CPRI IP Core Low Latency Variation Design Considerations

User's Guide

November 2009
ipug74_01.5

Using the CPRI IP Core with LatticeECP2M FPGAs

Introduction

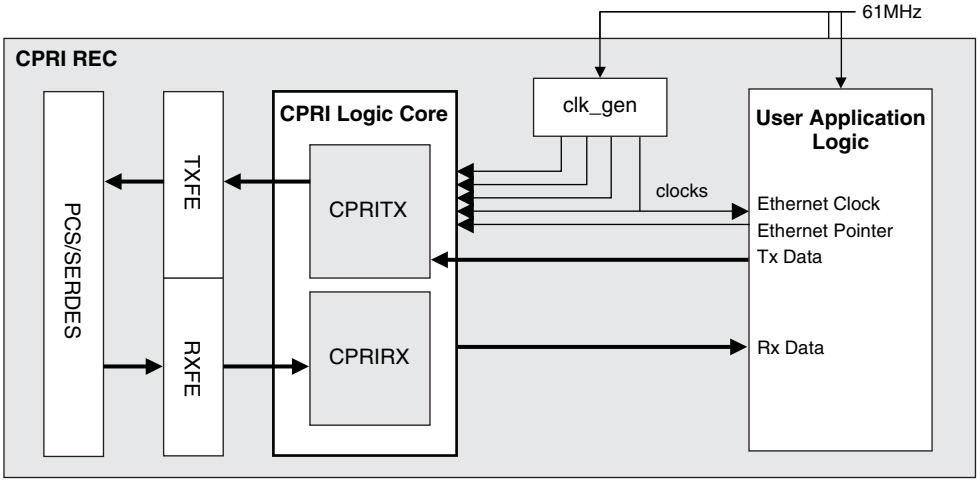
The Lattice CPRI IP core supports a low latency variation configuration that includes special design features enabling the creation of designs meeting stringent CPRI delay variation requirements. The "low latency" core configuration is equivalent to the basic configuration, but includes a modified SERDES/PCS interface that supports the ability to manage the variability in the absolute latency for data transmission through the core.

This document focuses on the detailed specifications associated with implementing and using the low latency CPRI configuration.

General Description

Figure 1 shows a system-level block diagram of the CPRI IP core. This block diagram shows the core configured to support REC applications. The CPRI core may also be configured to support RE applications. See the CPRI IP Core User's Guide for details. The complete CPRI IP core includes two key components, the CPRI IP logic core and separate logic blocks (RXFE and TXFE in Figure 1) that support the interface between the logic core and the SERDES/PCS functions integrated in the FPGA.

Figure 1. CPRI IP Core System-Level Block Diagram



The CPRI IP logic core is provided in NGO format and is identical for both the basic and low latency core configurations. However, the RXFE and TXFE logic blocks and the SERDES/PCS configuration differ significantly for the low latency configuration versus the basic configuration.

For the basic CPRI IP core configuration, the SERDES/PCS block is configured to support basic, complete 8b/10b functionality in both the transmit and receive directions. The SERDES/PCS configuration is controlled using a configuration txt file generated via IPexpress™ that specifies the settings of the SERDES block configuration memory cells that will be loaded via the FPGA program bitstream. No modifications to the SERDES configuration are required after the bitstream is loaded and the FPGA is functional. The RXFE and TXFE logic blocks for the basic configuration implement simple rate-adaptation functions between the CPRI logic and the SERDES/PCS block. These blocks are provided in RTL format in the basic configuration evaluation package included with the CPRI IP core.

For the low latency CPRI configuration, the SERDES/PCS block configuration is modified to bypass specific internal functions that insert latency variation in the data path. Where required, these bypassed functions are implemented in soft (FPGA) logic in the RXFE block where the latency variation can be tightly controlled or eliminated. The RXFE and TXFE blocks include additional specific capabilities to manage latency variation through the soft logic. Specific SERDES/PCS configuration and RXFE and TXFE block features supporting low latency variation include:

- The SERDES link high-speed clocks and core clocks are managed carefully to minimize latency variation due to clock routing delays and the associated variation across device PVT.
- The FPGA bridge FIFOs in the SERDES/PCS block are bypassed in both transmit and receive directions to eliminate the associated latency variation.
- The receive direction PCS word alignment function is implemented in soft logic, eliminating the unknown component of the latency variation associated with the SERDES deserializer 10-bit word alignment function.
- Automatic word aligner delay compensation is optionally supported, in which the phase of the receiver clock is adjusted based on the value of comma detection offset value so that the user always get a phase adjusted rx_clk_offset providing constant latency, aligned receive data.

The soft PCS functions are included in the RXFE block in NGO format. The remainder of the logic in the RXFE and TXFE logic blocks are provided in RTL format and may be used as-is or modified as necessary.

Note: Lattice supports the option of licensing only the soft PCS capability for users who wish to use it together with their own CPRI IP designs. Contact your Lattice sales representative for details.

The I/O signals for the CPRI IP logic core are described in detail in the CPRI IP Core User's Guide. The I/O signal descriptions for the soft PCS block used in the low latency configuration are given in Table 1.

Examples of RTL source supporting all of these capabilities on the LatticeECP2M35-672 device are provided in the low latency evaluation package included with the CPRI IP core. This evaluation package supports 1.2G and 2.4G line rate implementations.

Table 1. CPRI Soft PCS I/O

Signal Name	Direction	Width (Bits)	Description
rx_clk	I	1	61 MHz receive-side clock input
rst_n	I	1	Active low reset
rate_mode_2	I	1	1 = channel runs at 2.4G line rate 0 = channel runs at 1.2G line rate
loss_of_signal	I	1	Loss of signal indication from SERDES
force_realign	I	1	The rising edge will force the lsm out of sync
rxd_i	I	40	40-bit received data in 61MHz clock domain
rxd_o	O	32	32-bit data output after 8b/10b decoder
k_ctrl	O	4	4-bit control nibble indicating location of comma control character in a 32-bit word (bit 0 indicating lowest byte, bit 1 next byte, etc.)
code_violation	O	4	4-bit code violation nibble indicating location of code violation in a 32-bit word (bit 0 indicating lowest byte, bit 1 next byte, etc.)
rx_disp_err	O	1	1= disparity error
Sync_status	O	1	1= comma-based synchronization channel achieved
word_align_select	O	4	Word Align (WA) offset (0x0-0x9) 0 = comma word boundary aligned with recovered clock 1 = comma word boundary detected lagging recovered clock by 1/10th word (i.e. 1 line rate UI). 2 = comma word boundary detected lagging recovered clock by 2/10th word (i.e. 2 line rate UIs). ..... 9 = comma word boundary detected lagging recovered clock by 9/10th word (i.e. 9 line rate UIs). WA latency = (word_align_select) * UI bit period
lsm_state	O	4	External link state machine state

Functional Description

Figure 2 shows a block diagram of the key functionality implemented in the RXFE and TXFE blocks for the low latency CPRI configuration. As mentioned previously, the TXFE block is provided in RTL format while the RXFE

block includes a mix of RTL and NGO formats. When generating the CPRI IP core, all of the files supporting the RXFE and TXFE blocks are provided in the CPRI IP evaluation directory cpri_lowlatency_eval.

The TXFE block essentially performs rate adaptation between the 32-bit data interface of the transmit block in the CPRI logic core and the 8-bit data interface of the SERDES/PCS block. Careful management of the clocking for this block is key to achieving low latency variation.

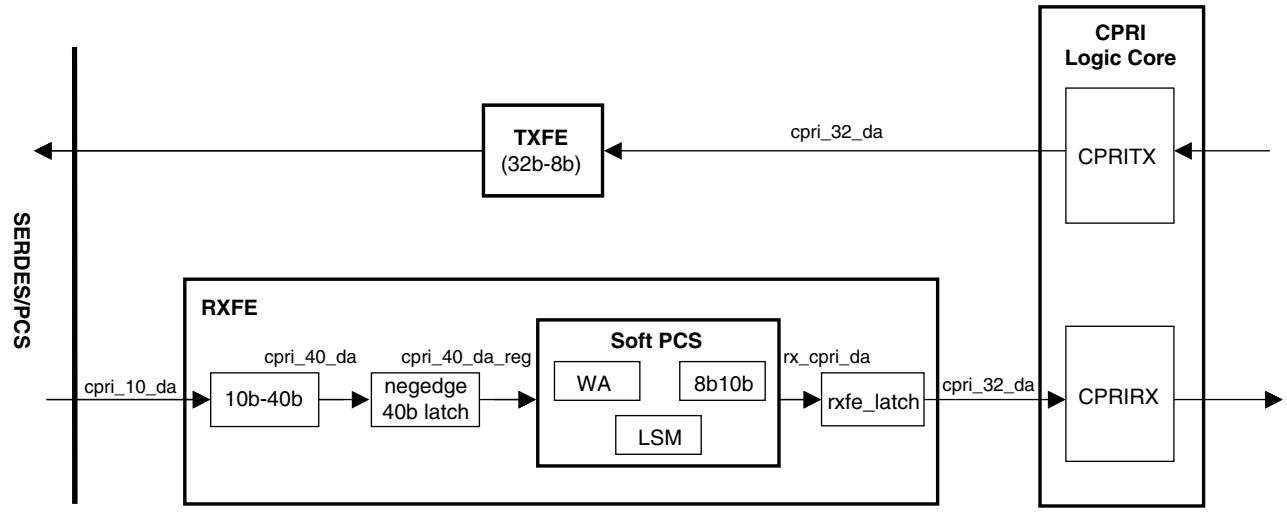
Key functions performed by the RXFE block include:

- Rate adaptation between the 10-bit interface to the SERDES/PCS and the 32-bit interface to the receive block in the CPRI logic core.
- 10-bit Word Alignment (WA)
- 10b/8b decoding

As indicated in Figure 2, the 10-bit receive data from the SERDES/PCS block is multiplexed into a 40-bit bus that interfaces with the soft PCS block, which performs the word alignment and 10/8b decoding. The output of the soft PCS interfaces with the CPRI logic core. Note that the K28.5 comma character (BCh) should always occur in the low byte of rx_cpri_da (b7:b0) with byte-level slip control. Complete details on this functionality can be observed by reviewing the corresponding files in the CPRI IP evaluation package. Again, careful management of the clocking for this block is key to achieving low latency variation.

The following sections provide details on how to configure the CPRI IP to support low latency variation.

Figure 2. RXFE and TXFE Logic Blocks



SERDES Configuration

To implement the low latency design, the SERDES/PCS is configured with a 10-bit interface and the basic functionality shown in Table 2.

Table 2. CPRI SERDES Configuration

Reference Clock Multiplier	10x	20x
Protocol	CPRI	CPRI
Bit Rate	1.2G	2.4G
8b/10b Interface		
ff_rxfullclk_ch[0:3]	122.88	245.76
ff_txser_fullclk	122.88	245.76
RX PCS		
Word Aligner	Bypass	Bypass
8b/10b Decoder	Bypass	Bypass
FIFO	Bypass	Bypass
TX PCS		
FIFO	Bypass	Bypass

Note: REFCLK = 122.88MHz

The SERDES/PCS configuration is initially specified using a configuration txt file generated via IPexpress. This file specifies the settings of the SERDES block configuration memory cells that will be loaded via the FPGA program bitstream. After the bitstream has been loaded and the FPGA is functional, the SERDES configuration must be modified via the LatticeECP2M™ SERDES Client Interface (SCI). Complete details on managing the SERDES configuration via the SCI are given in TN1124, LatticeECP2M SERDES/PCS Usage Guide.

The following details apply to the LatticeECP2M SERDES/PCS module configuration:

- When generating the SERDES/PCS configuration via IPexpress, the options for **Quad-based protocol** and **CPRI protocol mode** should be selected. With these settings, most of the parameters in the generated SERDES/PCS configuration txt file are set to the appropriate values for the low latency configuration.
- **TX 8b/10b encoder**: Set to bypass by default for the IPexpress CPRI protocol setting, this capability should be enabled for the low latency configuration. The SERDES TX 8b10b encoder is set to **NORMAL** by clearing the **enc_bypass** bit in per-channel control register **ch_02** (BA=2) for the corresponding SERDES via the SCI interface.
- **TX FIFO bypass**: Set to Normal by default for the IPexpress CPRI protocol setting, this capability should be set to **BYPASS** mode by setting the **tx_gear_bypass** bit in per-channel control register **ch_02** (BA=2) for the corresponding SERDES via the SCI interface.
- **RX FIFO bypass**: Set to Normal by default for the IPexpress CPRI protocol setting, this capability should be set to **BYPASS** mode by setting the **rx_gear_bypass** bit in per-channel control register **ch_03** (BA=3) for the corresponding SERDES via the SCI interface.
- **TX_SERSER_CLK**: This clock specifies the timing for the CPRI logic core TX block and SERDES TX interface. This clock is enabled by setting the **rx_ch** bit in per-channel control register **ch_01** (BA=1) for the corresponding SERDES via the SCI interface. Note that only one **rx_ch** bit may be enabled for multiple channel applications. Refer to the section on Multiple Channel Configuration Considerations for details on how to select this clock.
- **SERDES TX 8-bit Data Bus**: The SERDES/PCS interface is configured with a 10-bit interface. As discussed above, the TX 8b/10b encoder is enabled. The IP core sends eight data bits and one control bit to the SERDES/PCS. The mapping relationship between the eight data bits and one control bit from the IP core and the SERDES/PCS 10-bit port list is as follows:

ff_txdata_chi[9] -> 0
ff_txdata_chi[8] -> tx_cpri_ctl
ff_txdata_chi[7:0] -> tx_cpri_da[7:0]

Clock Configuration for High Link Delay Accuracy

Figure 3 shows the clock configuration for a core configured as a master. In this mode, the SERDES/PCS reference clock is generated by an oscillator and drives the SERDES/PCS. The SERDES/PCS generates recovered RX clock and TX clock signals. All other required data clocks are derived from the SERDES/PCS RX and TX output clocks for the RX and TX directions, respectively.

Figure 4 shows the clock configuration for a slave core. In this case, the SERDES/PCS reference clock (TX) is generated from the recovered clock that has been cleaned up using an external jitter-removal PLL. In both cases, the SERDES/PCS reference clock is 122.88MHz regardless of the line rate.

Some considerations:

- The RX high-speed recovered clock must be connected to the dedicated device clock tree via general routing. In the evaluation package, this clock is re-buffered using a PLL located very near to the SERDES/PCS. This configuration minimizes the routing delay and associated latency variation. Users may connect this clock to the primary tree via other means, such as via a CIB. For multi-channel applications, the user can also use secondary clock trees if there is contention for primary clock resources.
- The rx_clk and tx_clk signals are both generated from FFs and access the primary clock trees via CIBs. These FFs should be manually located to minimize overall routing delay and associated latency variations.
- Since the use of general routing impacts the clock routing variation, clock placement and routing results should be carefully checked to make sure the shortest clock routing paths are implemented. Differences in the specific design, device targeted and/or software version may yield different routing results. It is recommended that the design be checked in EPIC to verify the desired routing configuration has been achieved.
- The TXFE and part of the RXFE modules are involved in the data transfer between the SERDES/PCS 8-bit high-speed interface and the FPGA fabric. This transfer occurs at the full clock rate up to 245.76MHz. These modules should be PGROUPed and placed near the SERDES/PCS interface. The evaluation package provides an example for a LatticeECP2M35-672 device.

Figure 3. Clock Configuration for CPRI Master Application

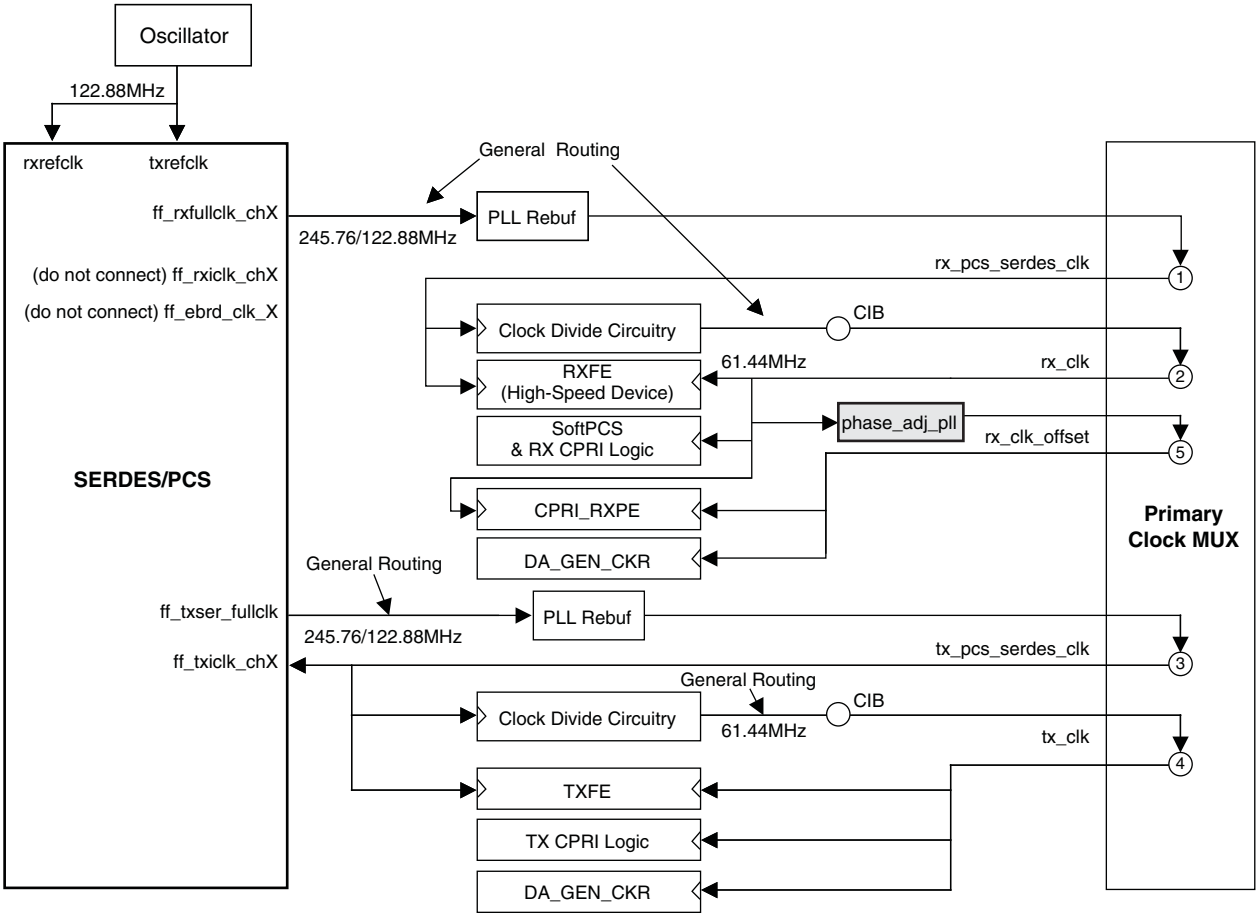
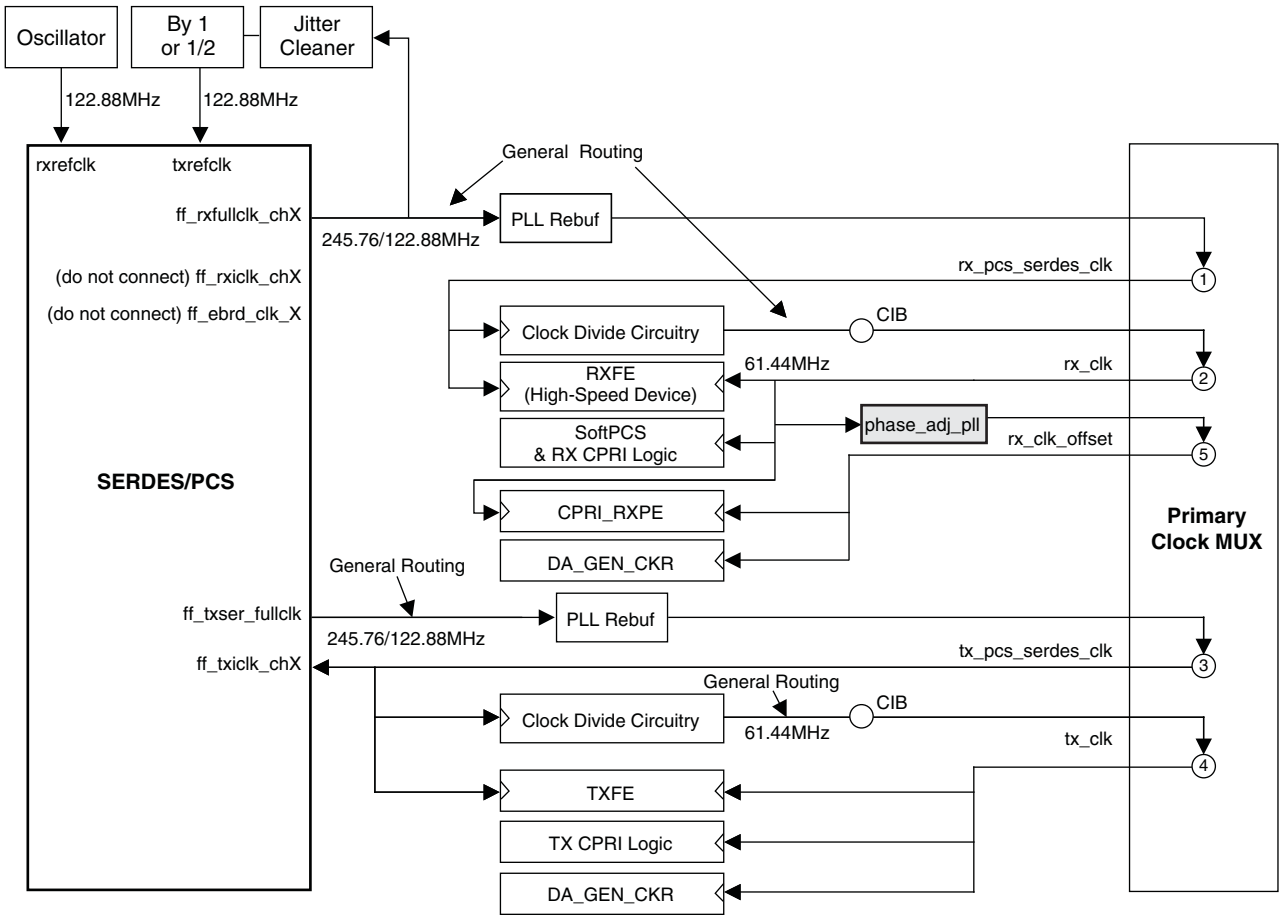


Figure 4. Clock Configuration for CPRI Slave Application

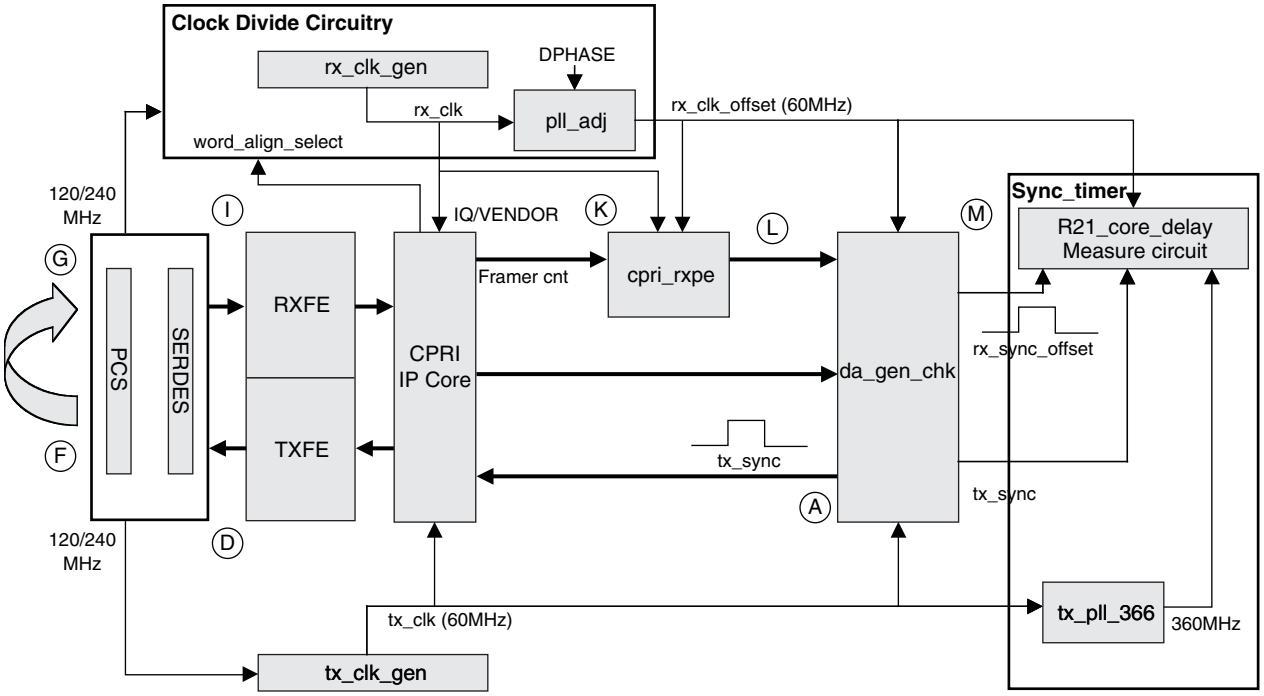


Core Delay Measurement

The delay through the CPRI IP core and associated SERDES/PCS can be modeled as shown in Figure 5. Delay though the FPGA has both a fixed logic component and a variable delay component. The variable delay is caused by clock routing.

The total delay can be divided into four parts: SERDES/PCS block delay, FPGA clock generation and routing delay, CPRI IP core delay, and soft PCS word aligner delay compensation. The total delay is defined as the time between the tx_sync and rx_sync_offset signals.

Figure 5. CPRI IP Delay Model



FPGA Clock Generation and Routing Delay

To calculate the total transceiver delay, in addition to the fixed delay through the SERDES/PCS and IP core, the clock routing variation for different devices and speed grades must also be considered. Tables 3 and 4 show results for the latency variation for the RX and TX user-side clock domains achieved for a LatticeECP2M35-672 device. These results are based on ispLEVER® TRACE tool analysis of the corresponding clock domains for worst case fast (setup) and slow (hold) performance. Since the PLL feedback compensates the primary clock routing from the PLL outputs to the flip-flop's CLK pin, the routing delay from PLL to FF CLK pin can be treated as at least 0. So the total rx_clk delay (for -6 speed grade device) is 3.106+/-9.945 ns between the fast and slow devices as shown in Table 3. Similarly, the total tx_clk delay is 2.447+/-0.752 ns as shown in Table 4.

Note that the clock delay value will vary depending on routing results. Designs with different clock routing may yield larger delays and associated latency variation.

Table 3. Contribution to rx_clk Routing Varied by Speed Grade in LatticeECP2M Devices

Net	PCS2PLL	RX PLL Feedback	CLK2OUT	Q2CLK	Total
	rx_pcs_serdes_clk_ch0	rx_pcs_serdes_clk	CLK_TO_OUT	Rx_clk	
-5	2.263	-0.119	0.333	2.028	4.505
-6	2.06	-0.106	0.309	1.789	4.052
-m	1.147	-0.061	0.15	0.925	2.161
Var -5	1.71+/-0.558	-0.09-/+0.029	0.24+/-0.091	1.48+/-0.552	3.333+/-1.172
Var -6	1.60+/-0.456	-0.08-/+0.023	0.23+/-0.079	1.36+/-0.432	3.106+/-0.945

Table 4. Contribution to tx_clk Routing Varied by Speed Grade in LatticeECP2M Devices

Net	PCS2PLL ff_txser_fullclk (With Feedback)	TX PLL Feedback tx_pcs_serdes_clk	CLK2OUT CLK_TO_OUT	Q2CLK tx_clk	Total
-5	1.312	-0.119	0.333	2.028	3.554
-6	1.207	-0.106	0.309	1.789	3.199
-m	0.681	-0.061	0.15	0.925	1.695
Var -5	1.00+/-.315	-0.09-/+.029	0.24+/-.091	1.48+/-.552	2.624+/-0.929
Var -6	0.94+/-.263	-0.08-/+.023	0.23+/-.079	1.36+/-432	2.447+/-0.752

In the RX direction, there are two sources of clock skew:

- **Delayed rx_pcs_serdes_clk in the FPGA fabric**.
  The SERDES/PCS output data needs to meet the timing between the SERDES/PCS output clock rx_pcs_serdes_clk_ch0 and skewed rx_pcs_serdes_clk. The general routing from SERDES/PCS output pin to PLL input mainly determines the variation of skewed rx_pcs_serdes_clk. This component is shown in the PCS2PLL column in Table 3.
- **Clock skew between rx_pcs_serdes_clk and rx_clk**.
  rx_clk is generated from rx_pcs_serdes_clk and is lagging with respect to rx_pcs_serdes_clk. The variation of the clock skew is shown in Table 3 in the columns CLK2OUT and Q2CLK.

In the TX direction, there are also two sources of clock skew:

- **Delayed tx_pcs_serdes_clk in the FPGA fabric**.
  The tx_pcs_serdes_clk should be connected to the primary clock tree directly from the SERDES/PCS output clock pin. The delay may vary among different devices. As long as the data path meets timing between the FPGA and the SERDES/PCS, then the SERDES/PCS block compensates the variation of this clock.
- **Clock skew between tx_clk and tx_pcs_serdes_clk**.
  tx_clk is generated from tx_pcs_serdes_clk and is lagging with respect to tx_pcs_serdes_clk. This clock skew will make the core signal transfer to the high-speed clock faster but still in the same clock cycle.

Soft PCS Word Aligner Delay Compensation

This optional capability is implemented by the cpri_rxpe block shown in Figure 5. This capability generates an adjusted rx_clk from the core clock derived from the value of the comma detection offset value such that the user always receives a fixed rx_clk_offset regardless of the value of the word aligner offset value.

The word aligner delay compensation contribution to latency is shown in Tables 5 and 6. The adjustment of rx_clk_offset is determined by the nature of the recovered clock and word alignment offset. If the word aligner offset is 0, less adjustment is provided; if the word aligner offset is 9, the most adjustment is provided. The minimal PLL dynamic phase adjust step for this configuration is 1 ns, 1/16th of the 62.5 MHz rx_clk period. An additional 1/4 of rx_clk fixed steps are added to the adjusted rx_clk_offset so that data transfer from rx_clk domain to rx_clk_offset domain can be achieved reliably by module cpri_rxpe.

This capability reduces latency variation due to word aligner offset from ~+/-4ns (1.2G line rate) or +/-2ns (2.4G line rate) without compensation to ~+/-400ps over all offset values. Note that this capability is optional, i.e. users may choose to compensate for word aligner latency variation by using the specific word aligner offset value in their latency calculations.

The calculation follows the equation:

Compensated Delay = (DPHASE + FIXED_LATENCY) x 1000/(RX_Clkx16) – Word_align_select x UI

Table 5. Word Aligner Delay Compensation Contribution to Latency

Block	Fixed Latency	
RX		
cpri_rxpe	Offset	Offset is controlled by PLL
da_gen_ckr	0	rx_sync_offset
Subtotal for RX	**Offset**	

Table 6. PLL Dynamic Phase Adjust Mapping

Word Aligner Offset	PLL Adjust Steps								
	1.228 Gbps 0.814ns			2.456 Gbps 0.407ns			3.072 Gbps 0.326ns[1]		
	DPHASE	Fixed Latency	Compensated Delay	DPHASE	Fixed Latency	Compensated Delay	DPHASE	Fixed Latency	Compensated Delay
0	0	4	4.069	0	4	4.069	0	4	4.069
1	1	4	4.272	0	4	3.662	1	4	4.761
2	2	4	4.476	1	4	4.272	1	4	4.435
3	2	4	3.662	1	4	3.866	1	4	4.110
4	3	4	3.866	2	4	4.476	2	4	4.801
5	4	4	4.069	2	4	4.069	2	4	4.476
6	5	4	4.272	2	4	3.662	2	4	4.150
7	6	4	4.476	3	4	4.272	3	4	4.842
8	6	4	3.662	3	4	3.866	3	4	4.517
9	7	4	3.866	4	4	4.476	3	4	4.191
	Delay Variation		**0.814**			**0.814**			**0.773**
			+/-0.407			**+/-0.407**			**+/-0.387**

1. Not available in the LatticeECP2M reference design.

Total CPRI IP Core Latency and Latency Variation in LatticeECP2M Devices

This section gives the information needed to calculate the CPRI and SERDES/PCS latency. The latency data shown in Table is calculated based on a 122.88 MHz reference clock and word_align_offset equal to 0. In Table , a tx_clk or rx_clk cycle is one period of a 61.44 MHz clock. A *_serdes_clk cycle is one period of the 245.78 MHz full rate SERDES clock. A "UI" refers to one line rate bit period.

Table 7. CPRI IP Core Latency and Latency Variation

Item	Block	Fixed Latency	Total Latency 122.88 MHz REF			
			-5 Speed Grade		-6 Speed Grade	
			Constant	Variation	Constant	Variation
TX (A->D)						
1	CPRI Core	3 tx_clk cycles	48.82		48.82	
2	Clock skew: tx_clk -> tx_pcs_serdes_clk		-1.72	+/-0.64	-1.59	+/-0.51
3	TXFE	2.5 tx_clk cycles + 1 tx_pcs_serdes_clk cycle	44.75		44.75	
4	ff_txser_fullclk delay		-0.91	+/-0.29	-0.86	+/-0.24
Subtotal for TX		**5.5 tx_clk+ 1 tx_pcs_serdes_clk + skews**	90.94		91.12	

Table 7. CPRI IP Core Latency and Latency Variation (Continued)

Item	Block	Fixed Latency	Total Latency 122.88 MHz REF			
			-5 Speed Grade		-6 Speed Grade	
			Constant	Variation	Constant	Variation
RX (I->M)						
5	Clock skew: rx_pcs_serdes_clk_ch0 -> rx_pcs_serdes_clk		1.62	+/-.53	1.52	+/-.43
6	RXFE	6 rx_clk cycles	97.64		97.64	
7	Clock skew: rx_pcs_serdes_clk -> rx_clk		1.72	+/-.64	1.59	+/-.51
8	CPRI Core	2 rx_clk cycles	32.55		32.55	
9	rxpe (word_align_offset n=0-9)	PLL offset steps	4.07[6]	(+/-.407 optional)	4.07[6]	(+/-.407 optional)
Subtotal for RX		**8 rx_clk cycles + skews**	137.60		137.37	
SERDES (Latency Based on SERDES Internal Clock)						
10	Tx SERDES		23.60	+/-.28	23.60	+/-.28
11	Rx SERDES		21.98		21.98	
12	word_align offset (n=0-9)	-nUI	-0[6]		-0[6]	
Subtotal for SERDES			45.58		45.58	
Total Delay for SERDES and Core			**274.1**	**+/-2.4 (2.9)**	**274.1**	**+/-2.0 (2.4)**

Notes:
Items 1, 3, 6 and 8 – Fixed delays based on the core tx_clk and rx_clk signals and PCS/SERDES full clock.

Item 2 – Item 2 refers to the clock skew between the low-speed tx_clk and the high-speed tx_pcs_serdes_clk.

Item 5 – Item 5 refers to clock skew between the received SERDES output clock rx_pcs_serdes_clk_ch0 and the PLL output clock rx_pcs_serdes_clk. The skew is approximately equal to the general routing delay measured from the SERDES output pin ff_f_clk_ch0 to the PLL input, plus the routing difference between the PLL CLKOS and PLL CLKOP (feedback).

Item 7 – Item 7 refers to the clock skew between the high-speed rx_pcs_serdes_clk and rx_clk. If the same type of clock dividing circuit is used for both RX and TX. If the floorplanning on both low-speed clock generation flip-flops is carefully managed, then the skew can be canceled out in the total delay calculation with Item 2. For further information about these skews, refer to the FPGA Clock Generation and Routing Delay section of this document.

Item 9 – Item 9 is the potential contribution of the optional automatic word aligner delay compensation capability. This capability is described in detail in the Soft PCS Word Aligner Delay Compensation section of this document. If this capability is not used, the fixed latency will be increased by an amount equal to the contents of the CPRI IP word_align_offset register multiplied by UI (.407ns at 2.4G line rate).

Items 10 and 11 – These are values for the SERDES latency based on the 122.88MHz reference clock and 2.456Gbps line rate.

Items 9 and 12 – This computation uses word_align_offset equal to 0 (n=0).

Cable Length Delay Calculation

The template logic in the reference design includes a round-trip R21 and core delay measurement circuit. The delay is measured from A->M, as shown in Figure 5. The delay measurement circuit runs at 368 MHz (6 times the 61.44 MHz core clock frequency), providing a resolution of 2.71ns. The value of the delay cycle count can be read in register 0x81F. To determine the total round-trip delay, multiply the register value times the 2.71ns resolution clock period. The cable delay can be determined by subtracting the CPRI core and SERDES/PCS delays from the total round-trip delay.

Multiple Channel Configuration Considerations

The CPRI IP core is configured for SERDES Quad based protocol applications when initially generated via IPexpress. It is possible to configure the SERDES/PCS on a per-channel basis to support multiple CPRI channels in a single SERDES quad. Considerations specific to multiple channel support include:

- All receive direction channels have independent timing and the specific SERDES output ff_rxfullclk_ch[0]-[3] should be used for each individual channel.
- All transmit direction channels in a SERDES quad use the same SERDES output clock ff_txser_fullclk.
- If all channels are operating at the same data rate, 1.2G or 2.4G, the SERDES should be configured in full rate mode and ff_txser_fullclk can be selected from any working channel (it is recommended to use the lowest indexed channel). The ff_txser_fullclk frequency will be 120MHz or 240MHz for line rates of 1.G or 2.4G, respectively.
- To support a mix of channels running at 1.2G and 2.4G, the SERDES channels operating at 2.4G should be configured for full rate mode and the channels operating at 1.2G should be configured for half rate mode. The ff_txser_fullclk should be selected from one of the 1.2G channels. The ff_txser_fullclk frequency will be 120MHz.

IP Core Generation

The CPRI low latency configuration is supported in the CPRI IP core available for download from the Lattice website (www.latticesemi.com/products/intellectualproperty). Complete details on downloading and generating the CPRI IP core are given in the CPRI IP Core User's Guide.

As mentioned previously, all of the files supporting the SERDES/PCS configuration and RXFE and TXFE blocks for the low latency configuration are provided in the CPRI IP evaluation directory, cpri_lowlatency_eval. The structure of cpri_lowlatency_eval is equivalent to the structure of the cpri_eval directory that supports the basic configuration with the following exceptions:

- During the IP evaluation generation, the following additional files are automatically generated to support the soft PCS capability implemented in the FPGA in LatticeECP2M devices:

  softpcs_top.ngo
  softpcs_beh.v
  softpcs_top_bb.v

- The testbench and source code supporting the low latency design are different than the ones supporting the basic design.

The procedures for instantiating, simulating and implementing the core are equivalent to those for the basic configuration of the core, which are described in the CPRI IP Core User's Guide. Note that the soft PCS capability must be instantiated in the user's top-level along with the CPRI IP logic core.

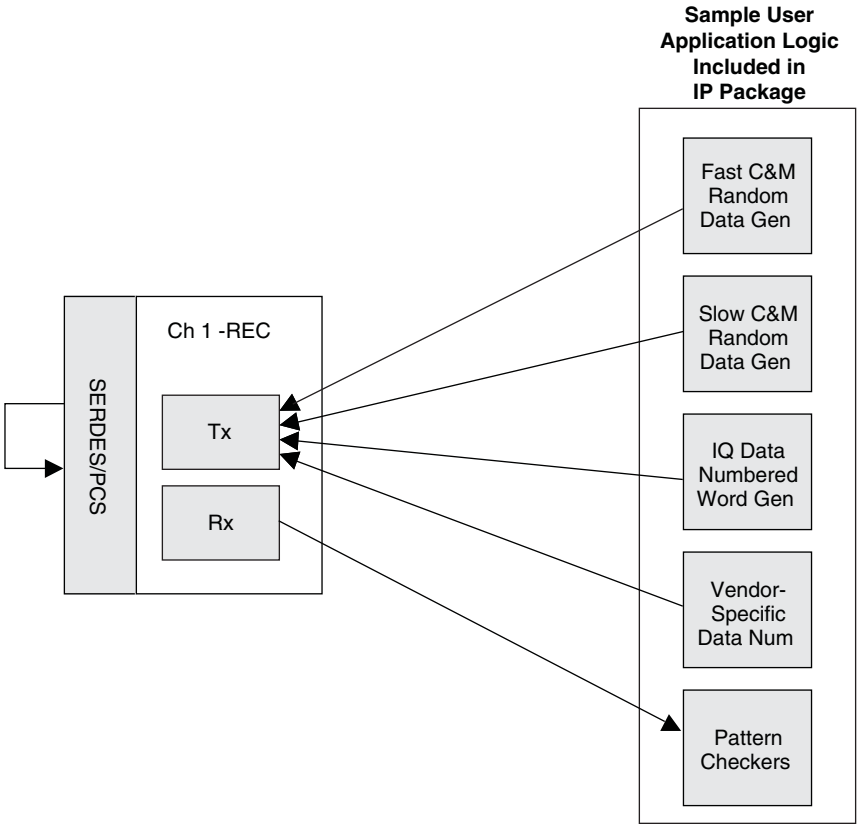
Low Latency Configuration Evaluation Capabilities

The CPRI IP core evaluation package includes evaluation simulation and implementation capabilities for the low latency configuration equivalent to those provided for the basic configuration of the core, which are described in the CPRI IP Core User's Guide.

Included in the evaluation package is a reference top-level file, cpri_reference_top.v, which 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 and a register module supporting programmable control of the system processor interface and SCI. Verilog source RTL for these modules are provided in \<project_dir>\cpri_lowlatency_eval\<username>\src\rtl\template\ecp2m. The top-level configuration is specified via the parameters defined in the cpri_defines.v file in \<project_dir>\cpri_lowlatency_eval\<username>\src\params.

The template logic supporting the REC configuration shown previously in Figure 1 is delivered with the CPRI IP core. Along with this sample top level a sample user application is included. It is expected that the end user will replace this sample application with their specific application design. The user application logic delivered in the template includes circuitry that can be used to test the CPRI IP Core. This circuitry includes 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 evaluation package 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 6.

Figure 6. Pattern Generators and Checkers Included with Delivered IP Core Testbench (One Direction Shown)



Reference Design Register Descriptions

There are no user-accessible registers within the CPRI core. All control and status appear as internal I/O at the core boundary. Registers are provided in the evaluation capability 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 8.

Note that these registers are equivalent to the registers provided for the basic CPRI configuration, which are given in the CPRI IP Core User's Guide, with a few additions. The additional registers for the low latency configuration are indicated in Table 8.

Table 8. CPRI Reference Design Register Map

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Control 0	0x800	1:0	LBR[1:0] - Line Bit Rate Control For maximum CPRI line rate enable by user	00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 2457.6 MHz
		2	FORCE_SM_STANDBY - Force startup state machine to standby	Zero to one transition
		4:3	PCS_SERDES_RATE[1:0] - Line rate supported by SERDES/PCS	00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 2457.6 MHz
CPRI Control 1	0x801	0	TX_L1_RAI	Remote Action Indication
		1	TX_L1_SDI	SAP Defect Indication
CPRI Control 2	0x802	0	HDLC_240_EN	Enable = 1, Disable = 0
		1	HDLC_480_EN	Enable = 1, Disable = 0
		2	HDLC_960_EN	Enable = 1, Disable = 0
		3	HDLC_1920_EN	Enable = 1, Disable = 0
		4	TX_HYP_RST_EN	Enable = 1, Disable = 0
		5	TX_BFN_RST_EN	Enable = 1, Disable = 0
CPRI Control 3	0x803	5:0	[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	0	TX_ETH_EMPTY	Ethernet FIFO error bits
		1	TX_ETH_FULL	Ethernet FIFO error bits
		2	RX_ETH_EMPTY	Ethernet FIFO error bits
		3	RX_ETH_FULL	Ethernet FIFO error bits
		4	PRO_INTRUPT	Processor interrupt
		5	VER_NUM_ERR	Version number error bit
		6	RX_LOS	Rx loss of signal
		7	RX_LOF	Rx loss of frame
CPRI Status Reg 0	0x806	1:0	RATE_MODE	Final negotiated line bit rate
		7:4	lsm_state[1]	Soft PCS RX external link state machine state
CPRI Status Reg 1	0x807	4:0	[0] - RX_L1_RST_RQSTACK [1] - RX_L1_RAI [2] - RX_L1_SDI [3] - RX_L1_LOS [4] - RX_L1_LOF	Received L1 inband protocol bits of byte Z.130.0
		5	VER_NUM_ERR	Version number error bit
		6	RX_LOS	Rx loss of signal
		7	RX_LOF	Rx loss of frame
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

Table 8. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Status Reg 3	0x80a	2:0	TX_HDLC_MODE	Final negotiated HDLC bit rate.
		6:4	RX_L1_HDLC_MODE	Received L1 inband protocol bits of byte Z.66.0
CPRI Status Reg 4	0x80b	2:0	CPRI_STUP_STATE	CPRI start up state
		3	rx_sync_status[1]	Soft PCS RX sync status from the external link statue machine
		7:4	word_align_select[1]	Soft PCS Word Aligner Selection Value (0-9)
Test Control	0x80c	0	REC_MD	REC Mode - 1=REC, 0=RE
		1	TEST_MD	Test mode - 1=short timer
		5	RXRST	RX Ethernet FIFO reset (debug only: active high)
		6	TXRST	TX Ethernet FIFO reset (debug only: active high)
		7	CORE_RESET	Used if core reset port not connected to GSR.
Test Loop Data Error Reg	0x80d	0	RX_ETH_PDO_DA_ERR	Test bench check loop back Ethernet data errors
		1	RX_IQ_DA_ERR	Test bench check loop back IQ data errors
		2	VENDOR_PDO_DA_ERR	Test bench check loop back vendor data errors
		3	RX_HDLC_PDO_DA_ERR	Test bench check loop back HDLC data errors
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.

Table 8. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
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.
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.

Table 8. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
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.
SOFTPCS Error Reg	0x81E	0	~rx_sync_status[1]	External link state machine not sync to the comma
		1	rx_pcs_cv	PCS code violation
		2	rx_pcs_disp_err	PCS disparity error

Table 8. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
R21 CPRI Delay Status Reg	0x81F	7:0	R21_cpri_coarse_timer[1]	R21 and CPRI coarse timer delay based on clock running 6 times core clock frequency.

1. Only supported for low latency configuration.

Using the CPRI IP Core with LatticeECP3 FPGAs

Introduction

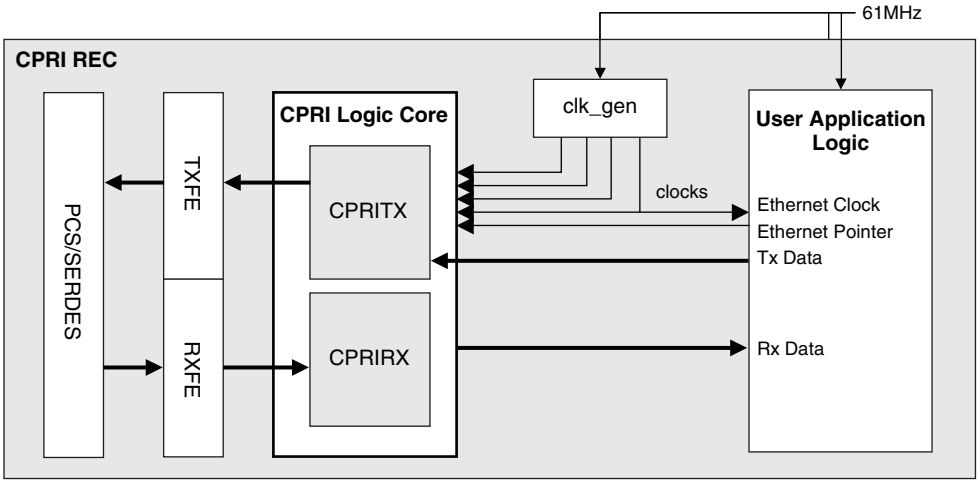
Lattice's CPRI IP core supports a low latency variation configuration that includes special design features enabling the creation of designs meeting stringent CPRI delay variation requirements. The "low latency" core configuration is equivalent to the basic configuration, but includes a modified SERDES/PCS interface that supports the ability to manage the variability in the absolute latency for data transmission through the core.

This document focuses on the detailed specifications associated with implementing and using the low latency CPRI configuration.

General Description

Figure 7 shows a system-level block diagram of the CPRI IP core. This block diagram shows the core configured to support REC applications. The CPRI core may also be configured to support RE applications. See the CPRI IP Core User's Guide for details. The complete CPRI IP core includes two key components, the CPRI IP logic core and separate logic blocks (RXFE and TXFE in Figure 7) that support the interface between the logic core and the SERDES/PCS functions integrated in the FPGA.

Figure 7. CPRI IP Core System-Level Block Diagram



The CPRI IP logic core is provided in NGO format and is identical for both the basic and low latency core configurations. However, the RXFE and TXFE logic blocks and the SERDES/PCS configuration differ significantly for the low latency configuration vs. the basic configuration.

For the basic CPRI IP core configuration, the SERDES/PCS block is configured to support basic, complete 8b10b functionality in both the transmit and receive directions. The SERDES/PCS configuration is controlled using a configuration txt file generated via IPexpress that specifies the settings of the SERDES block configuration memory cells that will be loaded via the FPGA program bitstream. No modifications to the SERDES configuration are required after the bitstream is loaded and the FPGA is functional. The RXFE and TXFE logic blocks for the basic configuration implement simple rate-adaptation functions between the CPRI logic and the SERDES/PCS block.

These blocks are provided in RTL format in the basic configuration evaluation package included with the CPRI IP core.

For the low latency CPRI configuration, the SERDES/PCS block configuration is modified to bypass specific internal functions that insert latency variation in the data path. Where required, these bypassed functions are implemented in soft (FPGA) logic where the latency variation can be tightly controlled or eliminated. The RXFE and TXFE blocks include additional specific capabilities to manage latency variation through the soft logic. Specific SERDES/PCS configuration and RXFE and TXFE block features supporting low latency variation include:

- The SERDES link high-speed clocks and core clocks are managed carefully to minimize latency variation due to clock routing delays and the associated variation across device PVT.
- The FPGA bridge FIFOs in the SERDES/PCS block are bypassed in both transmit and receive directions to eliminate the associated latency variation.
- The receive direction PCS word alignment function is implemented in LatticeECP3 hard PCS logic, eliminating the unknown component of the latency variation associated with the SERDES deserializer 10-bit word alignment function by reading word align offset status registers. The value can be read at channel register 0x22 with the base address.
- Automatic word aligner delay compensation is optionally supported, in which the phase of the receiver clock is adjusted based on the value of comma detection offset value so that the user always get a phase adjusted rx_clk_offset providing constant latency, aligned receive data.

The I/O signals for the CPRI IP logic core are described in detail in the CPRI IP Core User's Guide.

Examples of RTL source supporting all of these capabilities on the LatticeECP3-95-1156 device are provided in the low latency evaluation package included with the CPRI IP core. This evaluation package supports 614M, 1.2G, 2.4G, and 3.072G line rate implementations.

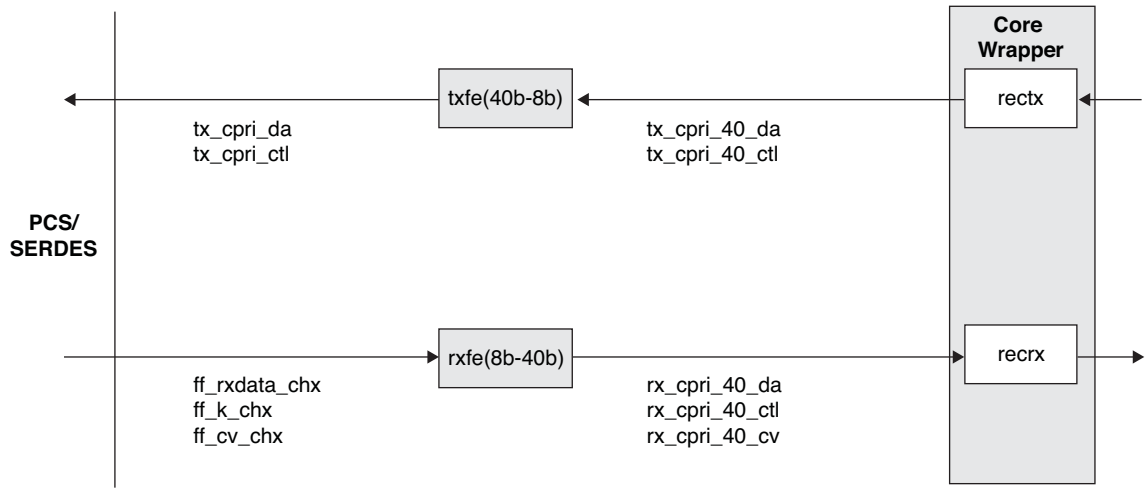
Functional Description

Figure 8 shows a block diagram of the functionality implemented in the RXFE and TXFE blocks for the low latency CPRI configuration. As mentioned previously, the TXFE block is provided in RTL format while the RXFE block includes a mix of RTL and NGO formats. When generating the CPRI IP core, all of the files supporting the RXFE and TXFE blocks are provided in the CPRI IP evaluation directory cpri_lowlatency_eval.

The TXFE block essentially performs rate adaptation between the 40-bit data interface of the transmit block in the CPRI logic core and the 8-bit data interface of the SERDES/PCS block. Careful management of the clocking for this block is key to achieving low latency variation.

The following sections provide details on how to configure the CPRI IP to support low latency variation.

Figure 8. RXFE and TXFE Logic Blocks



SERDES Configuration

To implement the low latency design, the SERDES/PCS is configured with a 8b10b interface and the basic functionality shown in Table .

Table 9. CPRI SERDES Configuration

Bit Rate (bps)		614M	1.2G	2.4G	3G
Protocol		CPRI			
Reference Clock Multiplier		10X Half	10x	20x	20x or 25x
refclk		122.88	122.88	122.88	153.76 for 20x 122.88 for 25x
8-Bit Interface	ff_rxfullclk_ch[0:3]	614.44	122.88	245.76	307.2
	ff_txfullclk_ch0	614.44	122.88	245.76	307.2
RX PCS	Word Aligner	No Bypass	No Bypass	No Bypass	No Bypass
	8b10b Decoder	No Bypass	No Bypass	No Bypass	No Bypass
	FPGA Bridge FIFO	Bypass	Bypass	Bypass	Bypass
	CTC	Bypass	Bypass	Bypass	Bypass
	SERDES Bridge	Bypass	Bypass	Bypass	Bypass
TX PCS	FPGA Bridge FIFO	Bypass	Bypass	Bypass	Bypass
	CTC	Bypass	Bypass	Bypass	Bypass
	SERDES Bridge	Bypass	Bypass	Bypass	Bypass

The SERDES/PCS configuration is initially specified using a configuration txt file generated via IPexpress. This file specifies the settings of the SERDES block configuration memory cells that will be loaded via the FPGA program bitstream. After the bitstream has been loaded and the FPGA is functional, the SERDES configuration must be modified via the LatticeECP3 SERDES Client Interface (SCI). Complete details on managing the SERDES configuration via the SCI are given in TN1176, LatticeECP3 SERDES/PCS Usage Guide.

What's New in LatticeECP3 Low Latency Design

Table 10 highlights the differences between low latency design in LatticeEC2M (as described above) and LatticeECP3, in terms of PCS/SERDES configuration.

Table 10. LatticeECP2M and Lattice ECP3 Low Latency Design Comparison

		LatticeECP2M	LatticeECP3
TXFE	Data width conversion	32-bit to 8-bit	40-bit to 8-bit
RXFE	Data width conversion	32-bit to 8-bit	40-bit to 8-bit
	Soft PCS	Yes	No
	CPRI core	32-bit	40-bit
PCS RX	Word aligner	Bypass	No bypass
	8b10b decoder	Bypass	No bypass
	PCS Tx		
	Tx full clock output	txser_fullclk	ff_txfull_clk_ch[x]

Clock Configuration for High Link Delay Accuracy

Figure 9 shows the clock configuration for a core configured as a master. In this mode, the SERDES/PCS reference clock is generated by an oscillator and drives the SERDES/PCS. The SERDES/PCS generates recovered RX clock and TX clock signals. All other required data clocks are derived from the SERDES/PCS RX and TX output clocks for the RX and TX directions, respectively. The figure shows the configuration without the 614Mbps line rate. If 614Mbps is supported, the clock mux will be inserted into the clock route before the rx_clk/tx_clk CIB.

Figure 10 shows the clock configuration for a slave core. In this case, the SERDES/PCS reference clock (TX) is generated from the recovered clock that has been cleaned up using an external jitter-removal PLL. In both cases, the SERDES/PCS reference clock is 122.88MHz.

Some considerations:

- The RX high-speed recovered clock must be connected to the dedicated device clock tree via general routing. In the eval package, this clock is connected to the nearest secondary clock tree CIB near to the SERDES/PCS. This configuration minimizes the routing delay and associated latency variation.
- The rx_clk and tx_clk signals are both generated from flip-flops and access the primary clock trees via CIBs. These flip-flops should be manually located to minimize overall routing delay and associated latency variations. For multi-channel applications, this may not be able to implemented for every channel.
- Since the use of general routing impacts the clock routing variation, clock placement and routing results should be carefully checked to make sure the shortest clock routing paths are implemented. Differences in the specific design, device targeted and/or software version may yield different routing results. It is recommended that the design be checked in EPIC to verify the desired routing configuration has been achieved.
- The TXFE and the RXFE modules are involved in the data transfer between the SERDES/PCS 8-bit high-speed interface and the FPGA fabric. This transfer occurs at the full clock rate up to 307.2MHz. These modules should be UGROUPed and placed near the SERDES/PCS interface. The evaluation package provides an example for a LatticeECP3-150EA-1156 device.

Figure 9. Clock Configuration for CPRI Master Applications

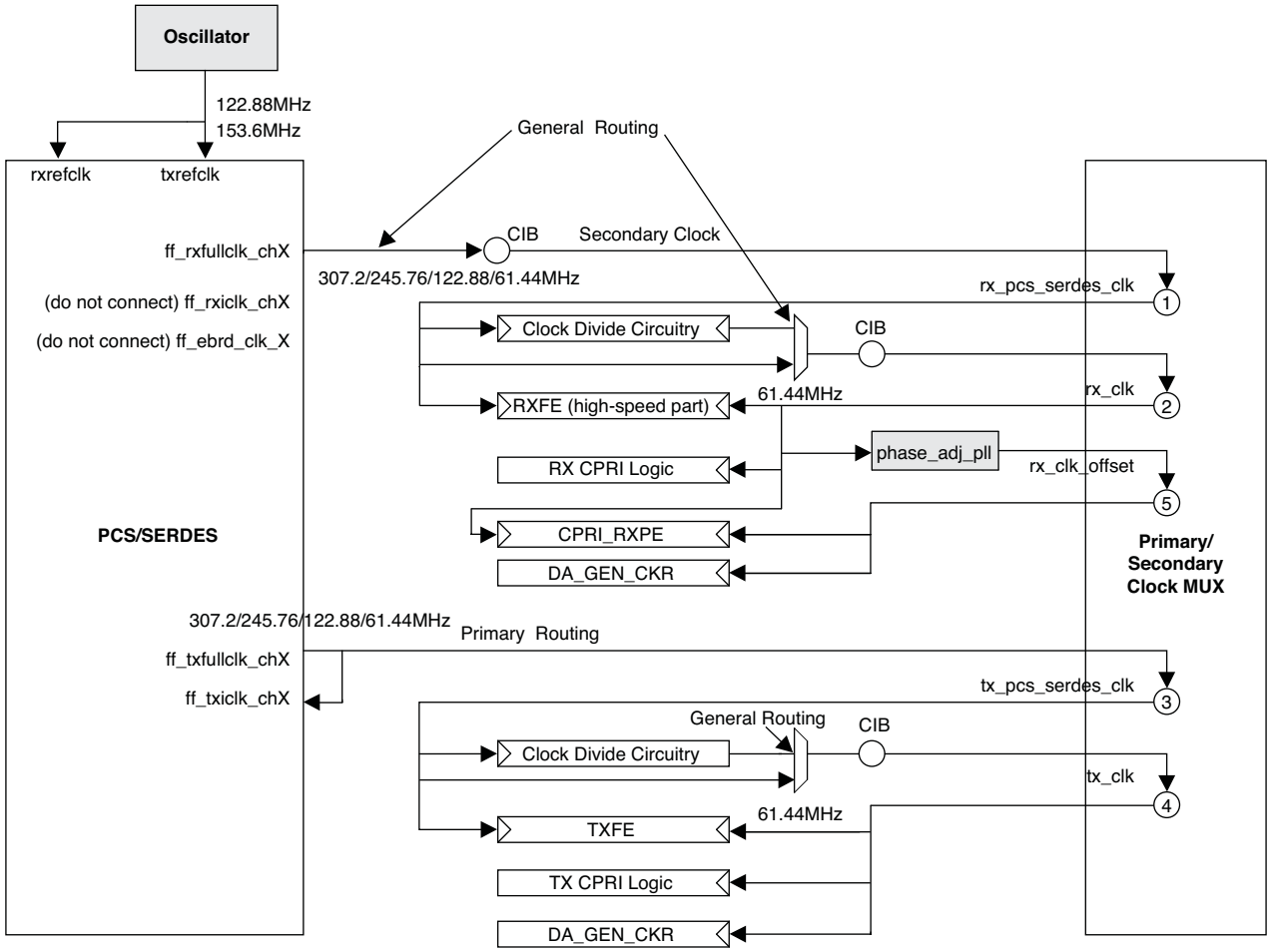
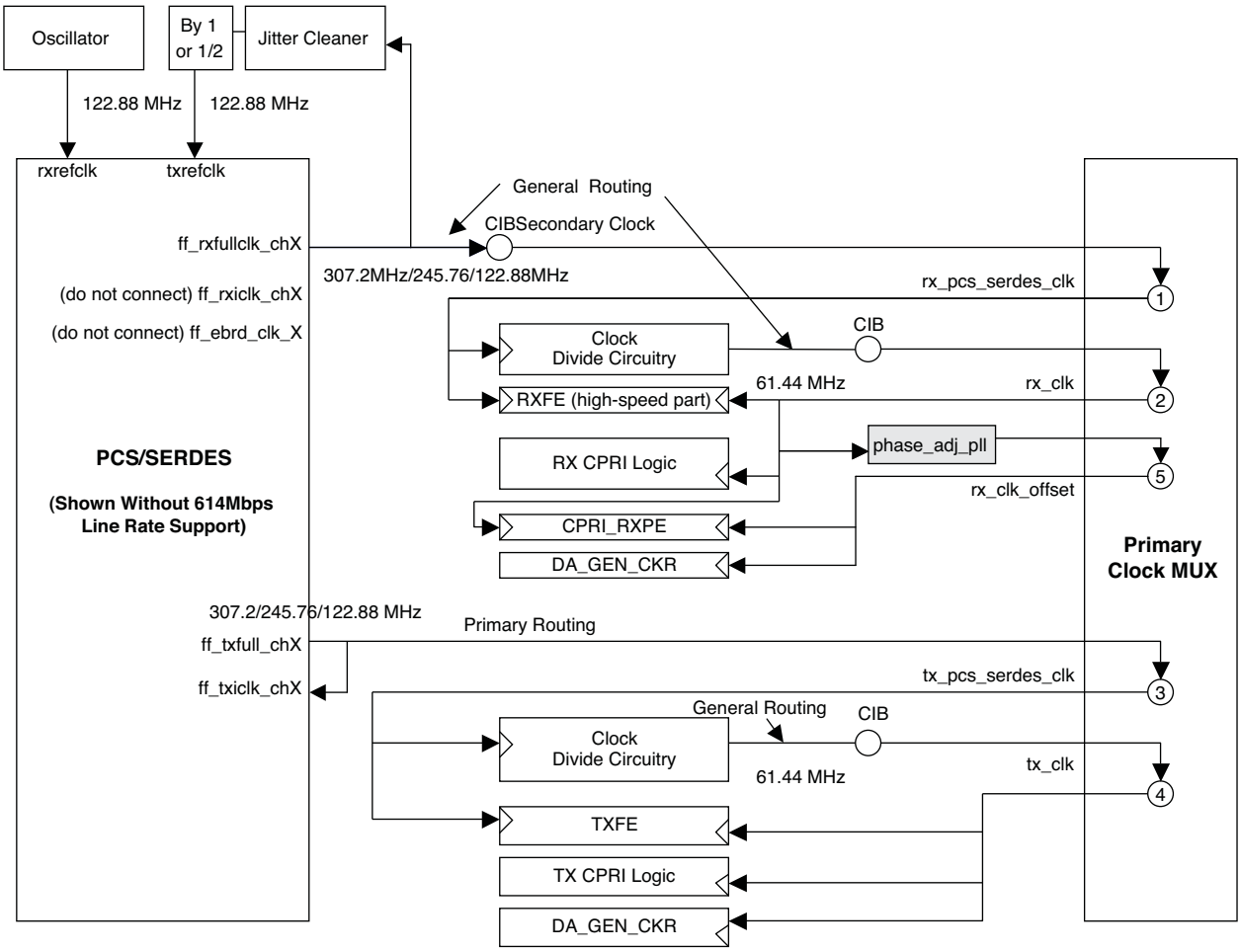


Figure 10. Clock Configuration for CPRI Slave Applications

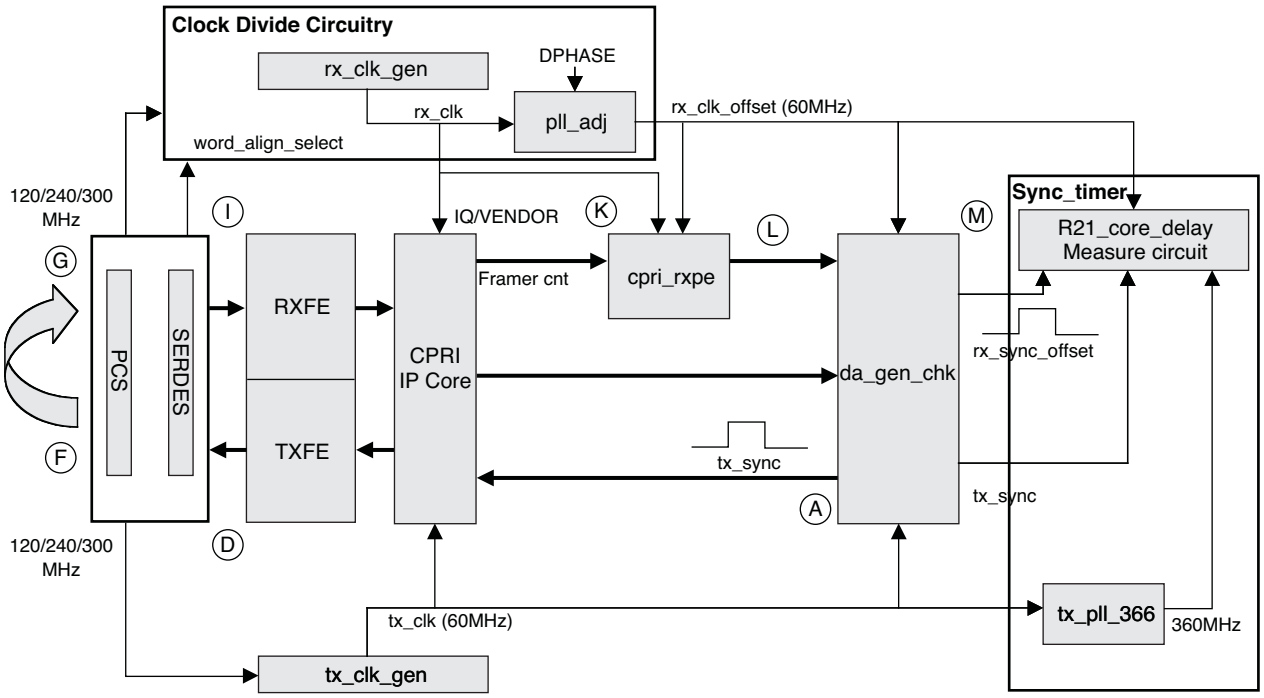


Core Delay Measurement

The delay through the CPRI IP core and associated SERDES/PCS can be modeled as shown in Figure 11. Delay though the FPGA has both a fixed logic component and a variable delay component. The variable delay is caused by clock routing.

The total delay can be divided into four parts: SERDES/PCS block delay, FPGA clock generation and routing delay, CPRI IP core delay, and soft PCS word aligner delay compensation. The total delay is defined as the time between the tx_sync and rx_sync_offset signals.

Figure 11. CPRI IP Delay Model



FPGA Clock Generation and Routing Delay

To calculate the total transceiver delay, in addition to the fixed delay through the SERDES/PCS and IP core, the clock routing variation for different devices and speed grades must also be considered. Tables 11 and 11 show results for the latency variation for the RX and TX user side clock domains achieved for a LatticeECP3-150EA-1156 device. These results are based on ispLEVER TRACE tool analysis of the corresponding clock domains for worst case fast (setup) and slow (hold) performance. The total rx_clk delay (for -7 speed grade device) is 3.70+/-1.76 ns between the fast and slow devices as shown in Table 11. If 614Mbps is not required, the clock mux is not needed, thus the variation can be reduced to +/-1.5ns. Similarly, the total tx_clk delay is 3.10+/-1.60 ns as shown in Table 11. Without a 614Mbps line rate, the clock mux can be removed, thus the variation will be reduced to +/-1.35ns.

Note that the clock delay value will vary depending on routing results and different designs with different clock routing may yield larger delays and associated latency variation.

Table 11. Contribution to rx_clk Routing Varied by Speed Grade

	PCS2FF	CLK2OUT	Q2Mux/MUX	MUX2CLK	
Net	rx_pcs_serdes_ch 0	CLK_TO_OUT	clk mux	Rx_clk	Total
6	2.817	0.303	0.847	2.119	6.086
7	2.555	0.273	0.774	1.854	5.456
-m	1.004	0.08	0.269	0.59	1.943
Median 6	1.9105	0.1915	0.558	1.3545	4.0145
Var -6	0.9065	0.1115	0.289	0.7645	2.0715
Median 7	1.7795	0.1765	0.5215	1.222	3.6995
Var -7	0.7755	0.0965	0.2525	0.632	1.7565

Table 12. Contribution to tx_clk Routing Varied by Speed Grade (ns)

Net	PCS2FF	CLK2OUT	Q2Mux/MUX	MUX2CLK	Total
	tx_pcs_serdes_ch0	CLK_TO_OUT	clk mux	tx_clk	
6	1.647	0.303	0.853	2.507	5.31
7	1.424	0.273	0.78	2.22	4.697
-m	0.436	0.08	0.273	0.707	1.496
Median 6	1.0415	0.1915	0.563	1.607	3.403
Var -6	0.6055	0.1115	0.29	0.9	1.907
Median 7	0.93	0.1765	0.5265	1.4635	3.0965
Var -7	0.494	0.0965	0.2535	0.7565	1.6005

In the RX direction, there are two sources of clock skew:

- Delayed rx_pcs_serdes_clk in the FPGA fabric.

  The SERDES/PCS output data needs to meet the timing with the routing delay between SERDES/PCS output clock rx_pcs_serdes_clk_ch0 and FF loads. The general routing and secondary clock routing mainly determines the variation of the skew of rx_pcs_serdes_clk_ch0.

- Clock skew between rx_pcs_serdes_clk and rx_clk.

  rx_clk is generated from rx_pcs_serdes_clk_ch0 and is lagging with respect to rx_pcs_serdes_clk_ch0. The variation of the clock skew is shown in Table 11 in the columns CLK2OUT and Q2CLK.

In the TX direction, there are also two sources of clock skew:

- Delayed tx_pcs_serdes_clk in the FPGA fabric.

  The tx_pcs_serdes_clk should be connected to the primary clock tree directly from the SERDES/PCS output clock pin. The delay may vary among different devices. As long as the data path meets timing between the FPGA and the SERDES/PCS, then the SERDES/PCS block compensates the variation of this clock.

- Clock skew between tx_clk and tx_pcs_serdes_clk.

  tx_clk is generated from tx_pcs_serdes_clk and is lagging with respect to tx_pcs_serdes_clk. This clock skew will make the core signal transfer to the high-speed clock faster but still in the same clock cycle.

PCS Word Aligner Delay Compensation

This optional capability is implemented by the cpri_rxpe block shown in Figure 11. This capability generates an adjusted rx_clk from the core clock derived from the value of the comma detection offset value such that the user always receives a fixed rx_clk_offset regardless of the value of the word aligner offset value.

The word aligner delay compensation contribution to latency is shown in Tables 13 and 14. The adjustment of rx_clk_offset is determined by the nature of the recovered clock and word alignment offset. If the word aligner offset is 0, less adjustment is provided; if the word aligner offset is 9, the most adjustment is provided. The minimal PLL dynamic phase adjust step for this configuration is 1 ns, 1/16th of the 62.5 MHz rx_clk period. An additional ¼ of rx_clk fixed steps are added to the adjusted rx_clk_offset so that data transfer from rx_clk domain to rx_clk_offset domain can be achieved reliably by module cpri_rxpe.

This capability reduces latency variation due to word aligner offset from ~+/-4ns (1.2G line rate), +/-2ns (2.4G line rate), or ~+/-1.7ns (3G line rate) without compensation to ~+/-400ps over all offset values. Note that this capability is optional, i.e. users may choose to compensate for word aligner latency variation by using the specific word aligner offset value in their latency calculations.

The calculation follows the equation:

Compensated Delay = (DPHASE + FIXED_LATENCY) x 1000/(RX_Clkx16) – Word_align_select x UI

Table 13. Word Aligner Delay Compensation Contribution to Latency

Block	Fixed Latency	
RX		
cpri_rxpe	Offset	Offset is controlled by PLL
da_gen_ckr	0	rx_sync_offset
Subtotal for RX	**Offset**	

Table 14. PLL Dynamic Phase Adjust Mapping

Word Aligner Offset	PLL Adjust Steps								
	1.228 Gbps 0.814ns			2.456 Gbps 0.407ns			3.072 Gbps 0.326ns		
	DPHASE	Fixed Latency	Compensated Delay	DPHASE	Fixed Latency	Compensated Delay	DPHASE	Fixed Latency	Compensated Delay
0	0	4	4.069	0	4	4.069	0	4	4.069
1	1	4	4.272	0	4	3.662	1	4	4.761
2	2	4	4.476	1	4	4.272	1	4	4.435
3	2	4	3.662	1	4	3.866	1	4	4.110
4	3	4	3.866	2	4	4.476	2	4	4.801
5	4	4	4.069	2	4	4.069	2	4	4.476
6	5	4	4.272	2	4	3.662	2	4	4.150
7	6	4	4.476	3	4	4.272	3	4	4.842
8	6	4	3.662	3	4	3.866	3	4	4.517
9	7	4	3.866	4	4	4.476	3	4	4.191
	Delay Variation		**0.814**			**0.814**			**0.773**
			+/-0.407			**+/-0.407**			**+/-0.387**

Total CPRI IP Core Latency and Latency Variation in LatticeECP3 Devices

This section gives the information needed to calculate the CPRI and SERDES/PCS latency for 3G configuration, the 2.4G and 1.2G data rate have different number of clock cycles in TXFE and RXFE module in the reference design. The latency data shown in Table 15 is calculated based on a 3.072G line rate and word_align_offset equal to 6. In Table 15, a tx_clk or rx_clk cycle is one period of a 61.44 MHz clock. A *_serdes_clk cycle is one period of the 307.2 MHz full rate SERDES clock. A "UI" refers to one line rate bit period. This is a result of including 614Mbps line rate support. If this is not needed, the variation can be reduced by +/-0.5ns from the +/-3.46ns shown below for a -7 device.

Table 15. CPRI IP Core Latency and Latency Variation in LatticeECP3-150EA-1156

Item	Block	Fixed Latency	Total Latency 153.6 MHz REF			
			-6 Speed Grade		-7 Speed Grade	
			Constant	Variation	Constant	Variation
Tx (A->D)						
1	CPRI core	3 tx_clk cycles	48.82		48.82	
2	Clock skew: tx_clk -> tx_pcs_serdes_clk		-2.36	+/-1.30	-2.17	+/-1.11
3	TXFE	2 tx_clk cycles + 9 tx_pcs_serdes_clk cycles	61.85		61.85	
4	ff_txfullclk delay		-1.04	+/-0.61	-0.93	+/-0.49
Subtotal for Tx		**6 tx_clk+ 9 tx_pcs_serdes_clk + skews**	107.27		107.58	
Rx (I->M)						
5	Clock delay: rx_pcs_serdes_clk_ch0		1.91	+/-0.91	1.78	+/-0.78
6	RXFE	7 rx_pcs_serdes_clk + 1 rx_clk cycles	39.06		39.06	
7	Clock skew: rx_pcs_serdes_clk -> rx_clk		2.10	+/-1.16	1.92	+/-0.98
8	CPRI core	2 rx_clk cycles	32.55		32.55	
9	rxpe (word_align_offset n=0-9)	PLL offset steps	4.07	(+/-0.387 optional)	4.07	(+/-0.387 optional)
Subtotal for Rx		**8 rx_clk cycles + skews**	79.78		79.40	
SERDES (Latency Based on SERDES Internal Clock)						
10	Tx SERDES	4 tx_clki + 18UI-133ps	18.75	+/-0.10	18.75	+/-0.10
11	Rx SERDES	8 rx_clki + 12UI + 831ps	30.78		30.78	
12	word_align offset (n=0-9)	-nUI	0		0	
Subtotal for SERDES			49.53		49.53	
Total Delay for SERDES and Core			236.5	+/-4.08 (4.47)	236.5	+/-3.46 (3.85)

Notes:
Items 1 and 8 – Fixed delay based on the core tx_clk and rx_clk signals.

Items 2 and 7 – Item 2 refers to the clock skew between the low-speed tx_clk and the high-speed tx_pcs_serdes_clk. Item 7 refers to the clock skew between the high-speed rx_pcs_serdes_clk and rx_clk. If the same type of clock dividing circuit is used for both Rx and Tx, and if the floor planning on both low-speed clock generation flip-flops is carefully managed, then the skew can be canceled out in the total delay calcula¬tion. For further information about these skews, please refer to the FPGA Clock Generation and Routing Delay section of this document.

Item 4 – Clock delay between the received SERDES output clock rx_pcs_serdes_clk_ch0 and input of the flip-flop that clocks the rx_clk.

Item 9 – Potential contribution of the optional automatic word aligner delay compensation capability. This capability is described in detail in the Soft PCS Word Aligner Delay Compensation section of this document. If this capability is not used, the fixed latency will be increased by an amount equal to the contents of the CPRI IP word_align_offset register multiplied by UI (0.387ns at 3G line rate).

Items 9 and 12 – This computation uses word_align_offset equal to 0 (n=0).

Cable Length Delay Calculation

The template logic in the reference design includes a round-trip R21 and core delay measurement circuit. The delay is measured from A->M as shown in Figure 11. The delay measurement circuit runs at 368 MHz (6 times the

61.44 MHz core clock frequency), providing a resolution of 2.71ns. The value of the delay cycle count can be read in register 0x81F. To determine the total round-trip delay, multiply the register value times the 2.71ns resolution clock period. The cable delay can be determined by subtracting the CPRI core and SERDES/PCS delays from the total roundtrip delay.

Multiple Channel Configuration Considerations
The CPRI IP core is configured for channel based CPRI protocol application when initially generated via IPexpress. It is possible to configure the SERDES/PCS on a per-channel basis to support multiple CPRI channels in a single SERDES quad. Considerations specific to multiple channel support include:

- All receive direction channels have independent timing and the specific SERDES output ff_rxfullclk_ch[0]-[3] should be used for each individual channel.
- In the transmit direction, it is not recommended to use a ff_txfullclk_ch[i] to drive ff_txi_clk_ch[i] for different channels in the same quad unless all the channels in this quad are configured with the same rates.
- If the channels in a quad are not configured with the same rates (for example, full and half rate) each channel should route its own ff_txfullclk to drive the ff_txi_clk on the same channel through the primary clock tree or a secondary regional clock.
- If a channel is configured with a 3.072Gbps line rate, the ff_txfullclk from this channel should be routed through the primary clock tree. When this channel is running at 3.072Gbps, other channels in this quad can only run at a 3.072Gbps line rate.
- It is OK to configure a single channel to be 3.072G, 2.456G, 1.2288G, or 614M line rates. For multi-channel using the same PCS/SERDES quad, only the x2 line rates configuration can be supported.

IP Core Generation

The CPRI low latency configuration is supported in the CPRI IP core available for download from the Lattice web site. Complete details on downloading and generating the CPRI IP core are given in the CPRI IP Core User's Guide.

As mentioned previously, all of the files supporting the SERDES/PCS configuration and RXFE and TXFE blocks for the low latency configuration are provided in the CPRI IP evaluation directory, cpri_lowlatency_eval. The structure of cpri_lowlatency_eval is equivalent to the structure of the cpri_eval directory that supports the basic configuration with the following exceptions:

The testbench and source code supporting the low latency design are different than the ones supporting the basic design.

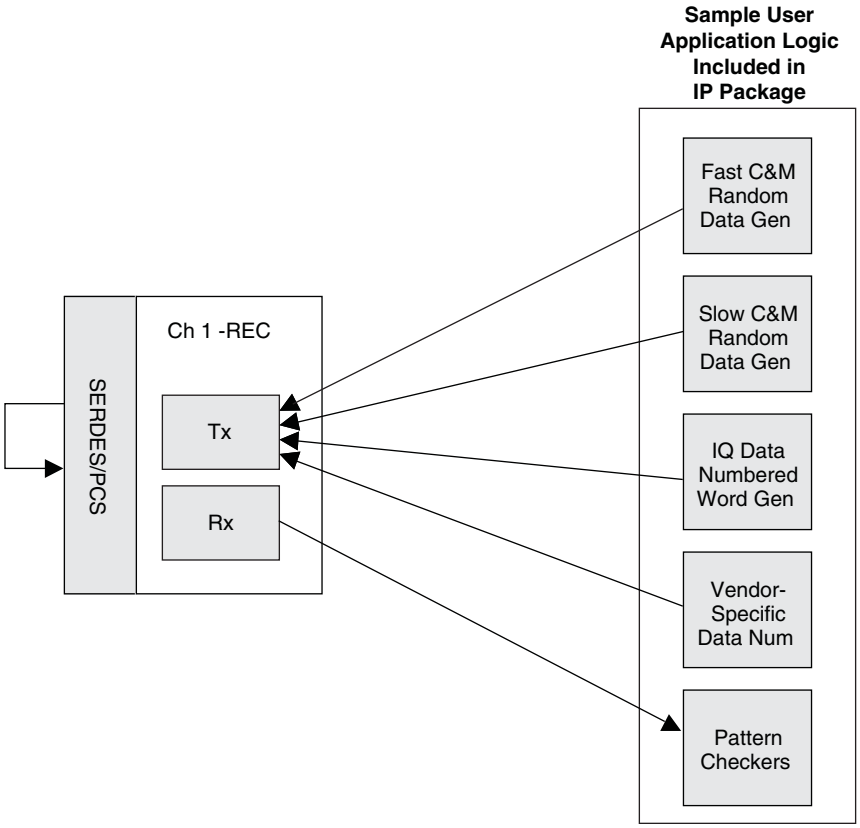
The procedures for instantiating, simulating and implementing the core are equivalent to those for the basic configuration of the core, which are described in the CPRI IP core user's guide. Note that the soft PCS capability must be instantiated in the user's top-level along with the CPRI IP logic core.

Low Latency Configuration Evaluation Capabilities
The CPRI IP core evaluation package includes evaluation simulation and implementation capabilities for the low latency configuration equivalent to those provided for the basic configuration of the core, which are described in the CPRI IP core user's guide.

Included in the evaluation package is a reference top-level file, cpri_reference_top.v, which customers 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 and a register module supporting programmable control of the system processor interface and SCI. Verilog source RTL for these modules are provided in \<project_dir>\cpri_lowlatency_eval\<username>\src\rtl\template\ecp3. The top-level configuration is specified via the parameters defined in the cpri_defines.v file in \<project_dir>\cpri_lowlatency_eval\<username>\src\params.

The template logic supporting the REC configuration shown in Figure 7 is delivered with the CPRI IP core. Along with this sample top level a sample user application is included. It is expected that the end user will replace this sample application with their specific application design. The user application logic delivered in the template includes circuitry that can be used to test the CPRI IP Core. This circuitry includes 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 evaluation package 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 12.

Figure 12. Pattern Generators and Checkers Included with Delivered IP Core Testbench (One Direction Shown)



Reference Design Register Descriptions

There are no user accessible registers within the CPRI core. All control and status appear as internal I/O at the core boundary. Registers are provided in the evaluation capability 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 .

Note that these registers are equivalent to the registers provided for the basic CPRI configuration, which are given in the CPRI IP core user's guide, with a few additions. The additional registers for the low latency configuration are indicated in Table 16.

Table 16. CPRI Reference Design Register Map

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Control 0	0x800	1:0	LBR[1:0] - Line Bit Rate Control For maximum CPRI line rate enable by user	00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 3072 MHz
		2	FORCE_SM_STANDBY - Force startup state machine to standby	Zero to one transition
		4:3	PCS_SERDES_RATE[1:0] - Line rate supported by SERDES/PCS	00 = 614.4 MHz 01 = 1228.8 MHz 10 = 2457.6 MHz 11 = 3072 MHz
CPRI Control 1	0x801	0	TX_L1_RAI	Remote Action Indication
		1	TX_L1_SDI	SAP Defect Indication
CPRI Control 2	0x802	0	HDLC_240_EN	Enable = 1, Disable = 0
		1	HDLC_480_EN	Enable = 1, Disable = 0
		2	HDLC_960_EN	Enable = 1, Disable = 0
		3	HDLC_1920_EN	Enable = 1, Disable = 0
		4	HDLC_2400_EN	Enable = 1, Disable = 0
		5	TX_HYP_RST_EN	Enable = 1, Disable = 0
		6	TX_BFN_RST_EN	Enable = 1, Disable = 0
CPRI Control 3	0x803	5:0	[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	0	TX_ETH_EMPTY	Ethernet FIFO error bits
		1	TX_ETH_FULL	Ethernet FIFO error bits
		2	RX_ETH_EMPTY	Ethernet FIFO error bits
		3	RX_ETH_FULL	Ethernet FIFO error bits
		4	PRO_INTRUPT	Processor interrupt
		5	VER_NUM_ERR	Version number error bit
		6	RX_LOS	Rx loss of signal
		7	RX_LOF	Rx loss of frame
CPRI Status Reg 0	0x806	1:0	RATE_MODE	Final negotiated line bit rate
		7:4	lsm_state[1]	PCS RX external link state machine state
CPRI Status Reg 1	0x807	4:0	[0] - RX_L1_RST_RQSTACK [1] - RX_L1_RAI [2] - RX_L1_SDI [3] - RX_L1_LOS [4] - RX_L1_LOF	Received L1 inband protocol bits of byte Z.130.0
		5	VER_NUM_ERR	Version number error bit
		6	RX_LOS	Rx loss of signal
		7	RX_LOF	Rx loss of frame
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	2:0	TX_HDLC_MODE	Final negotiated HDLC bit rate.
		6:4	RX_L1_HDLC_MODE	Received L1 inband protocol bits of byte Z.66.0

Table 16. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
CPRI Status Reg 4	0x80b	2:0	CPRI_STUP_STATE	CPRI start up state
		3	rx_sync_status[1]	PCS RX sync status from the external link statue machine
		7:4	word_align_select[1]	PCS Word Aligner Selection Value (0-9)
Test Control	0x80c	0	REC_MD	REC Mode - 1=REC, 0=RE
		1	TEST_MD	Test mode - 1=short timer
		5	RXRST	RX Ethernet FIFO reset (debug only: active high)
		6	TXRST	TX Ethernet FIFO reset (debug only: active high)
		7	CORE_RESET	Used if core reset port not connected to GSR.
Test Loop Data Error Reg	0x80d	0	RX_ETH_PDO_DA_ERR	Test bench check loop back Ethernet data errors
		1	RX_IQ_DA_ERR	Test bench check loop back IQ data errors
		2	VENDOR_PDO_DA_ERR	Test bench check loop back vendor data errors
		3	RX_HDLC_PDO_DA_ERR	Test bench check loop back HDLC data errors
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.

Table 16. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
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.
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.

Table 16. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
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.
SOFTPCS Error Reg	0x81E	0	~rx_sync_status[1]	External link state machine not sync to the comma
		1	rx_pcs_cv	PCS code violation
		2	rx_pcs_disp_err	PCS disparity error

Table 16. CPRI Reference Design Register Map (Continued)

Register Name	Register Address	Bit Position	Bit Name	Description
R21 CPRI Delay Status Reg	0x81F	7:0	R21_cpri_coarse_timer[1]	R21 and CPRI coarse timer delay based on clock running 6 times core clock frequency.

1. Only supported for low latency configuration.

Technical Support Assistance

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

Revision History

Date	Version	Change Summary
June 2008	01.0	Initial release.
April 2009	01.1	Added support for LatticeECP3 FPGA family.
June 2009	01.2	Updates to LatticeECP3 low latency timing numbers.
July 2009	01.3	Updated LatticeECP3 low latency timing numbers based on ispLEVER 7.2 SP2.
		Reorganized the document for the LatticeECP2M and LatticeECP3 device families.
August 2009	01.4	Updated Multiple Channel Configuration Considerations bullets.
November 2009	01.5	Updated the latency numbers with 614Mbps line rate reference design generated in ispLEVER 8.0.