, the CPRI IP core accepts user C&M data at either the HDLC and/or the Ethernet interface and interleaves that data with the user IQ data into the CPRI frame. In the receive direction, C&M data received on the CPRI link is disinterleaved from the user IQ data, and the HDLC or Ethernet encapsulated data is presented to the user at either the HDLC or Ethernet interfaces of the CPRI IP core.

The C&M data can be encapsulated in either HDLC (slow channel) or Ethernet (fast channel) as desired by the user. For HDLC, the CPRI IP Core only performs the interleaving of the C&M data with user IQ data. It does not provide any of the HDLC Level 2 functions such as framing and serial to parallel conversion. These functions must be done in the user application logic or in another IP module. For Ethernet, three modes are provided. In mode 1, the core does not provide the serial to parallel and 4B/5B conversion. The interface is a serial bitstream that contains 5B encoded nibbles. In mode 2, the core provides a standard MII interface that can be connected to a standard Ethernet MAC. In mode 3, the core provides an MII like interface that has the clocks as inputs instead of outputs. This interface can be connected to a PHY that is operating at the 100 Mbps standard rate.

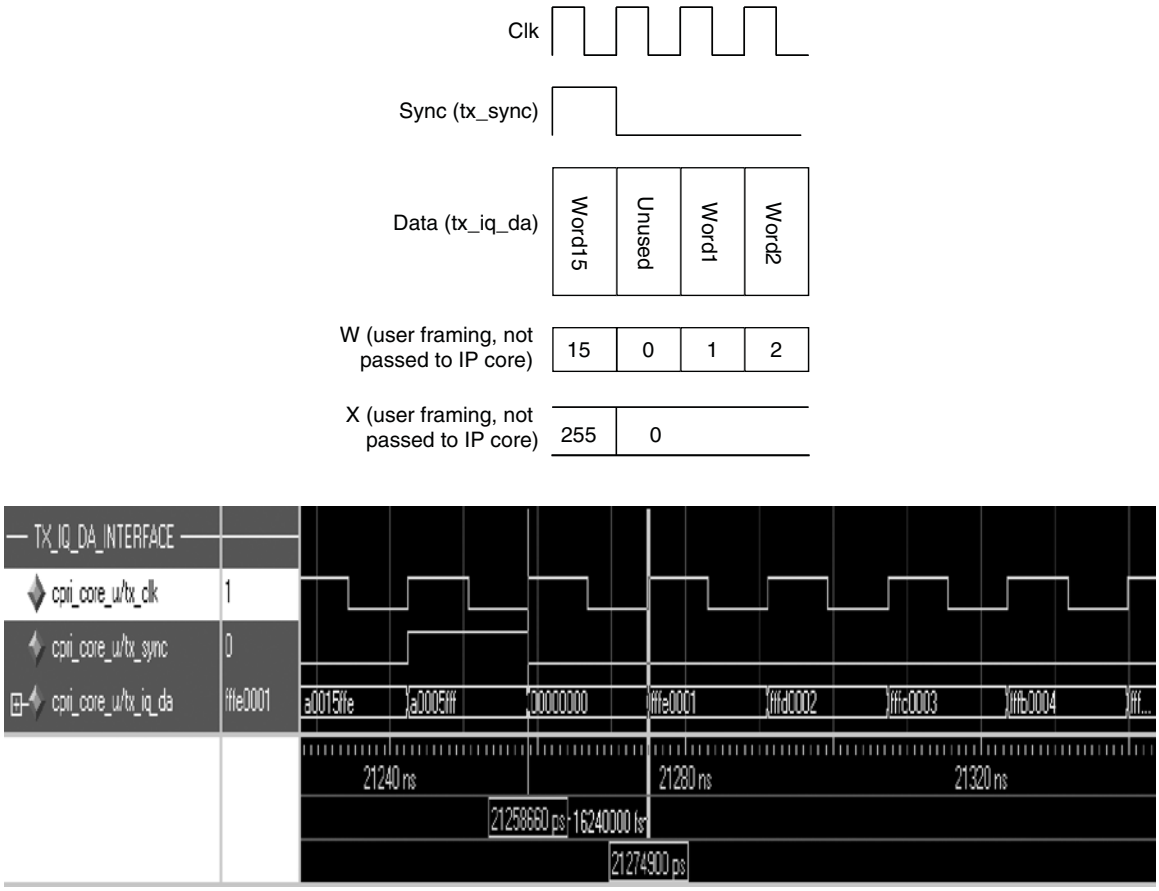
The following sections discuss the interfaces of the IQ data, HDLC, Ethernet, and vendor information interfaces of the CPRI IP core.

User Plane IQ Data Interface

The user IQ data interface supports the four line rates shown in Table 2-2. Both the transmit and receive directions provide a 40-bit IQ data bus. However, the number of bits used depends on the line rate selected by the user. For the 614.4 Mbps line rate, 8 of the 32 bits on the IQ data interface are used (bits 7:0). For 1228.8 Mbps, bits 15:0 are used, and for a line rate of 2457.6 Mbps, 32 bits are used. For a line rate of 3072 Mbps, all 40 bits are used.

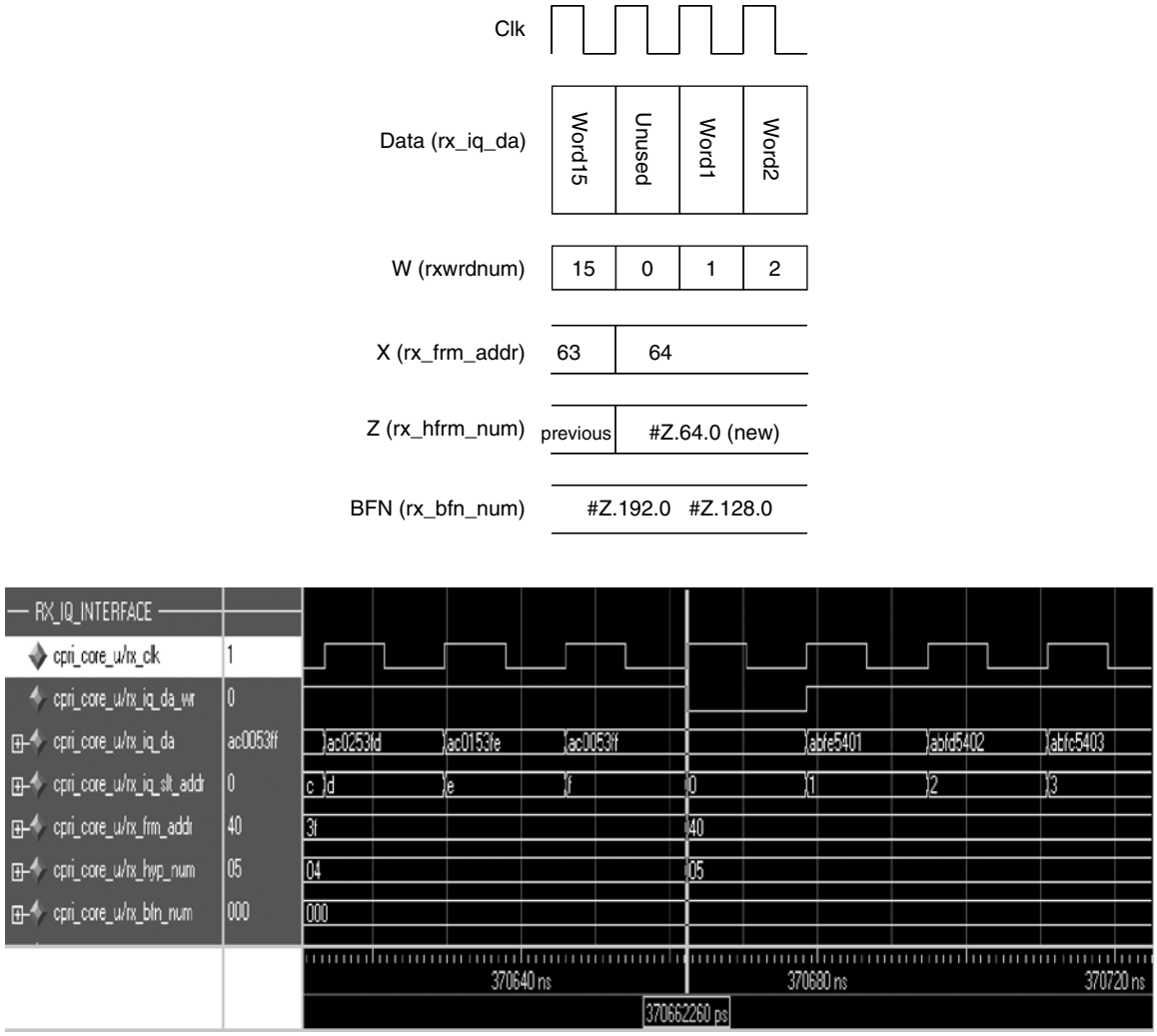
In the transmit direction, the user application must provide a sync pulse to the CPRI IP core to indicate the start of each CPRI BFN (every 150 hyperframes) frame. This sync pulse must be aligned with the IQ data, as shown in Figure 2-8.

Figure 2-8. Tx User IQ Interface Sync Pulse Alignment



In the receive direction, the CPRI IP Core provides IQ data to the user interface along with a word number (W), frame number (X), hyperframe number (Z), and a BFN number, as shown in Figure 2-9.

Figure 2-9. Rx IQ Interface Data and Frame Number Alignment



Ethernet Interface

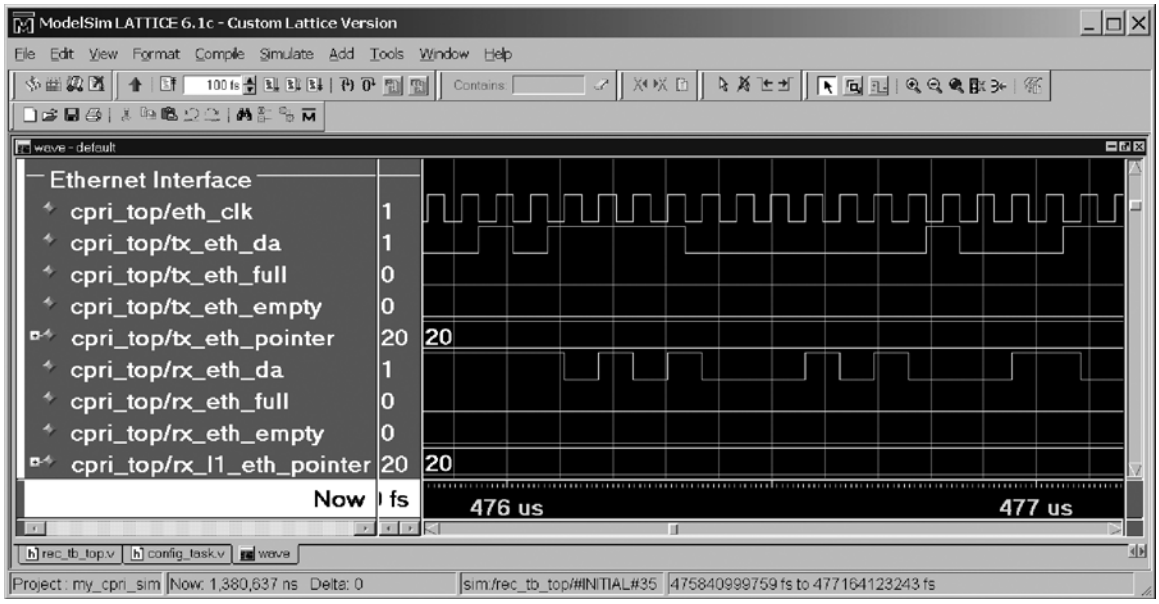
Mode 1 (Serial)

The CPRI IP Core Ethernet Interface transmits and receives Ethernet data to/from the user application logic using a simple serial data connection. The amount of bandwidth on the CPRI link which is allocated to Ethernet (fast C&M) data is determined by the pointer value (Z.194.0) which in turn determines which subchannel number the Ethernet data starts at within the CPRI hyperframe. The pointer value is set by the user using the tx_eth_pointer[7:0] inputs to the IP core. The transmitter and receiver of the REC and RE ends must be able to use the same pointer value.

Since a buffer is provided within the IP core for Ethernet data, the Ethernet interface clock between the IP core and the user application logic must be chosen such that the correct number of words are transferred between the IP core and the user's logic each frame or else the internal buffer may overrun or underrun. Due to the large number of possible pointer values available it is not practical for the IP core to generate the Ethernet interface clock. Instead, the IP core is designed such that the pointer value and the Ethernet interface clock are provided as inputs to the Core, giving the user the maximum flexibility. The template logic (cpri_top) shown in Figure 2-10, which will be delivered with the IP core, will include an example of how to generate and connect an Ethernet interface clock frequency of 61.44 MHz. Additional clock frequencies and pointer values may be set by the user by modifying the template logic. The difficulty which will be encountered in generating various Ethernet clock frequencies and pointer values will be entirely dependent on the clock frequencies available in the user's system. This is why the

generation of the Ethernet clock frequency, and the assignment of the pointer value, must be done in the template logic and not within the IP core.

Figure 2-10. Ethernet C&M Interface



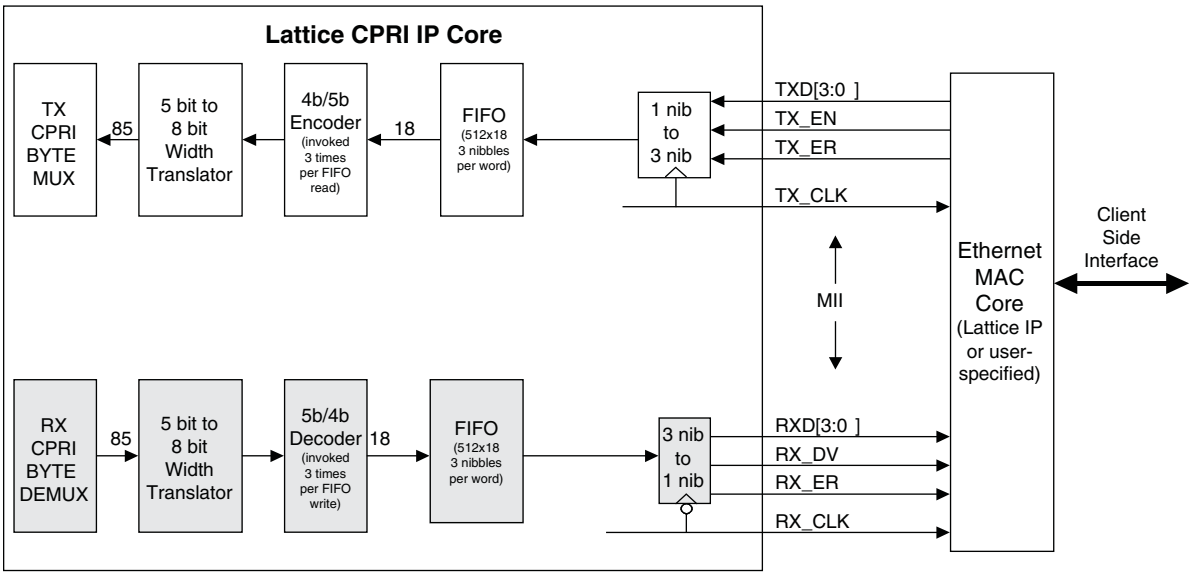
Mode 2 (Matched Rate MII)

The CPRI IP Core Ethernet Interface transmits and receives Ethernet data to/from the user application logic using a non-standard rate MII interface. The amount of bandwidth on the CPRI link which is allocated to Ethernet (fast C&M) data is determined by the pointer value (Z.194.0) which in turn determines which subchannel number the Ethernet data starts at within the CPRI hyperframe. The pointer value is set by the user using the tx_eth_pointer[7:0] inputs to the IP core.

Since a buffer is provided within the IP core for Ethernet data, the Ethernet interface clock between the IP core and the user application logic must be chosen such that the correct number of words are transferred between the IP core and the user's logic each frame or else the internal buffer may overrun or underrun. Due to the large number of possible pointer values available it is not practical for the IP Core to generate a periodic interface clock. Instead, the IP core is designed to produce a gapped clock. The Ethernet pointer and CPRI line rate are used to index a two parameter table that specifies the frequency and number of pulses contained in the gapped clock. The transmit Ethernet clock is based on the negotiated rate and transmit pointer. The receive Ethernet clock is based on the negotiated rate and the received L1 inband pointer in Z.194.0.

The core provides the 4B/5B code conversion required by the CPRI specification. The logic that interfaces the MII to CPRI byte mux/demux is shown in Figure 2-11.

Figure 2-11. Matched Rate MII CPRI Ethernet Interface



Note 1: TX_CLK and RX_CLK are gapped clocks with transition rates matching the effective data rate of the Fast C&M Channel specified by the pointer value in control byte #Z.194.0 and the CPRI line rate.
Note 2: Both FIFOs are read and written as needed to source and sink data/control from/to the MII and CPRI mux/demux.

Mode 3 (100 Mb/s Fixed Rate MII)

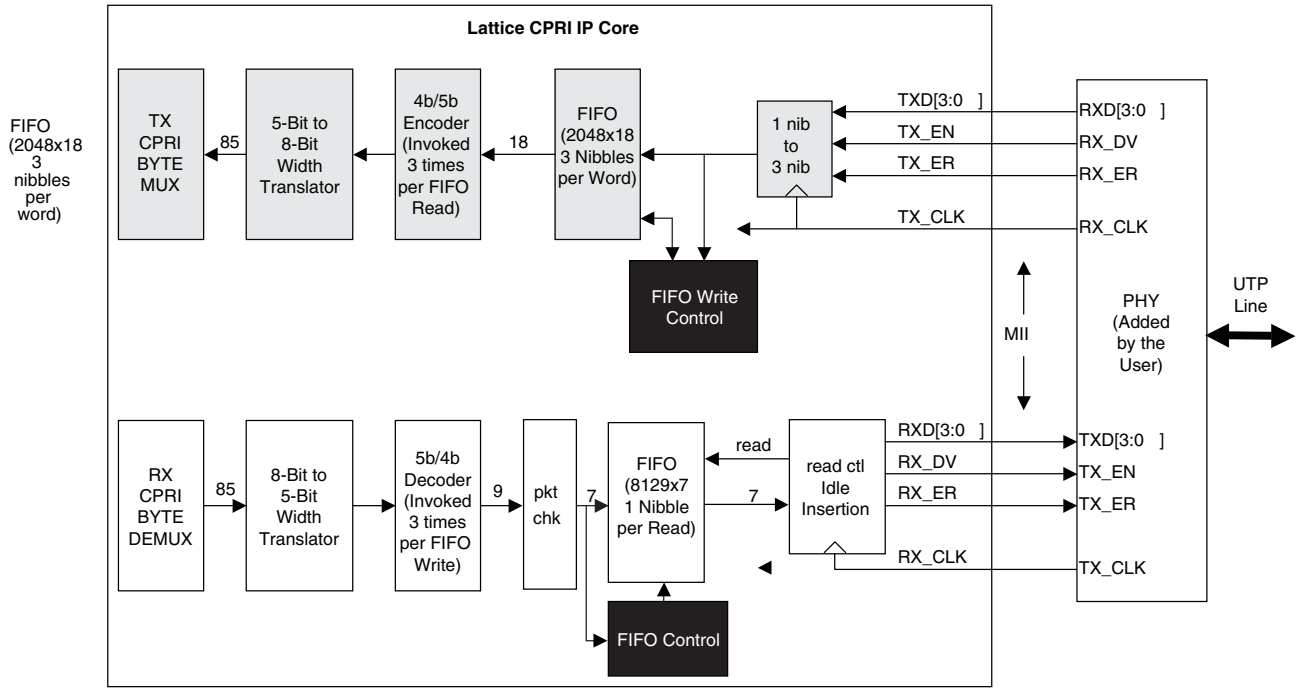
The CPRI IP Core Ethernet Interface transmits and receives Ethernet data to/from the user application logic using a 100 Mbps rate MII interface. The amount of bandwidth on the CPRI link which is allocated to Ethernet (fast C&M) data is determined by the pointer value (Z.194.0) which in turn determines which subchannel number the Ethernet data starts at within the CPRI hyperframe. The pointer value is set by the user using the tx_eth_pointer[7:0] inputs to the IP core.

The logic needed to support mode 3 is similar to that required for mode 2 with the exception of handling the reading and writing of the FIFOs. On the transmit side, the FIFO is always written at a faster rate than it is read. This means that the reading of this FIFO is the same as mode 2. Interframe gap is only written into the FIFO if the almost empty threshold is not maintained. This prevents the FIFO from being filled with idle information. It is the responsibility of the driver of the 100 Mbps link to not over run the FIFO. There is no flow control provided to accommodate the rate mismatch between the 100Mbps link and the CPRI link. Packets must be spaced to allow enough time for the CPRI to transmit packets. On the receive side, the FIFO is always read at a faster rate than it is written. When the FIFO input data is not idle and passes basic SSD and ESD checks, it is written to the FIFO. When idle is detected on the FIFO input the FIFO is not written. When an end of packet is detected, a request is sent to the FIFO Read Control to send the packet on to the MII. A complete packet is received before it is read from the FIFO. FIFO reads begin when a request is received from the FIFO Write Control. FIFO reads continue until an end of packet is detected. When an end of packet is detected, FIFO reads stop and a minimum of 24 nibble IDLEs are supplied to the MII interface.

For 3.072 Gbps CPRI link, the MAC Ethernet MAX bandwidth is 3.84M/256*(64-20)*40, that is 26.4 Mbps (when Ethernet pointer p is set to 20), and MIN bandwidth is 0.6 Mbps (when Ethernet pointer p is set to 63). For example, if your real Ethernet bandwidth is 10 Mbps, please set Ethernet pointer p as 48~55, the corresponding bandwidth is 5.4~9.6 Mbps. If Ethernet pointer p is set too small, the TX FIFO for Ethernet may be empty, and the Ethernet package may be inserted by duplicated data of the partial package. If Ethernet pointer p is set too large, the TX FIFO for Ethernet may be full.

The core provides the 4B/5B code conversion required by the CPRI specification. The logic that interfaces the MII to CPRI byte mux/demux is shown in Figure 2-12.

Figure 2-12. 100 Mbps Fixed Rate MII CPRI Ethernet Interface



1. TX_CLK and RX_CLK are supplied by the external PHY device and assumed to be 25MHz.
2. The transmit FIFO is read as needed to supply data/control to the CPRI mux.
3. The transmit FIFO is written as needed to sink packets from the MII and supply idle.
4. The receive FIFO is read and written as needed to source and sink packets from/to the MII and CPRI demux. Idle is supplied seperately.

HDLC Interface

A timing diagram for the Ethernet C&M interface is given in Figure 2-13. The CPRI IP Core HDLC interface transmits and receives HDLC data to/from the user application logic using a serial data connection. The user selects the desired frequencies which the IP core will support by setting the hdlc_(240,480,960,1920)_en input signals. The HDLC data rates shown in Table 2-3 are supported.

The CPRI IP core HDLC Interface does not perform any HDLC framing or processing. This must be performed in the user application logic or in another IP module. The transmitter of the IP core will negotiate with the receiver of the end of the CPRI link to achieve the maximum HDLC rate possible.

Figure 2-13. HDLC C&M Interface

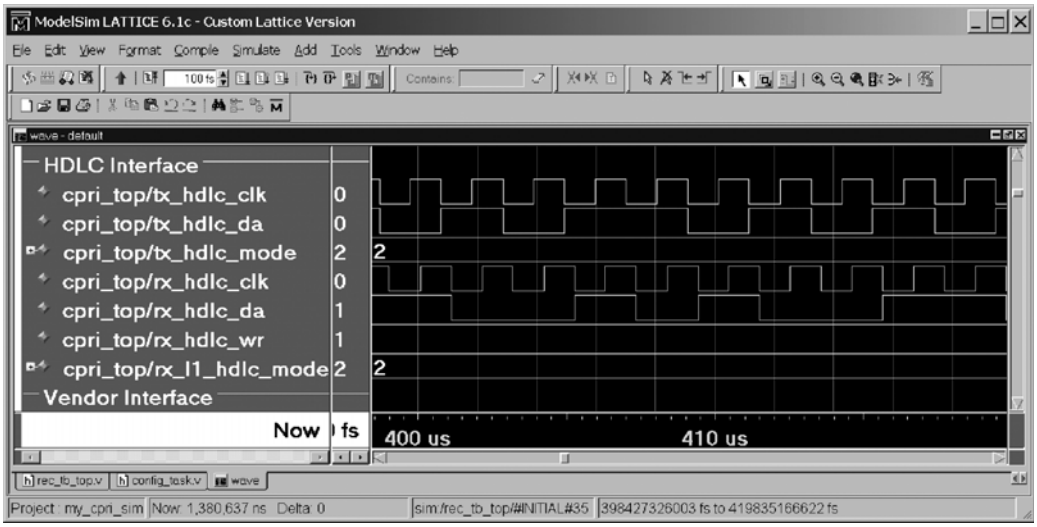


Table 2-3. HDLC Frequencies Supported

HDLC Option	Input Signal	HDLC Data Rate (Kbps)
0	None selected	HDLC disabled
1	hdlc_240_en = 1	240
2	hdlc_480_en = 1	480
3	hdlc_960_en = 1	960
4	hdlc_1920_en = 1	1920
5	hdlc_2400_en = 1	2400

L1 Inband Protocol Interface

The L1 Inband Protocol Interface provides signals that allow the user application logic to populate the control words listed in the CPRI specification, and to read the value of the L1 Inband Protocol control words that are received from the CPRI link.

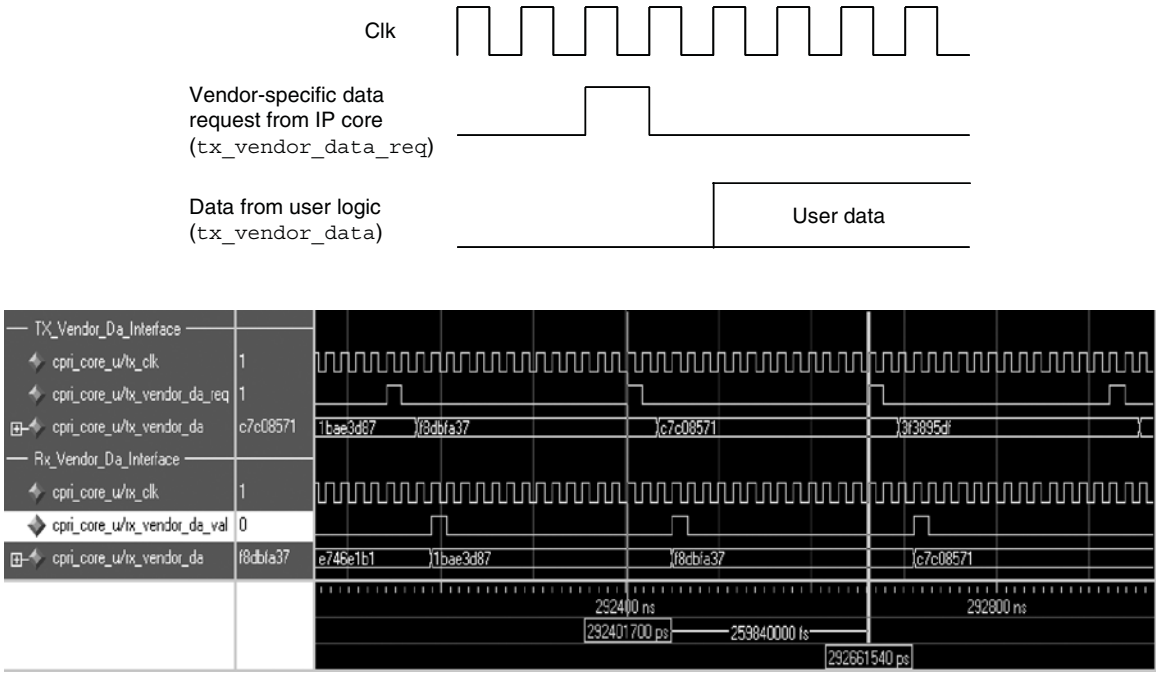
Vendor Specific Information

Figure 2-14 illustrates the timing of the vendor-specific information interface between the IP core and the user application logic. The IP core provides a simple parallel interface for merging vendor-specific information into the CPRI frame. The number of bytes available in the CPRI frame for vendor-specific information is dependent on the pointer value chosen. Any FIFOs required to support the bandwidth allocated to vendor-specific information must be implemented in the user logic.

The IP core provides a data request signal to the user application logic called tx_vendor_data_req. When this signal goes high, the user logic must return the vendor-specific data to the IP core within six clock cycles. The user logic must continue to provide vendor-specific data until the tx_vendor_data_req signal is once again asserted. If the user does not have any data to send in the vendor-specific bytes, then it must insert idle code.

In the receive direction, incoming vendor-specific information which is recovered from the CPRI link is presented to the user application logic along with a rx_vendor_da_val signal. The vendor-specific information interface uses either 8, 16, or 32 (40 in 3G) of the available data bits, depending on which CPRI line bit rate has been selected. This is similar to the IQ data interface described previously.

Figure 2-14. Timing for Vendor-Specific Information Interface



Start-up Sequence

The CPRI IP Core provides a start-up state machine which executes the startup state transitions as shown in Figure 30 in the CPRI Specification (Reference 2). This state machine will automatically perform the synchronization of Layer 1, and it will align the capabilities of the REC and the RE (line bit rate, protocol, C&M link speed, C&M protocol, and vendor specific signaling). The CPRI IP Core depends on software to program the correct data bus width into the LatticeECP3/ECP5 SERDES through the SCI Bus. For a 1228.8 or 2457.6 Mbps CPRI line bit rate, the SERDES must be provisioned for a 16-bit interface to the FPGA logic. For all CPRI line bit rate, the SERDES must be provisioned for an 8-bit interface to the FPGA logic to achieve low latency purpose. Since the provisioned mode of the SERDES may need to change while the start-up sequence is negotiating the CPRI line rate, a signal (chg_serdes_cfg) is provided to the user application logic indicating that the mode the SERDES should be programmed for needs to be changed. The user application must monitor this signal and program the SERDES to the correct mode while the REC and RE are attempting to match line bit rates. When the application has finished programming the PCS/SERDES it writes the rate that is being supported to lbr_en[4:3].

The IPexpress ™ and the Clarity Designer tools are used to create IP and architectural modules in the Diamond software. IPexpress is for LatticeECP3 CPRI IP Core and Clarity Designer is for ECP5 CPRI IP Core. Refer to "IP Core Generation" on page 27 for a description on how to generate the IP.

Table 3-1 provides the list of user configurable parameters for the CPRI IP core. The parameter settings are specified using the CPRI IP core Configuration GUI in IPexpress.

Table 3-1. IP Core Parameters

Parameters	Range/Options	Default
General Options		
Design Entry	Low Latency	Low Latency
Ethernet Mode	100 Mb/s, Matched, Serial	100 Mb/s
Eval Configuration		
Synthesis Tool	Synplify	Synplify

CPRI Configuration Dialog Box

Figure 3-1 shows the CPRI Configuration dialog box.

Figure 3-1. CPRI Configuration Dialog Box

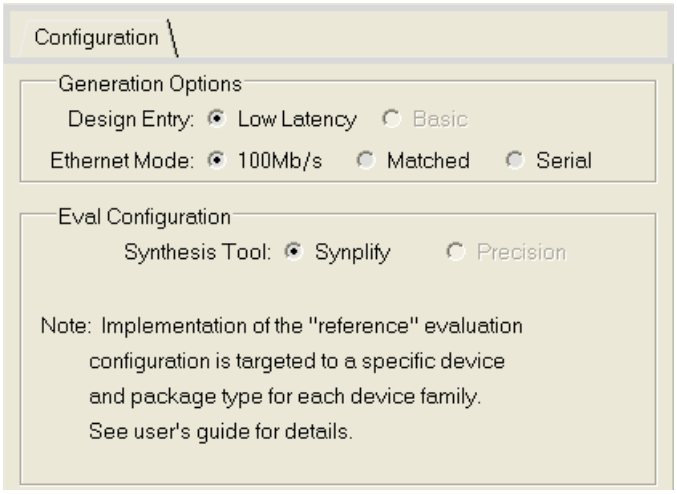
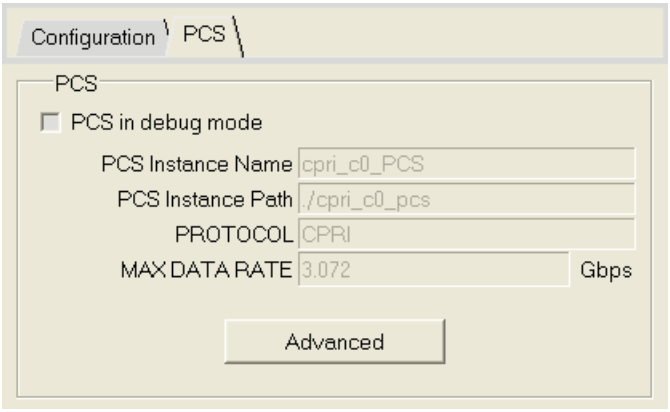


Figure 3-2 shows the CPRI PCS Configuration dialog box, it is for ECP5 CPRI IP Core only. The page 'PCS' is for ECP5UM PCS/SERDES settings. If the **PCS in debug mode** option is selected, the ECP5 DCU interface is displayed in <username>_phy_bb.v for debug. For further PCS configuration, click the **Advanced** button.

Figure 3-2. CPRI PCS Configuration Dialog Box

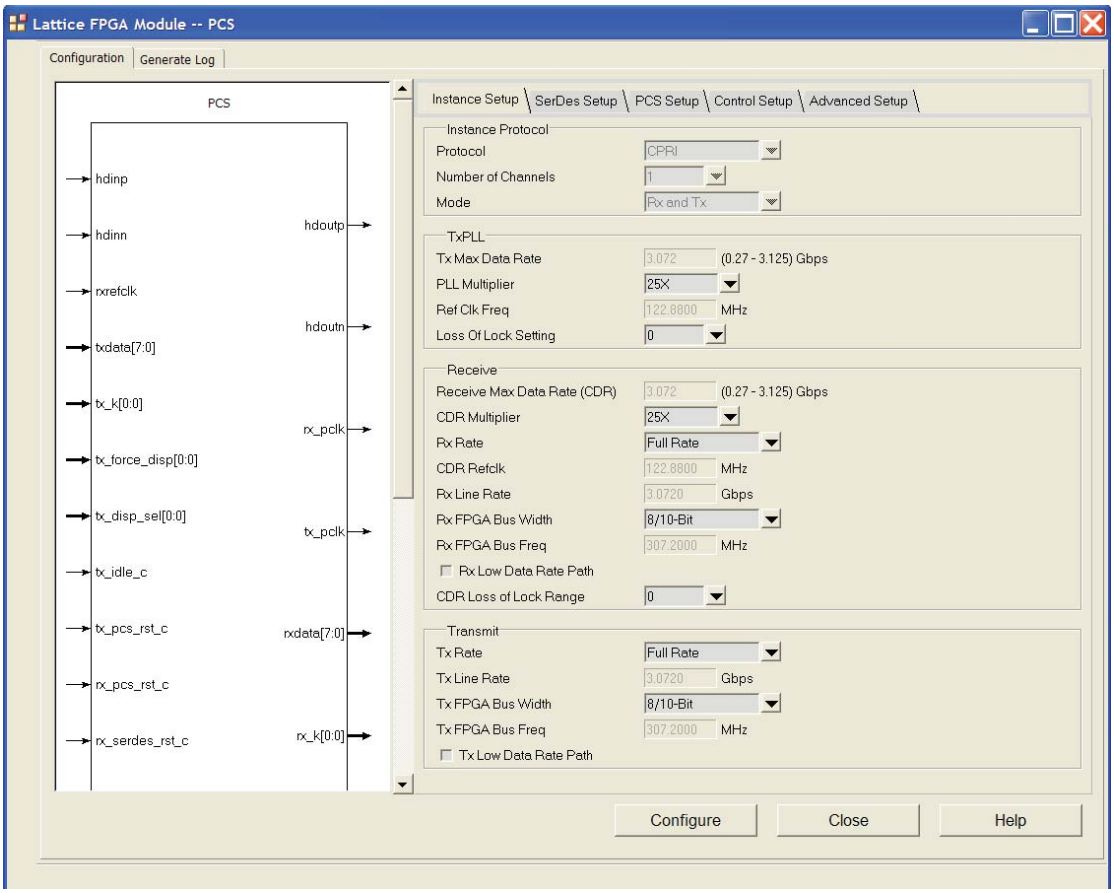


Clicking the **Advanced** button opens the CPRI PCS Advanced Configuration Dialog Box shown in Figure 3-3.

If only one PCS is used in the project, select the **Reset Sequence Select** option in Control Setup tab.

If you want to revise the SERDES electric character, change the parameters on the SerDes Setup tab. Refer to TN1261, ECP5 SERDES/PCS Usage Guide.

Figure 3-3. CPRI PCS Advanced Configuration Dialog Box



Generation Options

Design Entry

This option allows the user to specify support for either the “basic” core configuration or “low latency” core configuration. The “basic” core configuration implements all of the capabilities required to support the physical layer of the CPRI specification, except specific requirements related to link delay accuracy. The “low latency” core configuration is equivalent to the basic configuration. This configuration, however, includes a modified SERDES/PCS interface that supports the ability to manage the variability in the absolute latency for data transmission through the core. This is to meet the stringent CPRI link delay accuracy requirements. The “basic” mode is not supported.

Ethernet Mode

This option allows the user to specify the Ethernet mode supported by the IP core, either Serial, Matched Rate MII or 100Mb/s Fixed Rate MII. See “Ethernet Interface” on page 18 for a detailed description of these different modes.

Eval Configuration

Synthesis Tool

This option specifies evaluation configuration synthesis tool support for Synplify.

Programmable Parameters

The configuration settings listed in Table 3-2 are controlled via user-specified input signals to the IP core.

Table 3-2. CPRI Parameters Controlled via Input Signals to the IP Core

Input Signal	Values	Function
lbr_en[1:0]	0, 1, 2, 3	Selects the maximum line bit rate which the IP core will support: 0 – 614.4 Mbps maximum 1 – 1228.8 Mbps maximum 2, 3 – 2457.6 Mbps maximum 3 – 3072 Mbps in 3G version
force_sm_standby	0, 1	A zero-to-one transition forces the startup state machine to the Standby state.
pcs_serdes_rate[1:0]	0, 1, 2, 3	Selects the line bit rate which the PCS/SERDES will support: 0 – 614.4 Mbps maximum 1 – 1228.8 Mbps maximum 2 – 2457.6 Mbps maximum 3 – 3072 Mbps in 3G version
tx_eth_pointer[7:0]	0-63	Selects fast C&M pointer value
hdlc_240_en	0, 1	Enables 240 KHz as the maximum HDLC interface frequency
hdlc_480_en	0, 1	Enables 480 KHz as the maximum HDLC interface frequency
hdlc_960_en	0, 1	Enables 960 KHz as the maximum HDLC interface frequency
hdlc_1920_en	0, 1	Enables 1920 KHz as the maximum HDLC interface frequency
hdlc_2400_en	0, 1	Enables 2400 KHz as the maximum HDLC interface frequency
auto_cnt	0, 1	Enable master CPRI tx hfn and bfn updating. Default 1.
hyp_cnt_init	0-255	hfn used to update the control words or initialize the internal hfn counter. Default 0.
bfn_cnt_init	0-4095	bfn used to update the control words or initialize the internal bfn counter. Default 0.
tx_hyp_rst_en	0, 1	Enables tx_sync to reset tx_hyp_num
tx_bfn_rst_en	0, 1	Enables tx_sync to reset tx_bfn_num
test_md	0, 1	Speeds up timer to reduce test time: 0 – Normal 1 – Test
rec_md	0, 1	Sets core to REC or RE: 0 – RE 1 – REC

IP Core Generation

This chapter provides information on how to generate the CPRI IP core using the Diamond or ispLEVER software IPexpress tool, and how to include the core in a top-level design.

Licensing the IP Core

An IP core- and device-specific license is required to enable full, unrestricted use of the CPRI IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/Products/DesignSoftwareAndIP.aspx

Users may download and generate the CPRI IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The CPRI IP core also 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 time (approximately four hours) without requiring an IP license. See "Hardware Evaluation" on page 36 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond and ispLEVER EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

Getting Started

The CPRI IP core is available for download from the Lattice IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the CPRI IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be loaded.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL.
- **Device Family** – Device family to which IP is to be targeted (e.g. LatticeSCM, Lattice ECP2M, LatticeECP3, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

Figure 4-1. IPexpress Dialog Box (Diamond Version)

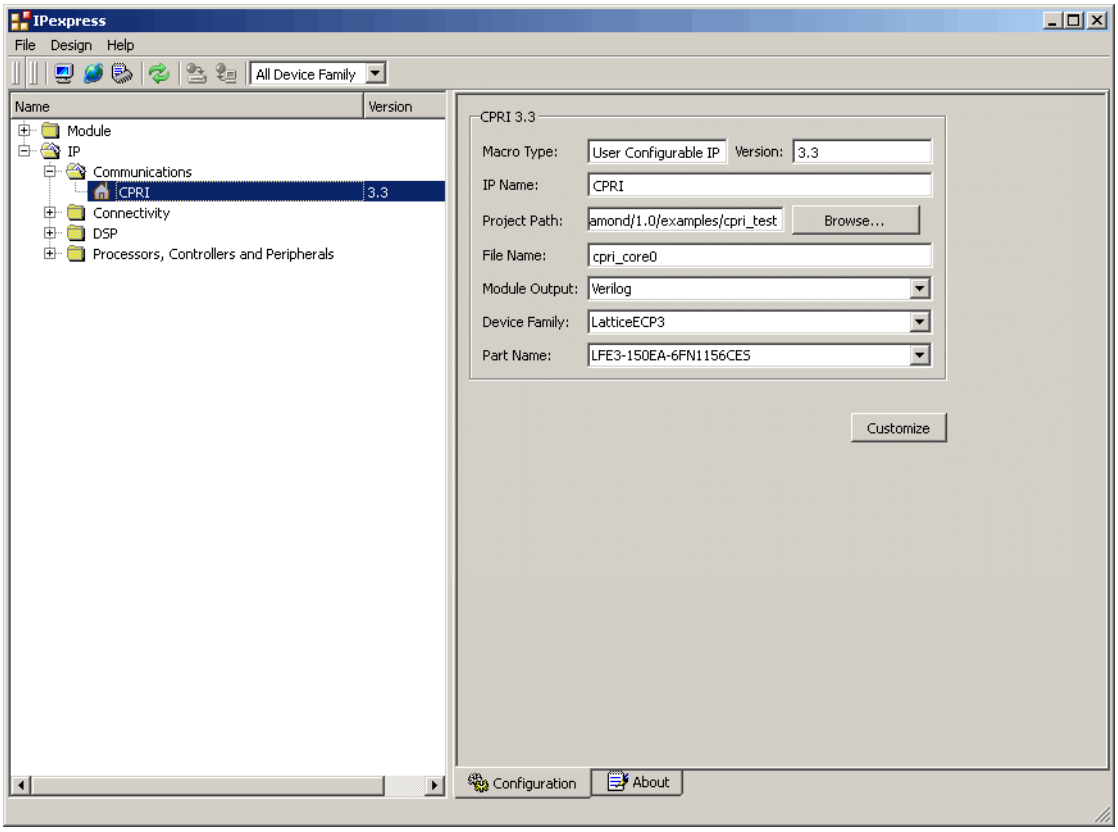
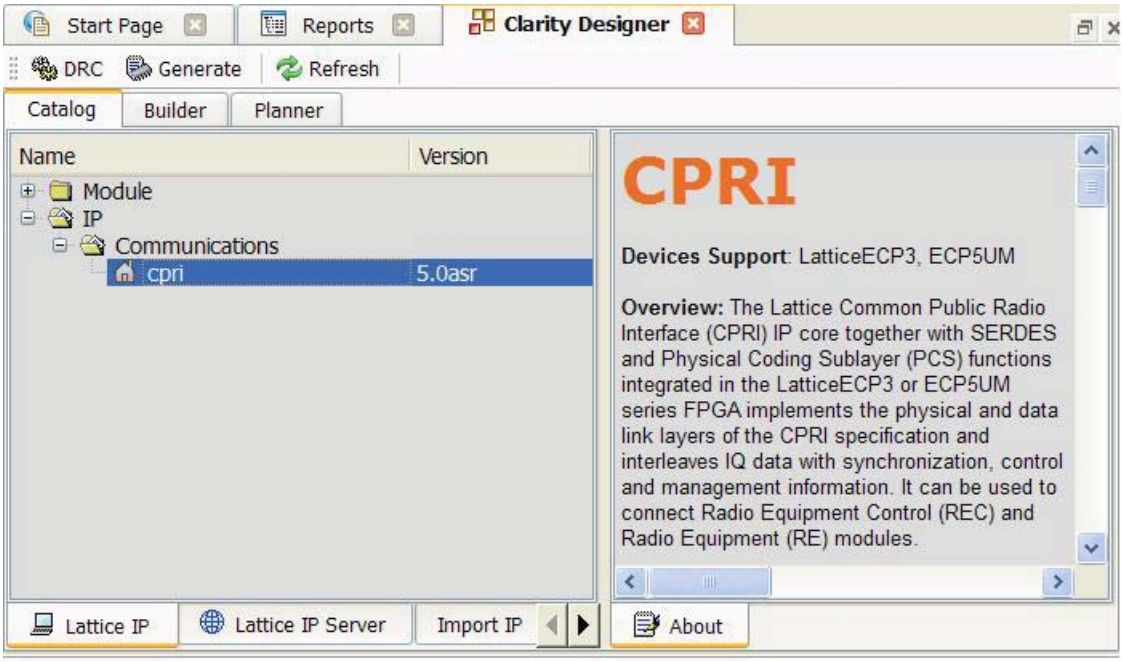


Figure 4-2. Clarity Dialog Box (Diamond Version)

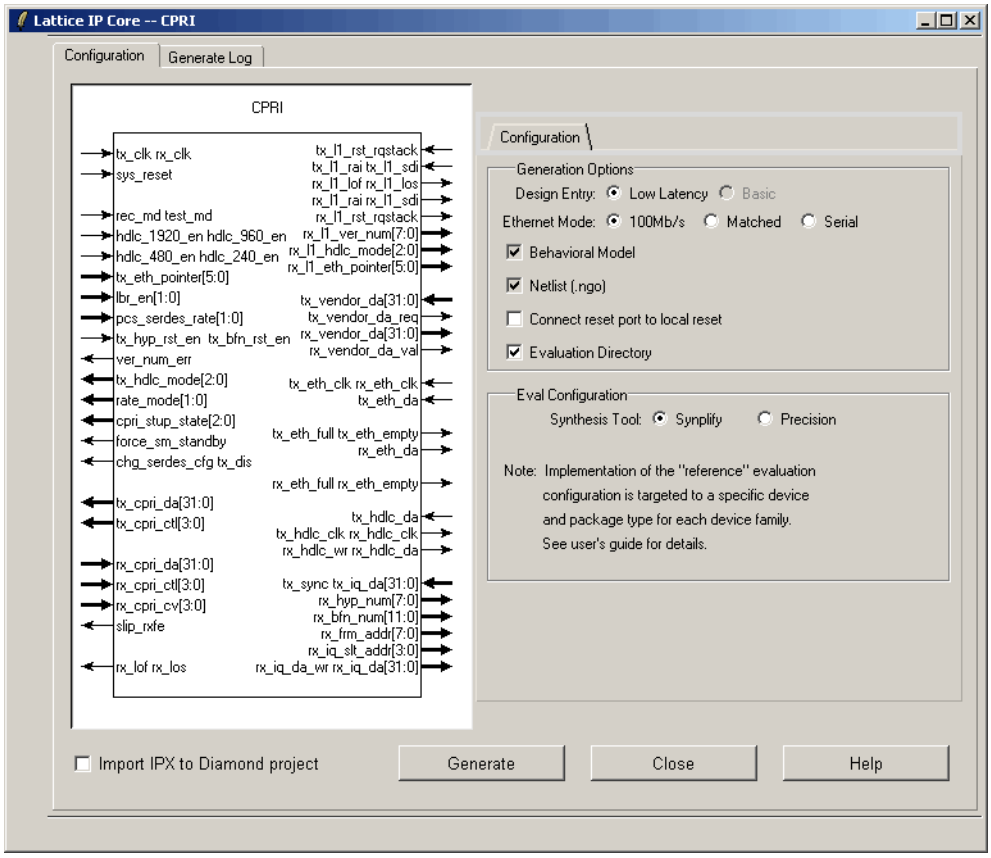


Note that if the IPexpress for LatticeECP3 or the Clarity Designer for ECP5 is called from within an existing project, Project Path, Module Output, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration:

1. Click the **Customize** button in the IPexpress tool dialog box to display the CPRI IP core Configuration GUI, as shown in Figure 4-3.
2. Select the IP parameter options specific to their application. Refer to "Parameter Settings" on page 24 for more information on the CPRI IP core parameter settings.

Figure 4-3. Configuration GUI (Diamond Version)



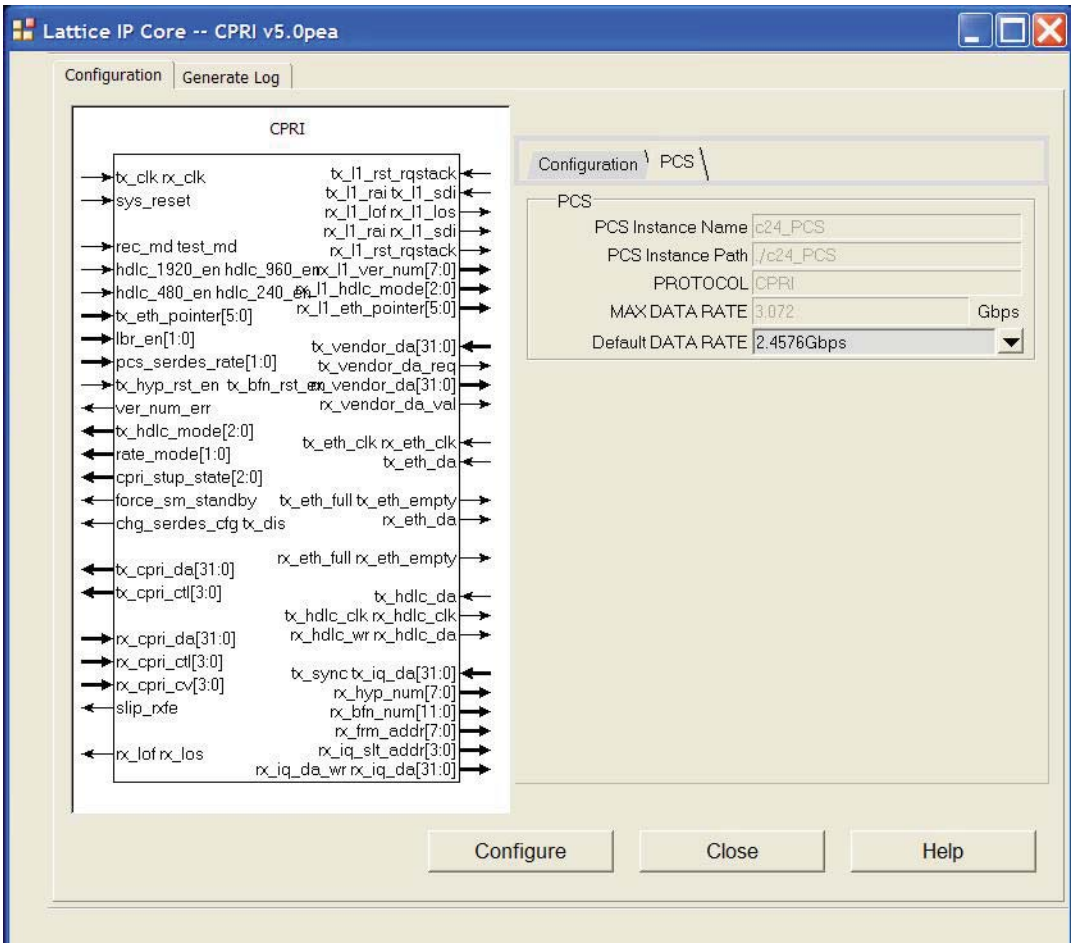
For ECP5 CPRI IP core, the core should be generated using the Diamond System Planner.

To generate the ECP5 CPRI IP Core in System Planner:

1. Create a new project with an ECP5 device.
2. Open System Planner and double-click **CPRI IP core** to open the CPRI IP GUI.
3. Configure the parameters. Note that there is one more package PCS.
4. Locate the DCU Channel into FPGA in the System Planner GUI.
5. Click the **Generate** button.

The PCS page is shown in Figure 4-4.

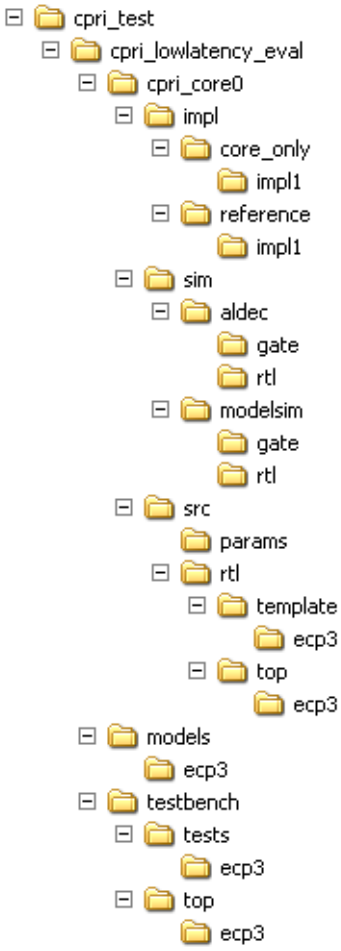
Figure 4-4. PCS User Interface



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified “Project Path” directory. The directory structure of the generated files is shown in Figure 4-5.

Figure 4-5. LatticeECP3 CPRI IP Core Directory Structure



The directory structure of ECP5 CPRI IP core, as shown in Figure 4-6, is different from LatticeECP3.

Figure 4-6. ECP5 CPRI IP Core Directory Structure

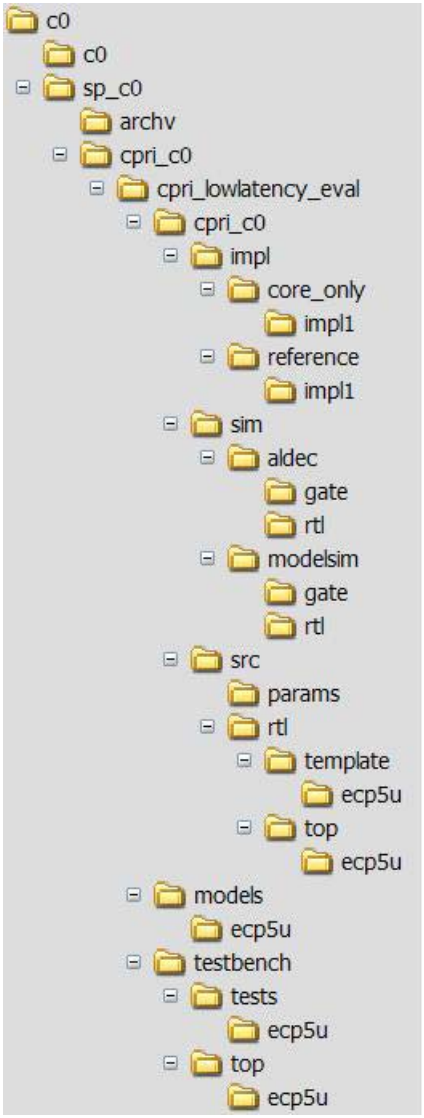


Table 4-1 provides a list of key files created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<*username*>_inst.v	This file provides an instance template for the IP.
<*username*>_bb.v	This file provides the synthesis black box for the user's synthesis.
<*username*>_beh.v	This file provides a behavioral simulation model.
<*username*>.ngo	This file provides the synthesized IP core.
<*username*>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.

Table 4-1. File List (Continued)

File	Description
<*username*>.ipx	IPexpress package file (Diamond only). This is a container that holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing this file to the associated Diamond project.
<username_PCS.lpc>	This is for CPRI PCS/SERDES model generation. It is located at username\username_PCS\ folder. (For ECP5 only.)

These are all of the files necessary to implement and verify the CPRI IP core in your own top-level design. The following additional files providing IP core generation status information are also generated in the "Project Path" directory:

- **<username>_generate.log** – Diamond or ispLEVER synthesis and map log file.
- **<username>_gen.log** – IPexpress IP generation log file.

Depending on whether the user has selected the basic or low latency configuration option, either the **\cpri_eval** or **\cpri_lowlatency_eval** directory is also created, respectively. The **\<cpri_eval>** (or **\<cpri_lowlatency_eval>**) and subtending directories provide files supporting the CPRI IP core evaluation. The **\<cpri_eval>** and **\cpri_lowlatency_eval** directories contains files/folders with content that is constant for all configurations of the CPRI IP core. The **\<username>** subfolder contains files/folders with content specific to the username configuration.

The **\cpri_eval** and **\cpri_lowlatency_eva**l directories are 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. **\<my_core_0>**, **\<my_core_1>**, etc.

Instantiating the Core

The generated CPRI 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. Two example RTL toplevelreference source files are provided in **\<***project_dir***>\cpri_eval\<***username***>\src\rtl\top**.

The top-level file cpri_reference_top.v is the same top-level that is used in the simulation model described in the next section. Users may use this top-level reference as the starting template for the top level for their complete design. Included in cpri_reference_top.v are logic, memory and clock modules supporting a driver/monitor module capability, a register module supporting programmable control of the CPRI core and system processor interface via the LatticeSC integrated SYSBUS capability. Verilog source RTL for these modules are provided in **\<***project_dir***>\cpri_eval\<***username***>\src\rtl\template**. The top-level configuration is specified via the parameters defined in the cpri_defines.v file in **\<***project_dir***>\cpri_eval\<***username***>\src\params**. A description of the CPRI register layout for this reference design is provided in "Register Descriptions" on page 41.

The top-level file cpri_CORE_ONLY_top.v supports the ability to implement just the CPRI core itself. This design is intended only to provide an accurate indication of the device utilization associated with the CPRI core and should not be used as an actual implementation example. To instantiate this IP core using the Linux platform, users must manually add one environment variable named "SYNPLIFY" to indicate the installation path of the OEM Synplify tool in the local environment file. For example,
SYNPLIFY=/tools/local/isptools8_1/isptools/synplify_linux.

Running Functional Simulation

The functional simulation includes a configuration-specific behavioral model of the CPRI IP Core (<*username*>_ beh.v) that is instantiated in an FPGA top level along with a user-side driver/monitor module and register implementation module. The top-level file supporting ModelSim eval simulation is provided in **\<***project_dir***>\cpri_eval\<***username***>\src**. This FPGA top is instantiated in an eval testbench provided in **\<***project_dir***>\cpri_eval\testbench** that configures FPGA test logic registers and CPRI IP core control and status registers via an included test file testcase.v provided in **\<***project_dir***>\cpri_eval\testbench** tests. Note the user can edit the testcase.v file to configure and monitor whatever registers they desire.

Users may run the eval simulation by doing the following.

Using Aldec Active-HDL

1. Open Active-HDL.
2. Under the Tools tab, select Execute Macro.
3. Browse to folder **\<***project_dir***>\cpri_eval\<***username***>\sim\aldec** and execute on of the “do” scripts shown.

Using Mentor Graphics ModelSim

1. Open ModelSim.
2. Under the File tab, select Change Directory and choose folder **\<***project_dir***>\cpri_eval\<***username***>\sim\modelsim.**
3. Under the Tools tab, select Execute Macro and execute one of the “do” scripts shown.

The simulation waveform results will be displayed in the simulator Wave window.

Three different evaluation simulations are provided with the CPRI IP core:

- Eval simulation using rate 614.4 MHz as the line rate
- Eval simulation using rate 1228.8 MHz as the line rate.
- Eval simulation using rate 2457.6 MHz as the line rate.
- Eval simulation using rate 3072 MHz as the line rate.

All of the simulations configure the CPRI and PCS/SERDES registers and wait for the receiver to synchronize with the incoming link. The following tests are then run:

1. Verify that the receiver has no IQ data errors for one frame.
2. Verify that the receiver has no vendor specific data errors for one frame.
3. Verify that the receiver has no Ethernet data errors for one frame.
4. Verify that the receiver has no HDLC data errors for one frame.

The procedure for running gate-level timing simulation is equivalent to that for running RTL functional simulation. After opening Active-HDL or ModelSim, the user executes one of the “do” scripts in the corresponding \sim\aldec\gate or \sim\modelsim\gate directory. Note that for VHDL gate netlist simulation, it is recommended to start with an empty work library.

Synthesizing and Implementing the Core in a Top-Level Design

The CPRI IP core itself is synthesized and is provided in NGO format when the core is generated. Users may synthesize the core in their own top-level design by instantiating the core in their top level as described previously and then synthesizing the entire design with either Synplicity or Precision RTL Synthesis.

Two example RTL top-level configurations supporting CPRI core top-level synthesis and implementation are provided with the CPRI IP core in **\<*project_dir*>\cpri_eval\<*username*>\impl**.

The top-level file cpri_core_only_top.v provided in **\<*project_dir*>\cpri_eval\<*username*>\src\rtl\top** supports the ability to implement just the CPRI core. This design is intended only to provide an accurate indication of the device utilization associated with the core itself and should not be used as an actual implementation example. The top-level file cpri_reference_top.v provided in **\<*project_dir*>\cpri_eval\<*username*>\src\rtl\top** supports the ability to instantiate, simulate, map, place and route the Lattice CPRI IP core in a complete example design. This reference design basically provides a CPRI driver/monitor on the client side of the core in an FPGA top-level file that contains the CPRI IP core and the driver/monitor. This is the same configuration that is used in the evaluation simulation capability described previously.

Note that implementation of the reference evaluation configuration is targeted to a specific device and package type for each device family (LatticeECP3 and ECP5). Specifically:

- LatticeECP3: LFE3-95E-7FN1156CES
- ECP5: LFE5UM-85F-7MG381C

Push-button implementation of both top-level configurations is supported via the ispLEVER project files, <*username*>_reference_eval.syn and <*username*>_core_only_eval.syn. These files are located in **\<*project_dir*>\cpri_eval\<*username*>\impl\<*configuration*>**.

For the Linux platform, the top-level synthesis must be run separately with a synthesis tool such as Synplify Pro, since Diamond and ispLEVER for Linux only accepts an EDIF entry. For the core only project, synthesis tcl files will be generated for this purpose, including cpri_core_only_top.tcl in the directory **\<*project_dir*>\cpri_eval\<*username*>\impl\core_only\syn_eval**.

The user can type the following command to synthesize the core_only top_level files in the above directory:

```
synplify_pro batch cpri_core_only_top.tcl
```

For the reference project, cpri_reference_top.tcl will be generated in the **\impl\reference\syn-eval** directory.

The synthesize command is:

```
synplify_pro –batch cpri_reference_top.tcl
```

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to **\<*project_dir*>\cpri_eval\<*username*>\impl\synplify (or precision)** in the Open Project dialog box.
3. Select and open <*username*>.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.

4. Select the **Process** tab in the left-hand GUI window.

5. Implement the complete design via the standard Diamond GUI flow.

To use this project file in ispLEVER:

1. Choose **File > Open Project.**
2. Browse to \<*project_dir*>**cpri_eval**\<*username*>**impl\synplify (or precision)** in the Open Project dialog box.
3. Select and open <*username*>.syn. At 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.

Implement the complete design via the standard ispLEVER GUI flow.

Hardware Evaluation

The CPRI 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 of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate.** The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.
9. Check the Generate Log tab to check for warnings and error messages.

10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Regenerating an IP Core in IPexpress Tool

To regenerate an IP core in IPexpress:

1. In IPexpress, choose **Tools > Regenerate IP/Module**.
2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core to regenerate, and click **Open**.
3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.
4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.
5. Click **Next**. The IP core's dialog box opens showing the current option settings.
6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.
7. Click **Generate**.
8. Click the Generate Log tab to check for warnings and error messages.

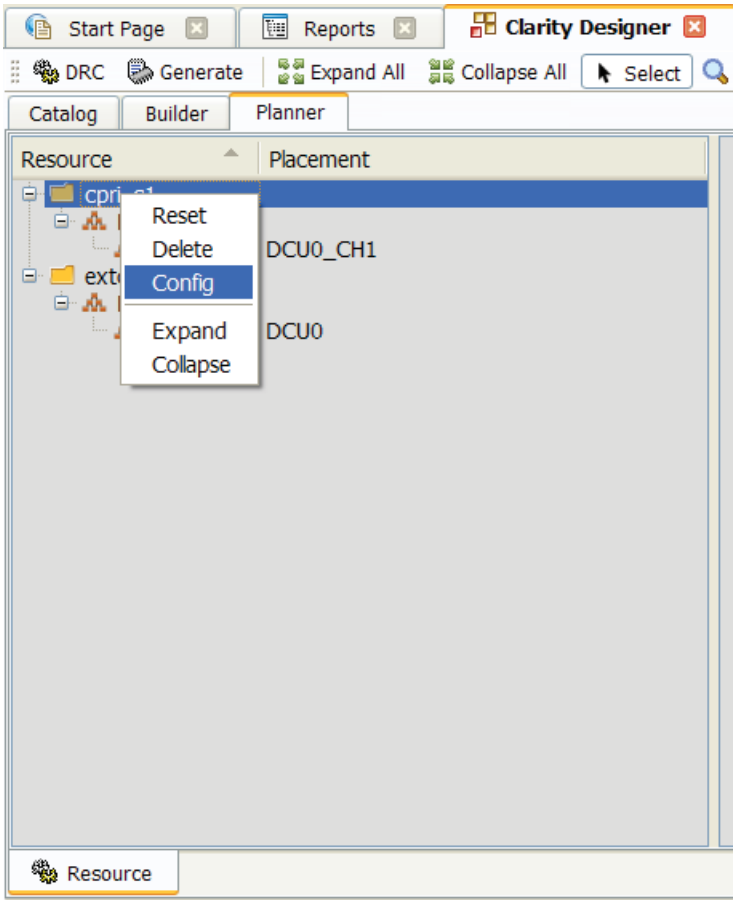
Regenerating an IP Core in Clarity Designer Tool

To regenerate an IP core in Clarity Designer:

Option 1

1. In the Clarity Designer Builder window, right-click on the existing IP instance and choose **Config**.

Figure 4-7. Clarity Designer Builder Window



2. In the dialog box, choose the desired options.

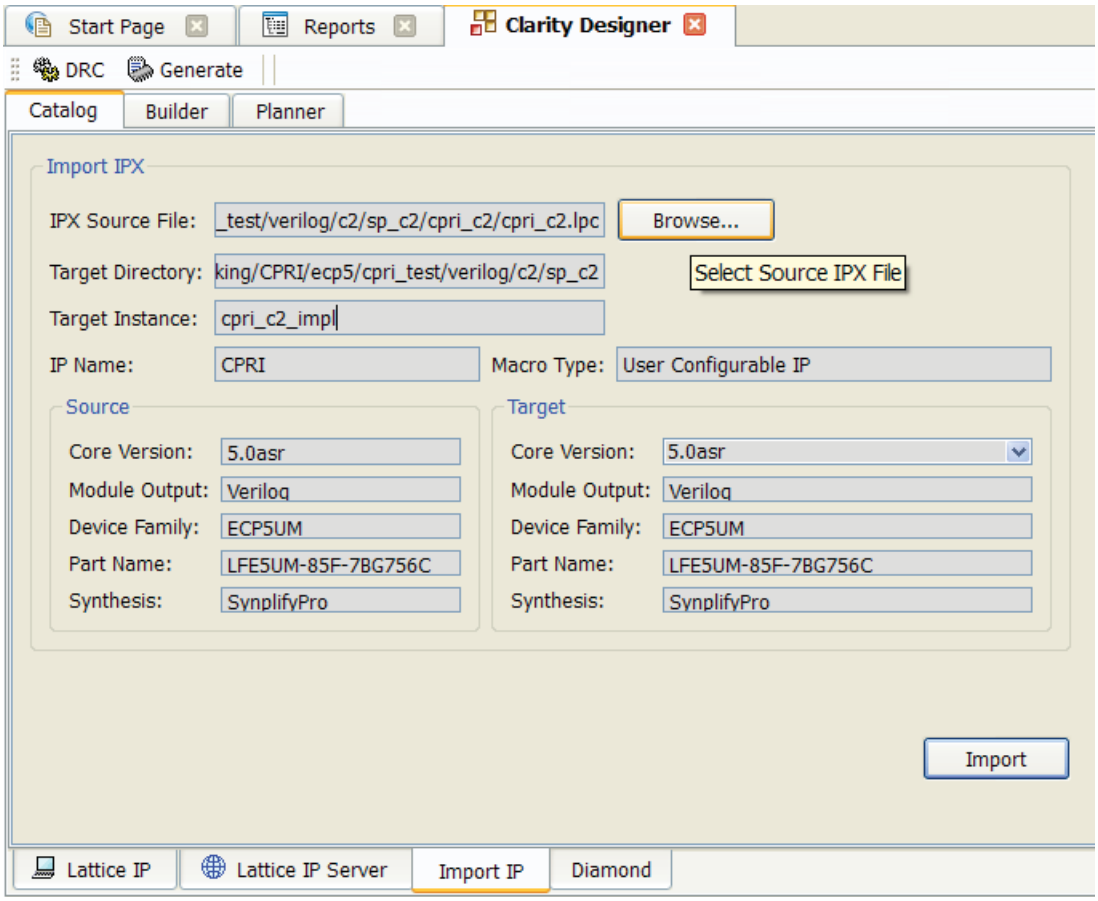
   For more information about the options, click **Help**. You may also click the **About** tab in the Clarity Designer window for links to technical notes and user guides. The IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.

3. Click **Configure**.

Option 2

1. In the Clarity Designer Catalog window, click the **Import IP** tab at the bottom.
2. In the Import IP tab, click **Browse** to choose the IPX/LPC source file of the module or IP to regenerate.

Figure 4-8. Clarity Designer Catalog Window



3. Specify the instance name in **Target Instance**. Note that this instance name should not be the same as any of the existing IP instances in the current Clarity Design project.
4. Click **Import**. The module's dialog box opens showing the option settings.
5. In the dialog box, choose the desired options.

   For more information about the options, click **Help**. You may also click the **About** tab in the Clarity Designer window for links to technical notes and user guides. The IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
6. Click **Configure**.

This chapter provides application support information for the CPRI IP core.

CPRI IP Basic Configuration Top-Level Reference Design

Note: A top-level reference design is also available for the low latency core configuration. This reference design is described in detail in IPUG74, CPRI IP Core Low Latency Variation Design Considerations User's Guide*.*

Included with the CPRI IP core is a reference top-level file, cpri_reference_top.v, that designers may use as the starting template for the top-level for their complete design. Included in cpri_reference_top.v are logic, memory and clock modules supporting a driver/monitor module capability, a register module supporting programmable control of the system processor interface. Verilog source RTL for these modules are provided in **\<*project_dir*>\cpri_eval\<*username*>\src\rtl\template**. The top-level configuration is specified via the parameters defined in the cpri_defines.v file in **\<*project_dir*>\cpri_eval\<*username*>\src\params**.

The constraints files provided are for a specific device, package type, and minimum speed grade for each device family supported. These values are shown in Table 5-1.

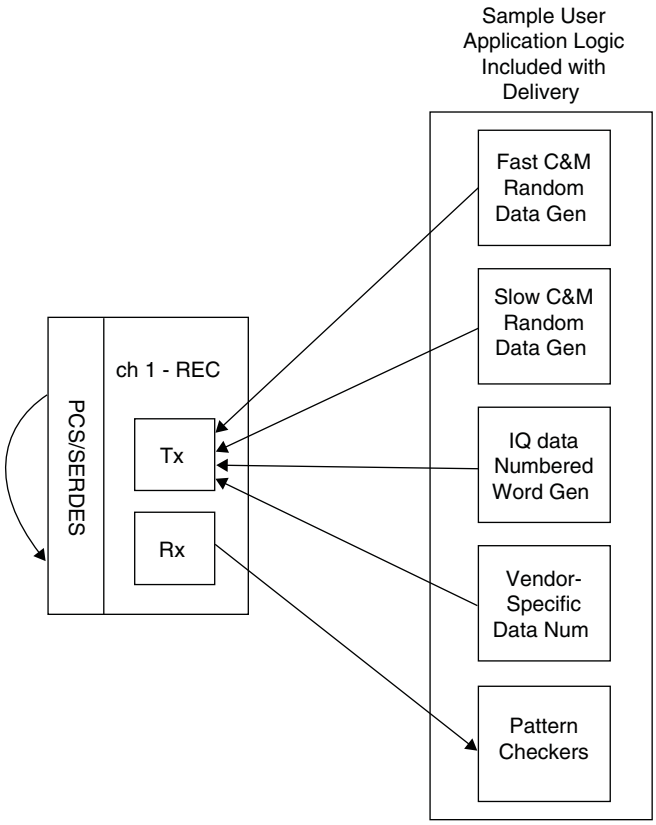
Table 5-1. Supported Devices for Reference Design

Device Family	Device	Package	Minimum Speed Grade
LatticeECP3	LFE3-95E	1156-ball fpBGA	-7
ECP5	LFE5UM-85F	756-ball fpBGA	-7

Test Bench

The template logic shown previously in Figure 2-4 on page 10 which is delivered with the IP core, contains the REC configuration. Along with this sample top level a sample user application is included. This user application is intended to be removed by the end customer and replaced with their own application. The user application logic, which is delivered in the template, includes circuitry which can be used to test the CPRI IP core. This circuitry consists of pseudo-random number generators for the fast and slow C&M channels, as well as circuits which insert numbered words into the user IQ data and the vendor specific channels. The delivery also includes a testbench that connects the top-level template logic as an REC interface, and has test data being sent from the transmitter to the receiver, as shown in Figure 5-1.

Figure 5-1. Pattern Generators and Checkers Included with Delivered IP Core Testbench (One Direction Shown)



Register Descriptions

There are no user accessible registers within the core. All control and status appear as internal I/O at the core boundary. Registers are provided at the top (template) level to drive and monitor all the control and status that interfaces with the core. These registers are shown in Table 5-2. Additional registers supporting the low latency core configuration are described in IPUG74, *CPRI IP Core Low Latency Variation Design Considerations User's Guide.*

Table 5-2. CPRI Testbench Register Map

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Control 0	0x800	4:0	lbr[1:0] – Line Bit Rate Control For maximum CPRI line rate enable by user FORCE_SM_STANDBY – Forces startup state machine to standby PCS_SERDES_RATE[1:0] – Line rate supported by PCS/SERDES	00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 2457.6 MHz zero to one transition 00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 2457.6 MHz
CPRI Control 1	0x801	1:0	TX_L1_SDI TX_L1_RAI	SAP defect indication and remote action indication

Table 5-2. CPRI Testbench Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Control 2	0x802	5:0	TX_BFN_RST_EN TX_HYP_RST_EN HDLC_1920_EN HDLC_960_EN HDLC_480_EN HDLC_240_EN	Enable = 1 Disable = 0
CPRI Control 3	0x803	5:0	TX_ETH_POINTER	Tx Ethernet pointer value
CPRI Control 4	0x804	0	TX_L1_RST_RQSTACK	Source for down link reset request or up link reset acknowledge
CPRI Error Reg 0	0x805	7:5,3:0	RX_LOF RX_LOS VER_NUM_ERR PRO_INTRUPT RX_ETH_FULL RX_ETH_EMPTY TX_ETH_FULL TX_ETH_EMPTY	Rx loss of frame Rx loss of signal Version number error bit Processor interrupt Ethernet FIFO error bits
CPRI Status Reg 0	0x806	1:0	RATE_MODE	Final negotiated line bit rate
CPRI Status Reg 1	0x807	7:0	RX_LOF RX_LOS VER_NUM_ERR RX_L1_LOF RX_L1_LOS RX_L1_SDI RX_L1_RAI RX_L1_RST_RQSTACK	Rx loss of frame Rx loss of signal Version number error bit Received L1 inband protocol bits of byte Z.130.0
VERSION Reg	0x808	7:0	RX_L1_VER_NUM	Received L1 inband protocol bits of byte Z.2.0
CPRI Status Reg 2	0x809	5:0	RX_L1_ETH_POINTER	Received L1 inband protocol bits of byte Z.194.0
CPRI Status Reg 3	0x80a	6:4, 2:0	RX_L1_HDLC_MODE[2:0] TX_HDLC_MODE[2:0]	Received L1 inband protocol bits of byte Z.66.0 Final negotiated HDLC bit rate
CPRI Status Reg 4	0x80b	2:0	CPRI_STUP_STATE	CPRI start up state
Test Control	0x80c	1:0	7 – Core reset, used if core reset port is not connected to GSR 1 – TEST_MD Test mode, 1 = short time 0 – REC_MD, REC mode, 1 = REC, 0 = RE	
Test Loop Data Error Reg	0x80d	3:0	Rx_hdlc_pdo_da_err Vendor_pdo_da_err Rx_iq_da_err Rx_eth_pdo_da_err	Test bench check loop back data errors

Table 5-2. CPRI Testbench Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
TB_CNTRL	0x80E	0	TB_CNTRL [0]	If 0: Check IQ data based on Numbered Word Gen. If 1: Fill data message with value in register TB_RXIQ
		1	TB_CNTRL [1]	If 0: Check VENDOR Data based on Data Num. If 1: Fill data message with value in register TB_RXVEND
		2	TB_CNTRL [2]	If 0: Check ETHR C/M based on random Data Gen. If 1: Fill ETHR C/M with value in register TB_RXETHR
		3	TB_CNTRL [3]	If 0: Check HDLC C/M based on random Data Gen. If 1: Fill HDLC C/M with value in register TB_RXHDLC
		4	TB_CNTRL [4]	If 0: Generate IQ data based on Numbered Word Gen. If 1: Fill data message with value in register TB_TXIQ
		5	TB_CNTRL [5]	If 0: Generate VENDOR data based on Data Num. If 1: Fill data message with value in register TB_TXVEND
		6	TB_CNTRL [6]	If 0: Generate ETHR C/M based on random Data Gen. If 1: Fill ETHR C/M with value in register TB_TXETHR
		7	TB_CNTRL [7]	If 0: Generate HDLC C/M based on random Data Gen. If 1: Fill HDLC C/M with value in register TB_TXHDLC
TB_TXHDLC	0x80F	7:0	TB_TXHDLC	This byte is used to fill HDLC messages when TB_CNTRL [7] is set to 1.
TB_TXETHR	0x810	7:0	TB_TXETHR	This byte is used to fill ETHR messages when TB_CNTRL [6] is set to 1.
TB_TXVEND	0x811	7:0	TB_TXVEND	This byte is used to fill VEND messages when TB_CNTRL [5] is set to 1.
TB_TXIQ	0x812	7:0	TB_TXIQ	This byte is used to fill IQ messages when TB_CNTRL [4] is set to 1.
TB_RXHDLC	0x813	7:0	TB_RXHDLC	This byte is used to check HDLC messages when TB_CNTRL [3] is set to 1.
TB_RXETHR	0x814	7:0	TB_RXETHR	This byte is used to check ETHR messages when TB_CNTRL [2] is set to 1.
TB_RXVEND	0x815	7:0	TB_RXVEND	This byte is used to check VEND messages when TB_CNTRL [1] is set to 1.
TB_RXIQ	0x816	7:0	TB_RXIQ	This byte is used to check IQ messages when TB_CNTRL [0] is set to 1.
TB_HDLC_CNT_HB	0x900	7:0	TB_HDLC_CNT [15:8]	HDLC message bit error counter, high byte. A read of this register latches both bytes and clears the internal counter. The internal counter freezes at maximum count.
TB_HDLC_CNT_LB	0x901	7:0	TB_HDLC_CNT [7:0]	HDLC message bit error counter, low byte.
TB_ETHR_CNT_HB	0x902	7:0	TB_ETHR_CNT [15:8]	ETHR message bit error counter, high byte. A read of this register latches both bytes and clears the internal counter. The internal counter freezes at maximum count.
TB_ETHR_CNT_LB	0x903	7:0	TB_ETHR_CNT [7:0]	ETHR message bit error counter, low byte.

Table 5-2. CPRI Testbench Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
TB_VEND_CNT_HB	0x904	7:0	TB_VEND_CNT [15:8]	VEND message bit error counter, high byte. A read of this register latches both bytes and clears the internal counter. The internal counter freezes at maximum count.
TB_VEND_CNT_LB	0x905	7:0	TB_VEND_CNT [7:0]	VEND message bit error counter, low byte.
TB_IQ_CNT_HB	0x906	7:0	TB_IQ_CNT [15:8]	IQ message bit error counter, high byte. A read of this register latches both bytes and clears the internal counter. The internal counter freezes at maximum count.
TB_IQ_CNT_LB	0x907	7:0	TB_IQ_CNT [7:0]	IQ message bit error counter, low byte.
TB_ERRINJ	0x817	0	TB_ERRINJ [0]	Inject an IQ message bit error each time this bit is changed from 0 to 1
		1	TB_ERRINJ [1]	Inject an VEND message bit error each time this bit is changed from 0 to 1
		2	TB_ERRINJ [2]	Inject an ETHR message bit error each time this bit is changed from 0 to 1
		3	TB_ERRINJ [3]	Inject an HDLC message bit error each time this bit is changed from 0 to 1.
		7:4	UNUSED	UNUSED
SUBCH_SAMPLE	0x818	7:0	SUBCH_SAMPLE	The CPRI Design will sample bit 0 of 0x00818 and initiate a memory sample of all the 64 subchannels when that bit is 1. When the CPRI Design memory sample is done, the CPRI Design will clear 0x00818.
SUBCH_MEM	0x819	7:0	SUBCH_MEM	When the CPRI Design clears 0x00818, the first byte (OFFSET=0) of the sampled 1024 SUBCHANNEL BYTES will be available at 0x819. Once a read access is performed on 0x819 to sample OFFSET Byte 0, the next available byte (OFFSET=1) will be available at 0x819 for the subsequent read access. Subsequent read accesses will access the subsequent bytes of SUBCH_MEM.
TB_ETH_IDLE_HB	0x81A	5:0	TB_ETH_IDLE_SIZE_HB	Testbench Ethernet Idle size High Byte - This is the upper 6 bits of a constant used to set the number of nibbles of idle in the ethernet message generated by the testbench.
TB_ETH_IDLE_LB	0x81B	7:0	TB_ETH_IDLE_SIZE_LB	Testbench Ethernet Idle size Low Byte - This is the lower 8 bits of a constant used to set the number of nibbles of idle in the ethernet message generated by the testbench.
TB_ETH_DATA_HB	0x81C	5:0	TB_ETH_DATA_SIZE_HB	Testbench Ethernet Data size High Byte - This is the upper 6 bits of a constant used to set the number of nibbles of data in the ethernet message generated by the testbench.
TB_ETH_DATA_LB	0x81D	7:0	TB_ETH_DATA_SIZE_LB	Testbench Ethernet Data size Low Byte - This is the lower 8 bits of a constant used to set the number of nibbles of data in the ethernet message generated by the testbench.

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

E-mail Support

techsupport@latticesemi.com

Local Support

Contact your nearest Lattice sales office.

Internet

www.latticesemi.com

IEEE

IEEE offers publications and technology standards on its web site at http://www.ieee.org.

References

Complete details on the CPRI IP core low latency configuration are given in:

- IPUG74, CPRI IP Core Low Latency Variation Design Considerations User's Guide

LatticeECP3

- HB1009, LatticeECP3 Family Handbook

ECP5

- HB1012, LatticeECP3 Family Handbook

Revision History

Date	Document Version	IP Core Version	Change Summary
September 2006	01.0	1.0	Initial release.
January 2007	01.1	2.0	Updated to include LatticeECP2M.
April 2007	01.2	2.2	Input lbr_en was expanded from 2 bits to 4 bits. The additional three bits provide two functions. Function 1: Adds the ability to force the Startup stater machine to the standby state. Function 2: Adds the ability to tell the core which rate the PCS/SERDES supports. These functions were added to make the rate selection process work properly.
July 2007	01.3	2.3	Updated LatticeSC/M and LatticeECP2M/S appendices.
			Added support for LatticeECP2MS.
September 2007	01.4	2.4	References to tx_eth_pointer[7:0] replaced with tx_eth_pointer[5:0].
May 2008	01.5	2.7	Added information for txrst and rxrst to Signal Descriptions table.
			Added information about using this IP core with the Linux operating system.
June 2008	01.6	2.7	Included information for Aldec, ModelSim, mixed mode, and rtl/gate simulation.
			Added information for low latency variation IP configuration.
August 2008	01.7	2.7	Updated Introduction section to include information regarding the v3.0 CPRI specification, requirement R-31 (line rate auto-negotiation).
			Removed the Core Generation text section.
			Removed the Instantiating the Core text section.
			Removed the Running Functional Simulation text section.
			Removed the Synthesizing and Implementing the Core in a Top-Level Design text section.
			Removed the Hardware Evaluation text section.
August 2008	01.8	2.7	Updated Rx IQ Interface Data and Frame Number Alignment diagram.
September 2008	01.9	2.7	Updated Introduction text section.
May 2009	02.0	3.0	Updated to include LatticeECP3 and 3G line rate support.
			Updated for enhanced fast C&M packet check for Fixed Ethernet mode, and support for back-to-back Ethernet packet in the CPRI link.
November 2009	02.1	3.2	Updated LatticeECP3 3G line rate TX HDLC support.
			Updated utilization tables with ispLEVER 8.0.
June 2010	02.2	3.2	Divided the document into chapters and added table of contents.
			Added Quick Facts tables in Chapter 1, “Introduction.”
			Added new content in Chapter 4, “IP Core Generation.”
September 2010	02.3	3.3	Added support for Diamond software throughout.
			Added new content in Chapter 4, “IP Core Generation.”
April 2014	02.4	5.0asr	Added support for Diamond 3.2.
			Added support for ECP5 device family.
			Updated document with new corporate logo.
			Updated Lattice Technical Support information.

Appendix A:

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the CPRI IP core.

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

LatticeECP3 FPGAs

Table A-1. Performance and Resource Utilization[1]

Mode	SLICEs	LUTs	Registers	sysMEM EBRs
Serial (Low Latency)	1139	1359	1522	4
Matched (Low Latency)	1342	1714	1632	2
Fixed (Low Latency)	1433	1848	1691	6

1. Performance and utilization data are generated targeting an LFE3-95E-7FN1156CES device using Lattice Diamond 1.0 and Synplify Pro for Lattice D-2009.12L-1 software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.

Ordering Part Number

The Ordering Part Number (OPN) for the CPRI IP core targeting LatticeECP3 devices is CPRI-E3-U4.

ECP5 FPGAs

Table A-2. Performance and Resource Utilization[1]

Mode	SLICEs	LUTs	Registers	sysMEM EBRs
Serial (Low Latency)	1235	1517	1480	4
Matched (Low Latency)	1452	1861	1590	2
Fixed (Low Latency)	1473	1980	1646	6

Ordering Part Number

The Ordering Part Number (OPN) for the CPRI IP core targeting LFE5UM devices is CPRI-E5-U and CPRI-E5-UT.

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Development Software ***category:***

Click to view products by Lattice ***manufacturer:***

Other Similar products are found below :

RAPPID-560XBSW RAPPID-567XFSW DG-ACC-NET-CD SRP004001-01 SW006021-1NH SW163052 SYSWINEV21 Core429-SA SW500006-HPA CWP-BASIC-FL W128E13 CWP-PRO-FL SYSMACSE210L SYSMACSE203L AD-CCES-NODE-1 NT-ZJCAT1-EV4 CWA-BASIC-FL RAPPID-567XKSW CWA-STANDARD-R SW89CN0-ZCC CWA-LS-DVLPR-NL VDSP-21XX-PCFLOAT RAPPID-563XMSW IPS-EMBEDDED SWR-DRD-L-01 SDAWIR-4532-01 SYSMAC-SE201L MPROG-PRO535E AFLCF-08-LX-CE060-R21 WS02-CFSC1-EV3-UP SYSMAC-STUDIO-EIPCPLR LIB-PL-PC-N-1YR-DISKID SYSMACSE2XXL LS1043A-SWSP-PRM 1120270005 1120270006 MIKROBASIC PRO FOR FT90X (USB DONGLE) MIKROC PRO FOR AVR (USB DONGLE LICENSE) MIKROC PRO FOR FT90X (USB DONGLE) MIKROBASIC PRO FOR AVR (USB DONGLE LICEN MIKROBASIC PRO FOR FT90X MIKROC PRO FOR DSPIC30/33 (USB DONGLE LI MIKROC PRO FOR FT90X MIKROC PRO FOR PIC32 (USB DONGLE LICENSE 52202-588 MIKROPASCAL PRO FOR ARM (USB DONGLE LICE MIKROPASCAL PRO FOR FT90X MIKROPASCAL PRO FOR FT90X (USB DONGLE) MIKROPASCAL PRO FOR PIC32 (USB DONGLE LI SW006021-2H